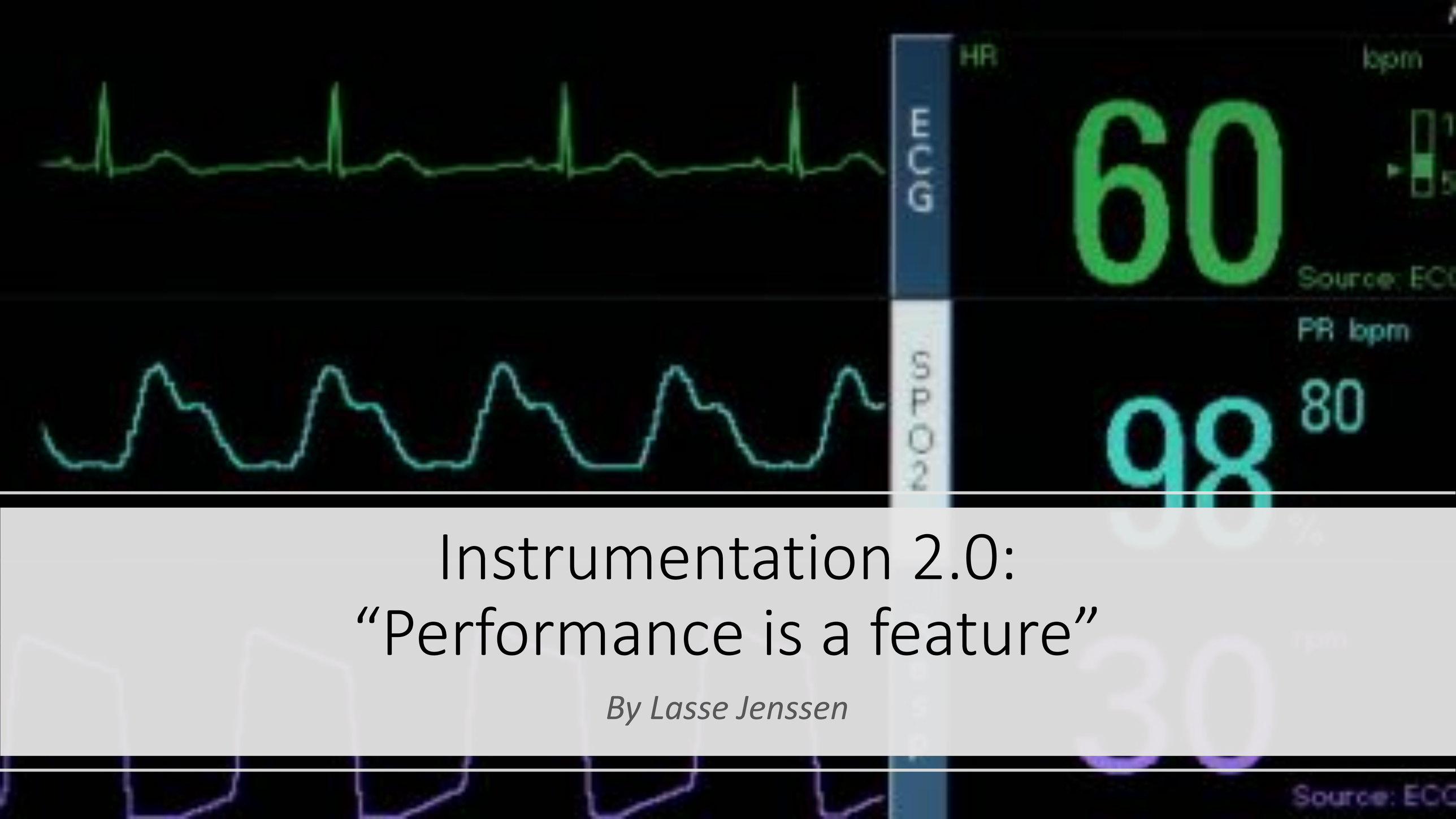


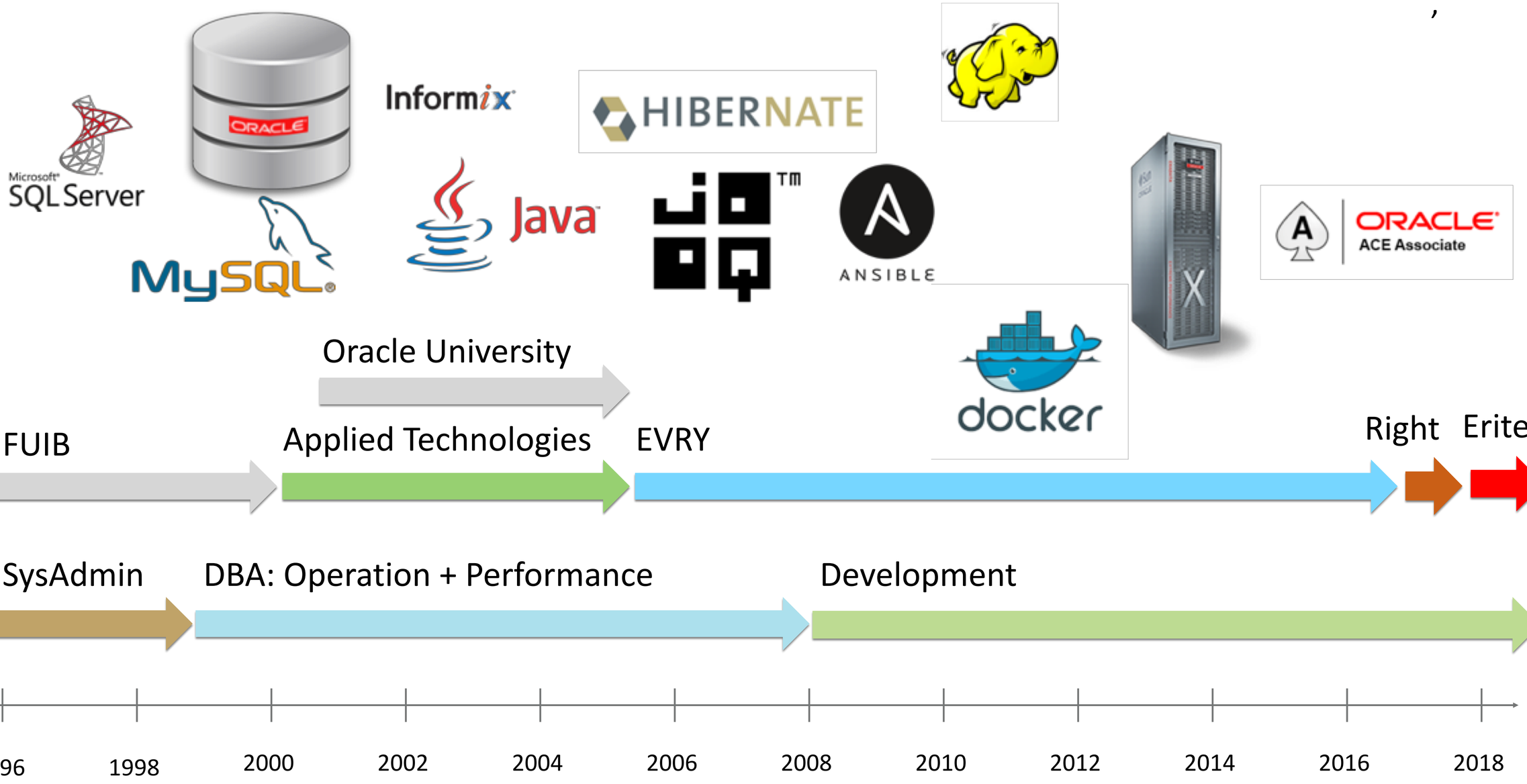
Instrumentation 2.0: “Performance is a feature”

By Lasse Jenssen

Lasse Jenssen

www.jcon.no/oracle

@lasjen - lasse.jenssen@eritec.no


Microsoft SQL Server
ORACLE
Informix
HIBERNATE
MySQL
Java
ANSIBLE
ORACLE ACE Associate
docker
Oracle University
FUIB
Applied Technologies
EVRY
Right
Eritec
SysAdmin
DBA: Operation + Performance
Development
1996
1998
2000
2002
2004
2006
2008
2010
2012
2014
2016
2018

Performance

Debugging

Method

Visibility & Insight

Forecasting

Why instrumentation?

"If you can't measure it, you can't manage it." – David Garvin

Oracle – builtin

Output

Logging

Counters

Oracle Trace

Instrumentation 1.0

“If you can’t measure it, you can’t manage it.” – David Garvin

Oracle – builtin

Output

Logging

Counters

Oracle Trace

Instrumentation 1.0

“If you can’t measure it, you can’t manage it.” – David Garvin

ORACLE®

- Out of the box

Oracle Wait Interface (OWI)

Oracle Trace

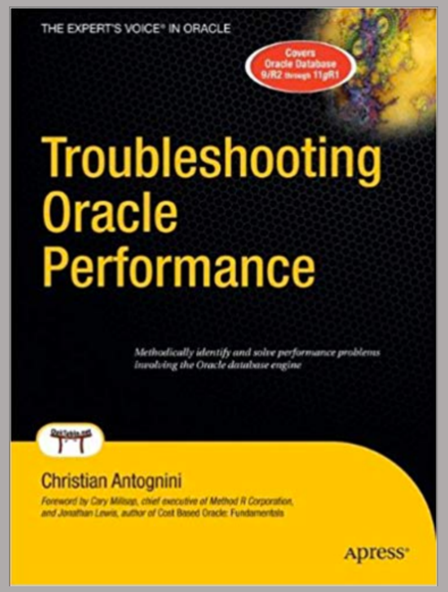
v$session, v$lock, v$latchholder

vsql, vactive_session_history

AWR, ADDM, Statspack

```
select
  sql_id, parsing_schema_name usr,
  round(s.buffer_gets/ case when s.executions = 0 then 1
                            else s.executions
                       end, 2)                                  gets_pr_exe,
  round(s.buffer_gets/
     case when s.rows_processed=0 and s.executions=0 then 1
          when s.rows_processed<s.executions         then s.executions
          else s.rows_processed end, 2)                    gets_pr_row,
  buffer_gets lio, executions nr_exe, rows_processed nr_rows, sql_text
from v$sql s
where 1=1
  and parsing_schema_name not in ('SYS')
  and sql_text not like 'DECLARE job BIN%'
  -- and sql_id not in ('6mcpb06rctk0x','5h5tk3d5gszky')
  -- and parsing_schema_name='HR'
order by 3 desc;
```

It depends!



Probably good – **less than about 5** gets_per_row

Probably acceptable – **up to about 10-20** gets_per_row

Probably inefficient – **more than about 15-20** gets_per_row

How many tables are being joined? Range Scans vs Equality Scans? etc.

SQL_ID	CH...	GETS_PER_EXE	GETS_PER_ROW	EXECUTIONS	ROWS_P...	OUTL...	SQL_TEXT
bna10qw5nhqcv	0	279778,95	279778,95	5767	5767	(null)	/* select
g3ykm0ypx8jpx	5	23494,46	12187216	5705	10	(null)	/* schema=
c9wrdbn75w1mt	0	13908,18	19436,06	40630	29074	(null)	/* schema=
g3ykm0ypx8jpx	3	1484,99	1438,71	14142	14597	(null)	/* schema=
6g9ffmmbwzx2n	1	1175,82	146,57	44	360	(null)	/* load co
73g0kvg9dvg97	0	1096,36	91,6	870	10424	(null)	/* load co
fmxzd7uvdc497	0	1085,74	168,01	92	600	(null)	/* load co
g509nz08aq4qd	0	823,18	1117,06	706	520	(null)	/* schema=
4ttaufcaza07n	1	819,18	889,75	352	324	(null)	/* schema=
7v5tpy0zkn0m4	1	624,01	30384,62	632	12	(null)	/* criteri
btyr3gryagmfw	0	360,91	360,91	2829	0	(null)	/* select
7qz15ftr29vy4	0	360,9	360,9	2883	0	(null)	/* select
44br57re853dz	0	52,88	6,87	735129	5658376	(null)	/* schema=
fxanr8k2a17xj	0	32,85	32,85	322	322	(null)	/* insert
fxanr8k2a17xj	2	32,04	32,04	12843	12843	(null)	/* insert

Note! These queries WERE “fast enough” – except in some rear occasions.

Oracle – builtin

Output

Logging

Counters

Oracle Trace

Instrumentation 1.0

“If you can’t measure it, you can’t manage it.” – David Garvin

```
create procedure doSomething(a_i IN number, b_i IN number) {

   doPart1(a_i);
   doPart2(b_i);

}
/

set timing on
begin
   doSomething(1,2);
end;
/

PL/SQL procedure successfully completed.

Elapsed: 00:00:04.329
```

Where is time being spent?

“If you can’t measure it, you can’t manage it.” – David Garvin

```
procedure out(l_text IN varchar2) as
begin
   dbms_output.put_line(l_text);
end;
```

Instrumentation: Output

```
create procedure doSomething(l_sec1 IN number, l_sec2 IN number) as
   t0 pls_integer;
   t1 pls_integer;
   t2 pls_integer;
   t3 pls_integer;
begin
   out('STARTED  doSomething: '|| to_char(sysdate,'HH24:MI:SS'));
   out('BIND: l_sec1=' || l_sec1 || ', l_sec2='|| l_sec2 );

   t0 := dbms_utility.get_time*10;
   doPart1(l_sec1);
   t1 := dbms_utility.get_time*10;
   doPart2(l_sec2);
   t2 := dbms_utility.get_time*10;

   out('doPart1 - Completed : ' || to_char(t1-t0) || ' ms');
   out('doPart2 - Completed : ' || to_char(t2-t1) || ' ms');
   out('FINISHED doSomething: ' || to_char(t2-t0) || ' ms');
end;
```

Instrumentation: Output

```
STARTED  doSomething: 21.01.2019 12:46:43
Variables: l_sec1=1, l_sec2=1
doPart1 - Completed :  102 ms
doPart2 - Completed : 4219 ms
FINISHED doSomething: 4323 ms


PL/SQL procedure successfully completed.

Elapsed: 00:00:04.329
```

Oracle – builtin

Output

Logging

Counters

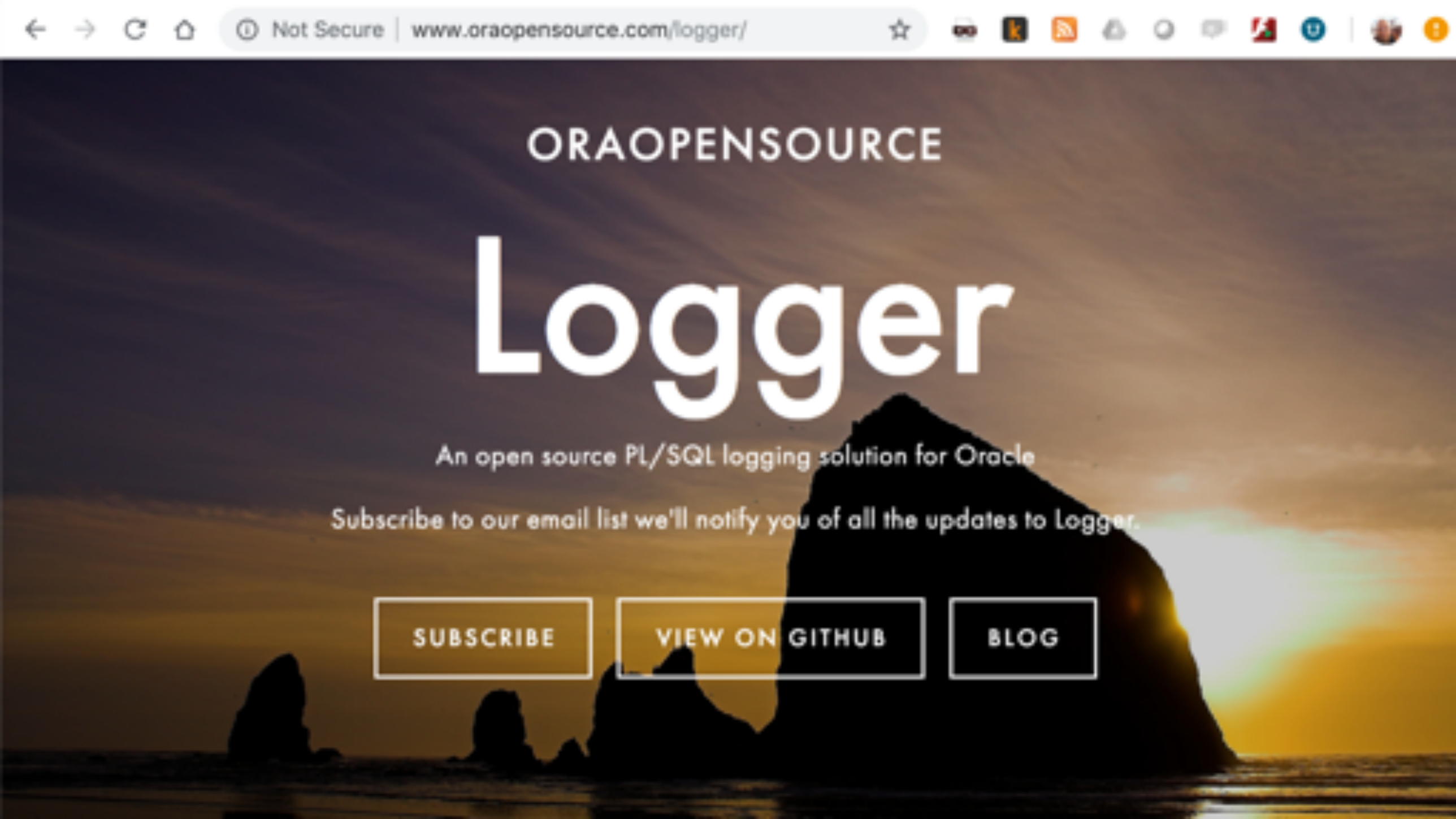
Oracle Trace

Instrumentation 1.0

"If you can't measure it, you can't manage it." – David Garvin

```
public void doSomething(int a, int b) {
    Logger log = LoggerFactory.getLogger(Demo.class);
    int t0, t1, t2, t3;
    t0 = TimeUtil.gettime()*10;
    doPart1(a,b);
    t1 = TimeUtil.gettime()*10;
    doPart2(a,b);
    t2 = TimeUtil.gettime()*10;

    if (debug || t3-t0 > 3000) {
       log.info("Total    : " + t3-t0 + " ms");
       log.info("*doPart1: " + t1-t0 + " ms");
       log.info("*doPart2: " + t2-t1 + " ms");
    }
}
```

```
28-11-2018 13:32:18.539 [main] INFO no.eritec.Demo.doSomething - Total   : 4323ms
28-11-2018 13:32:18.539 [main] INFO no.eritec.Demo.doSomething - *doPart1:  102ms
28-11-2018 13:32:18.539 [main] INFO no.eritec.Demo.doSomething - *doPart2: 4219ms
```


Not Secure
www.oraopensource.com/logger/
ORAOPENSOURCE
Logger
An open source PL/SQL logging solution for Oracle
Subscribe to our email list we'll notify you of all the updates to Logger.
SUBSCRIBE
VIEW ON GITHUB
BLOG

GitHub, Inc. [US] | https://github.com/OraOpenSource/Logger

Search or jump to... / Pull requests Issues Marketplace Explore

OraOpenSource / Logger

forked from tmuth/Logger---A-PL-SQL-Logging-Utility

Unwatch 62 | Unstar 197 | Fork

Code | Issues 97 | Pull requests 7 | Projects 1 | Wiki | Insights

Logger is used by Oracle developers to instrument their PL/SQL code http://www.oraopensource.com/logger/

334 commits | 2 branches | 7 releases | 8 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or dow

This branch is 201 commits ahead, 1 commit behind tmuth:master. | Pull request

martindsouza doc fixes — Latest commit 600f0c4 on Jun

build	#127: Add logger_prefs.pref_type	4 ye
demos	#127: Add logger_prefs.pref_type	4 ye

```
$ sql system@localhost:1522/ORCL

SQLcl: Release 4.2.0 Production on Mon Jan 21 13:28:33 2019
…

SQL> grant connect, create view, create job, create table, create sequence,
create trigger, create procedure, create any context to devdata;

Grant succeeded.


$ cd logger_3.1.1/
$ sql devdata@localhost:152/ORCL

SQLcl: Release 4.2.0 Production on Mon Jan 21 13:28:33 2019
…

SQL> @logger_install.sql

_____________________________________________________________________________
User has all required privileges, installation will continue.
_____________________________________________________________________________
```

```
begin
   logger.log('This is a debug message. (level = 16)’);
   logger.log_information('This is an informational message. (level = 8)');
   logger.log_warning('This is a warning message. (level = 4)');
   logger.log_error('This is an error message (level = 2)');
   logger.log_permanent('This is a permanent message, good for upgrades and ' ||
                        'milestones. (level = 1) ');
end;
/

select id, logger_level level, text from logger_logs_5_min order by id;

  ID LEVEL TEXT
---- ----- ----------------------------------------------------------
  10    16 This is a debug message. (level = 16)
  11     8 This is an informational message. (level = 8)
  12     4 This is a warning message. (level = 4)
  13     2 This is an error message (level = 2)
  14     1 This is a permanent message, good for upgrades and milestones.(level = 1)
```

```
create procedure doSomething(l_sec1 IN number, l_sec2 IN number) as
   t0 pls_integer;
   t1 pls_integer;
   t2 pls_integer;
   t3 pls_integer;
begin
   logger.log('START doSomething: l_sec1='||l_sec1||', l_sec2='||l_sec2 );

   t0 := dbms_utility.get_time*10;
   doPart1(l_sec1);
   t1 := dbms_utility.get_time*10;
   doPart2(l_sec2);
   t2 := dbms_utility.get_time*10;

   logger.log('SUB=> doPart1 - Completed : ' || to_char(t1-t0) || ' ms');
   logger.log('SUB=> doPart2 - Completed : ' || to_char(t2-t1) || ' ms');
   logger.log('FINISHED doSomething: Total ' || to_char(t2-t0) || ' ms');
end;
```

```
SELECT id, logger_level "LEVEL", text,
       to_char(time_stamp,'HH24:MI:SS.SS') "TIME_STAMP",
       scope, unit_name, extra, sid
FROM logger_logs ORDER BY id;
```

TABLE: logger_logs

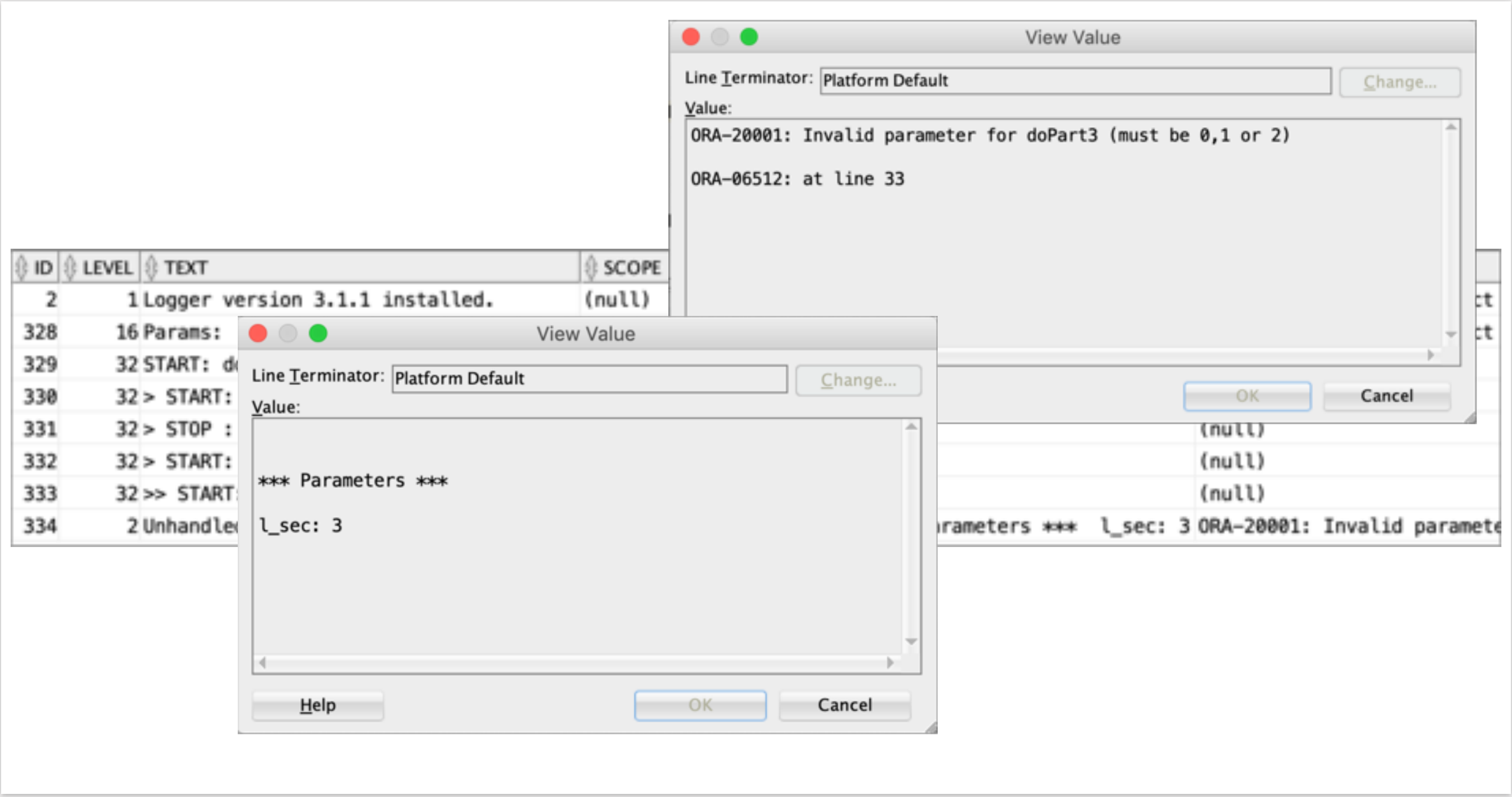
https://github.com/oraopensource/logger/

ID	LEVEL	TEXT	TIME_STAMP	SCOPE	UNIT_NAME	EXTRA	SID
2	1	Logger version 3.1.1 installed.	12:29:03.03	(null)	(null)	(null)	268
177	16	START doSomething: l_sec1=1, l_sec2=2	15:01:54.54	(null)	BLOCK	(null)	142
178	16	SUB=> doPart1 - Completed : 110 ms	15:01:58.58	(null)	BLOCK	(null)	142
179	16	SUB=> doPart2 - Completed : 4250 ms	15:01:58.58	(null)	BLOCK	(null)	142
180	16	FINISHED doSomething: Total 4360 ms	15:01:58.58	(null)	BLOCK	(null)	142

```
create procedure doSomething(l_sec1 IN number, l_sec2 IN number) as
   l_scope  logger_logs.scope%type := 'doSomething';
   l_params logger.tab_param;
begin
   logger.append_param(l_params, 'l_sec1', l_sec1);
   logger.append_param(l_params, 'l_sec2', l_sec2);

   logger.log('START', l_scope, null, l_params);
   logger.time_reset;
   logger.time_start('doSomething');

   doPart1(l_sec1);

   doPart2(l_sec2);

   logger.time_stop('doSomething');
   logger.log('END', l_scope);
end;
```

	LEVEL	TEXT	TIME_STAMP	SCOPE	UNIT_NAME	EXTRA
2	1	Logger version 3.1.1 installed.	12:29:03.0303	(null)	(null)	(null)
308	16	Params:	11:03:07.0707	dosomething	BLOCK	*** Parameters *** l_sec1:...
309	32	START: doSomething	11:03:07.0707	(null)	DOSOMETHING	(null)
310	32	> START: doPart1	11:03:07.0707	(null)	DOPART1	(null)
311	32	> STOP : doPart1 - 00:00:00.106783	11:03:07.0707	(null)	DOPART1	(null)
312	32	> START: doPart2	11:03:07.0707	(null)	DOPART2	(null)
313	32	>> START: doPart3	11:03:07.0707	(null)	DOPART3	(null)
314	32	>> STOP : doPart3 - 00:00:04.231723	11:03:11.1111	(null)	DOPART3	(null)
315	32	> STOP : doPart2 - 00:00:04.23593	11:03:11.1111	(null)	DOPART2	(null)
316	32	STOP : doSomething - 00:00:04.350082	11:03:11.1111	(null)	DOSOMETHING	(null)
317	16	END	11:03:11.1111	dosomething	BLOCK	(null)

```
procedure doPart3(l_sec IN number) as
  l_scope logger_logs.scope%type := 'doPart3';
  l_params logger.tab_param;
begin
  logger.time_start(l_scope);
  logger.append_param(l_params, 'l_sec', l_sec);

  if l_sec < 0 or l_sec > 2 then
     RAISE_APPLICATION_ERROR(-20001,'Invalid parameter (must be 0-2)');
  end if;

  dbms_lock.sleep(l_sec*2.1);

  logger.time_stop(l_scope);
exception when others then
   logger.log_error('Unhandled Exception', l_scope, null, l_params);
   raise;
end;
```

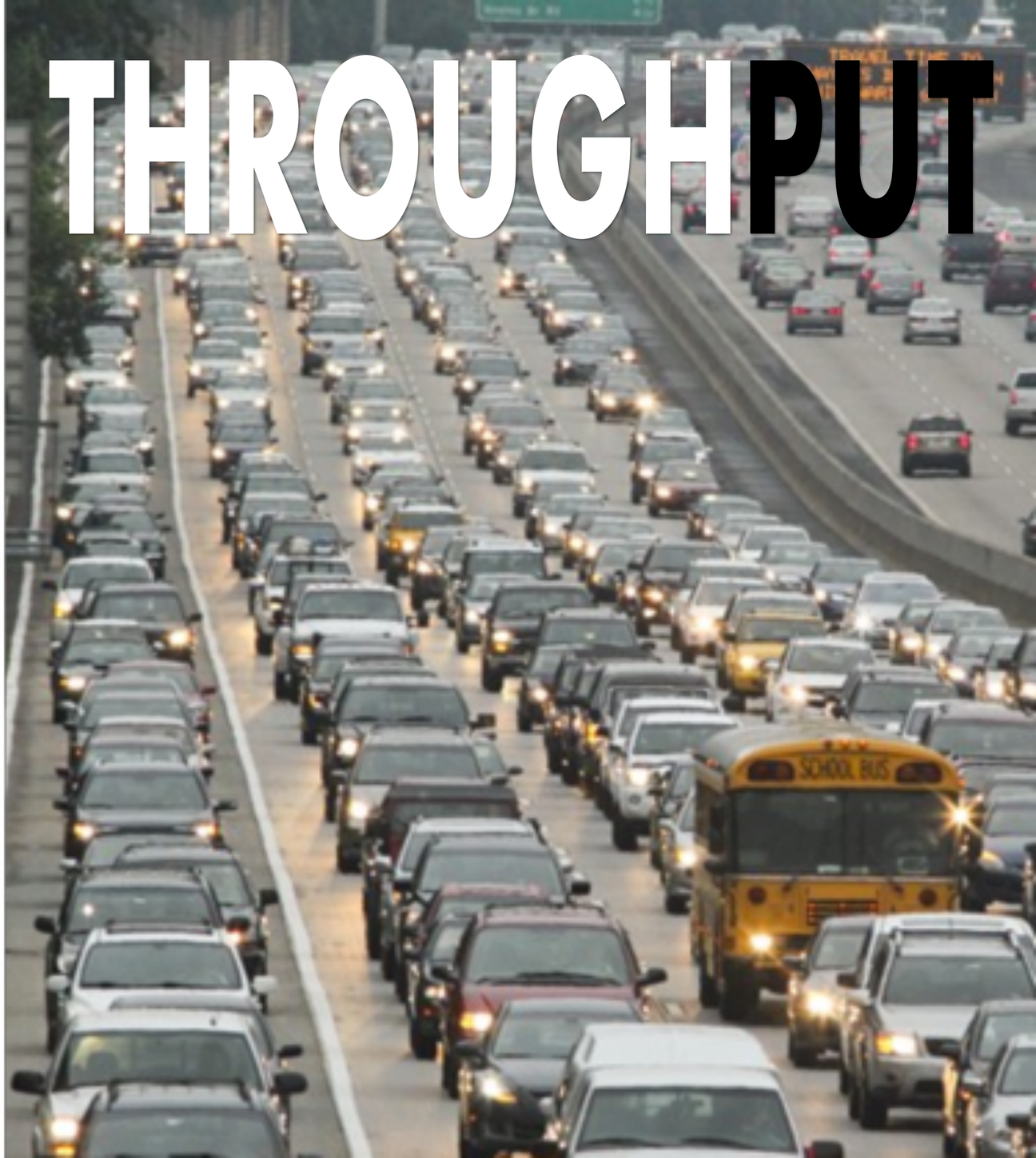

View Value
Line Terminator: Platform Default
Change...
Value:
ORA-20001: Invalid parameter for doPart3 (must be 0,1 or 2)
ORA-06512: at line 33
OK
Cancel
View Value
Line Terminator: Platform Default
Change...
Value:
*** Parameters ***
l_sec: 3
Help
OK
Cancel
ID LEVEL TEXT SCOPE
2 1 Logger version 3.1.1 installed. (null)
328 16 Params:
329 32 START: d
330 32 > START:
331 32 > STOP :
332 32 > START:
333 32 >> START
334 2 Unhandle
(null)
(null)
(null)
rameters *** l_sec: 3 ORA-20001: Invalid paramete

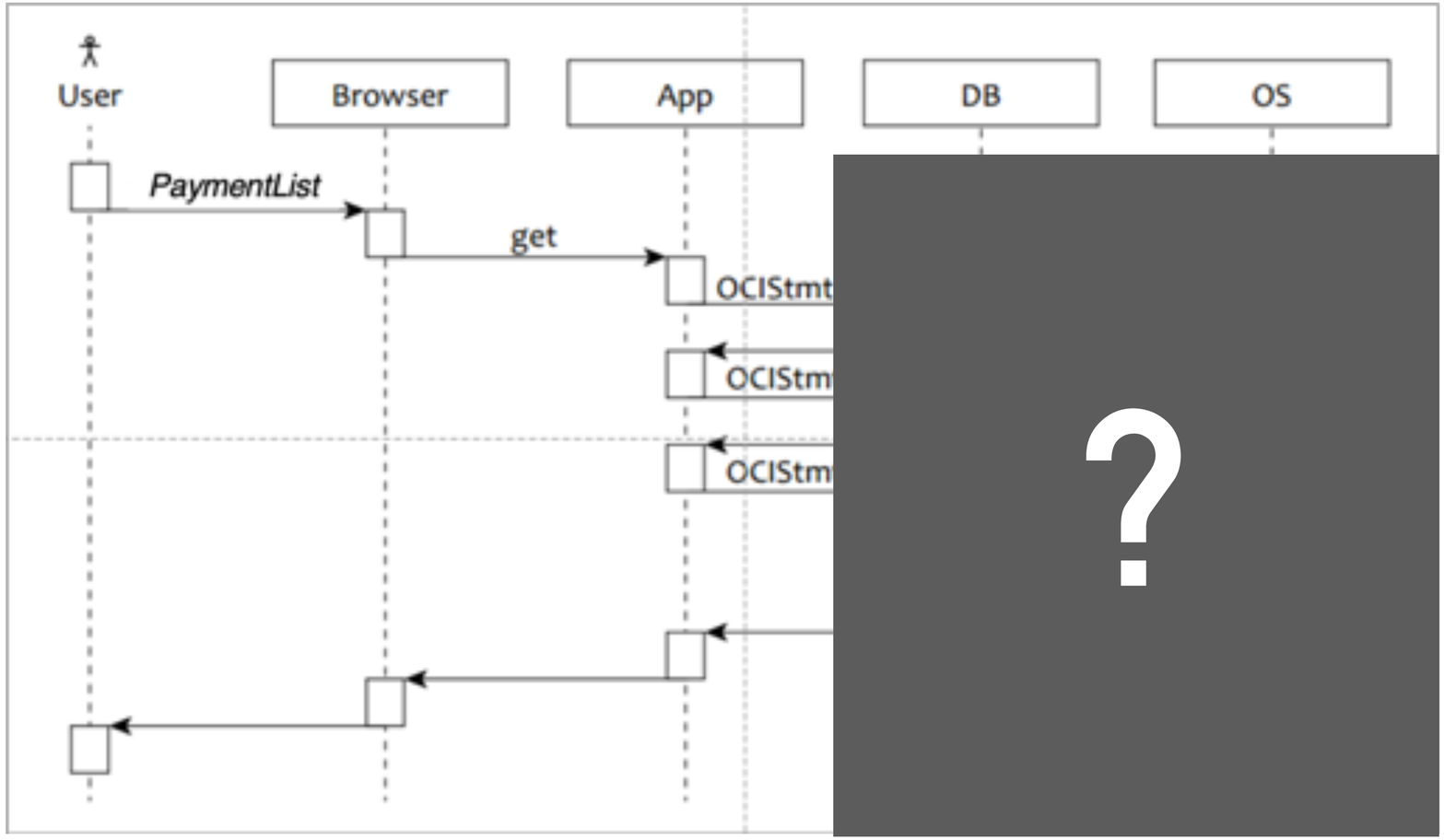
```
exec logger.purge_all;

exec logger.set_level (logger.g_error_name);
```

ID	LEVEL	TEXT	TIME_STAMP	SCOPE	UNIT_NAME	EXTRA	SID
2	1	Logger version 3.1.1 installed.	12:29:03.0303	(null)	(null)	(null)	268
335	8	Log level set to ERROR for client_id...	08:45:18.1818	(null)	(null)	(null)	142
336	2	Unhandled Exception ORA-20001: Inva...	08:45:39.3939	dopart3	BLOCK	*** Parameters *** l_sec: 3	142

“Performance is measured in terms of …


RESPONSE
TIME


THROUGHPUT
SCHOOL BUS



“Users feel the variance, not the mean”

https://www.ge.com/sixsigma/SixSigma.pdf

End-to-end Metrics

Oracle Extended Trace

Instrumentation 2.0

"Performance is a feature." – Cary Millsap

Oracle End-to-End Metrics

MODULE ACTION CLIENTID

v$session v$sql

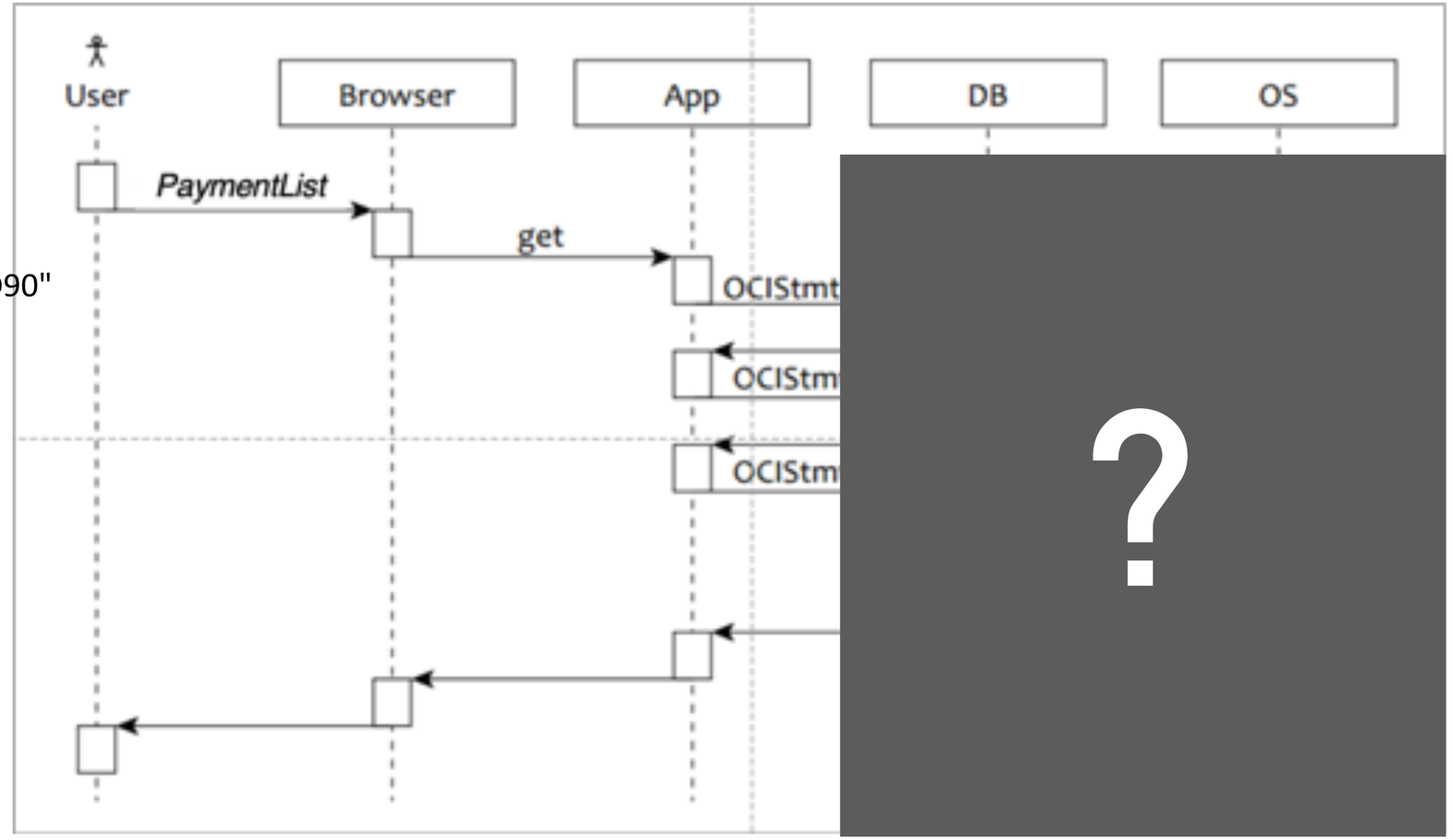
v$sqlarea v$sqlarea_plan_hash

v$active_session_history

v$sql_monitor

ExperienceID

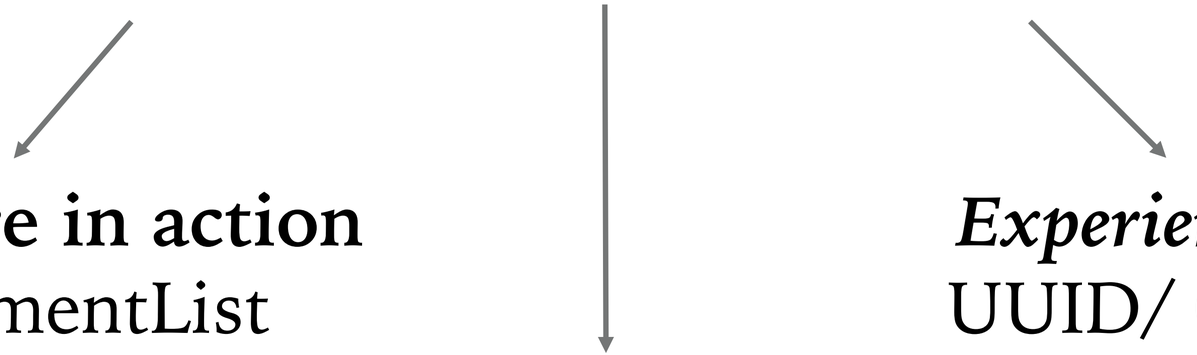
"08B77430-B717-4ACA-A092-BD970987AD90"



“Users feel the variance, not the mean”

https://www.ge.com/sixsigma/SixSigma.pdf

Module, Action, Clientid



Feature in action
PaymentList

ExperienceID
UUID/ GUID

User
lasse.jenssen@eritec.no

```
Connection conn = datasource.getConnection();
Properties p = new Properties();

UUID exerience_id = UUID.randomUUID();

p.add("OCSID.CLIENTID”, experience_id().toString());
p.add("OCSID.MODULE",   "PaymentList");
p.add("OCSID.ACTION",   "lasse@eritec.no");

conn.setClientInfo(p);

… <application code>

conn.close;
```

```
BEGIN
   DBMS_APPLICATION_INFO.set_module(
      module_name => 'PaymentList',
      action_name => 'lasse@eritec.no');

   DBMS_SESSION.set_identifier(SYS_GUID());

   <application code>

   DBMS_APPLICATION_INFO.set_module(
      module_name => ' ' ,
      action_name => ' ');

   DBMS_SESSION.set_identifier(' ');
END;
```

```
procedure doSomething(...) as
begin
    <my code>;
end;
```

Application (JDBC)

If PLSQL –
Use rapper code

Where should you set the metrics?

"If you can't measure it, you can't manage it." – David Garvin

```
procedure doSomethingM(...) as
begin
    setMetrics(...);
    doSomething(...);
    resetMetrics(...);
end;
```

Visibility & Insight

```
SELECT * FROM v$session   WHERE module='PaymentList';

SELECT * FROM v$sql       WHERE module='PaymentList';

SELECT * FROM logger_logs WHERE module='PaymentList'
ORDER by client_identifier, id;
```

```
exec dbms_monitor.serv_mod_act_trace_enable(
        service_name => 'PAY_SRV',
        module_name  => 'PaymentList',
        action_name  => 'lasse@eritec.no',
        waits        => TRUE,
        binds        => TRUE,
        plan_stat    => 'ALL_EXECUTIONS' );

select * from dba_enabled_traces;
```

```
*** 2014-09-24 12:25:11.332
*** SESSION ID:(56.4539) 2014-09-24 12:25:11.332
*** CLIENT ID:(067e6162-3b6f-4ae2-a171-2470b63dff00) 2014-09-24 12:25:11.332
*** SERVICE NAME:(pay_srv) 2014-09-24 12:25:11.332
*** MODULE NAME:(PaymentList) 2014-09-24 12:25:11.332
*** ACTION NAME:(lasse@eritec.no) 2014-09-24 12:25:11.332
*** CONTAINER ID:(3) 2014-09-24 12:25:11.332


EXEC #140491268512760:c=0,e=55,p=0,cr=0,cu=0,mis=0,r=0,dep=0,og=0,…
WAIT #140491268512760: nam='SQL*Net message to client' ela= 1 driver id=…
WAIT #140491268512760: nam='SQL*Net message from client' ela= 137 driver …
=====================
…
```

> *trcsess output=myfile01.trc clientid="08B77430-B717-4ACA-A092-BD970987AD90" **

Profile by Subroutine

e following table shows the decomposition of total task response time by internal subroutine.

	Subroutine	Duration seconds	Duration % R	Cumulative duration seconds	Cumulative duration % R	Call count	Duration per call (seconds) mean	min	skew	max
1.	db file sequential read	0.129	46.3%	0.129	46.3%	3	0.042887	0.021816		0.053708
2.	unaccounted-for within dbcalls	0.065	23.5%	0.194	69.8%	330	0.000198	-0.000109		0.012987
3.	CPU service, EXEC calls	0.059	21.3%	0.253	91.1%	89	0.000665	0.000000		0.034788
4.	CPU service, PARSE calls	0.008	2.9%	0.261	94.1%	16	0.000506	0.000236		0.000981
5.	SQL*Net message from client	0.007	2.7%	0.269	96.7%	6	0.001229	0.000000		0.001979
6.	CPU service, FETCH calls	0.004	1.3%	0.272	98.0%	182	0.000020	0.000007		0.000380
7.	unaccounted-for between dbcalls	0.003	1.1%	0.276	99.1%	15	0.000207	0.000018		0.001521
8.	row cache lock	0.002	0.6%	0.277	99.7%	6	0.000279	0.000003		0.000971
9.	rdbms ipc reply	0.000	0.2%	0.278	99.9%	4	0.000112	0.000048		0.000274
10.	CPU service, unreported call(s)	0.000	0.0%	0.278	100.0%	1	0.000125	0.000125		0.000125
11.	CPU service, CLOSE calls	0.000	0.0%	0.278	100.0%	86	0.000001	0.000000		0.000038
12.	SQL*Net message to client	0.000	0.0%	0.278	100.0%	6	0.000002	0.000001		0.000003
13.	Total	0.278	100.0%							

ECID

DBOP

Real-Time
SQL Monitoring

Instrumentation 2.1

"Performance is a feature." – Cary Millsap

ECID – Execution Context ID

Oracle Fusion Middelware

Oracle Dynamic Monitoring Service (DMS)

Oracle HTTP server

Oracle Weblogic

Oracle Database

```
public void setMetrics(Connection c) throws SQLException {
    Properties p = new Properties();
    p.setProperty("OCSID.MODULE", l_feature);
    p.setProperty("OCSID.ACTION", l_user);
    p.setProperty("OCSID.CLIENTID", UUID.randomUUID().toString());
    p.setProperty("OCSID.ECID",   l_context);
    c.setClientInfo(p);
}
```

v$session v$active_session_history v$sql_monitor

Execution Context Id (ECID)

Write ECID to the tracefile when enabling Oracle Extended Trace

Created on Feb 1, 2019 6:50 AM by Lasse Jenssen - Last Modified: Feb 1, 2019 6:50 AM

10 You have voted up. ACTIVE

The ECID content should be written to the Oracle Trace file when enabling Oracle Extended Trace through the DBMS_MONITOR.serv_mod_act_trace_enable procedure.
The ECID should be written to the trace file in the same manner as the MODULE, ACTION and CLIENTID tags.

Tags (edit):

0 Comments

ADD A COMMENT

ECID

DBOP

Real-Time
SQL Monitoring

Instrumentation 2.1

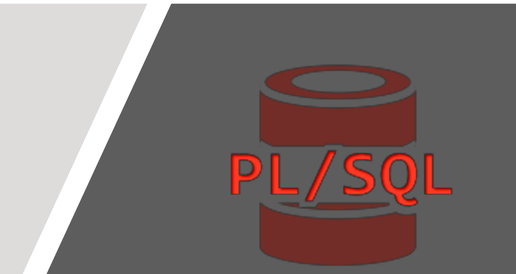
"Performance is a feature." – Cary Millsap

```
public void setMetrics(Connection c) throws SQLException {
    Properties p = new Properties();

    p.setProperty("OCSID.MODULE", l_feature);
    p.setProperty("OCSID.ACTION", l_user);
    p.setProperty("OCSID.CLIENTID", UUID.randomUUID().toString());
    p.setProperty("OCSID.ECID",   l_context);
    p.setProperty("OCSID.DBOP",   l_feature);

    c.setClientInfo(p);
}
```

Real-time SQL Monitoring

```
DECLARE
    l_dbop_eid number;
BEGIN
    l_dbop_eid := DBMS_SQL_MONITOR.begin_operation (
        dbop_name => 'findAllUsers',
        dbop_eid  => l_dbop_eid,
        forced_tracking => DBMS_SQL_MONITOR.force_tracking );

    <code>

    DBMS_SQL_MONITOR.end_operation (
        dbop_name => 'findAllUsers',
        dbop_eid  => l_dbop_eid );
END;
/
```

Real-time SQL Monitoring



```
SELECT DBMS_SQL_MONITOR.report_sql_monitor(
  dbop_name     => 'findAllUsers',
  type          => 'TEXT',
  report_level => 'ALL') AS report
FROM dual;
```

Status	Duration	SQL Id or DBOP Name	Exec Id	Start	User	Module/Action	Dop	DB Time	IOs	SQL Text
DONE	5.0s	findAllUsers	125	01/23/2019 07:20:39	DEMO	HibernateTest/findAllUsers		0.02s		
DONE	10s	findAllUsers	118	01/23/2019 07:06:21	DEMO	HibernateTest/findAllUsers		0.02s		
DONE	10s	findAllUsers	111	01/23/2019 07:02:40	DEMO	HibernateTest/findAllUsers		0.01s		
DONE	10s	findAllUsers	104	01/23/2019 06:56:54	DEMO	HibernateTest/findAllUsers		0.02s		
DONE	9.0s	findAllUsers	97	01/23/2019 06:56:08	DEMO	HibernateTest/findAllUsers		0.01s		
DONE	10s	findAllUsers	90	01/23/2019 06:54:08	DEMO	HibernateTest/findAllUsers		0.02s		
DONE	10s	findAllUsers	83	01/23/2019 06:48:25	DEMO	HibernateTest/findAllUsers		0.04s		
DONE	10s	findAllUsers	69	01/22/2019 22:46:50	DEMO	HibernateTest/findAllUsers		0.02s		
DONE	10s	findAllUsers	65	01/22/2019 22:46:24	DEMO	HibernateTest/findAllUsers		0.02s		
DONE	10s	findAllUsers	46	01/22/2019 22:43:06	DEMO	HibernateTest/findAllUsers		0.02s		
DONE	11s	findAllUsers	42	01/22/2019 22:41:59	DEMO	HibernateTest/findAllUsers		0.02s		
DONE	10s	findAllUsers	19	01/22/2019 22:16:51	DEMO	HibernateTest/findAllUsers		0.02s		
DONE	10s	findAllUsers	13	01/22/2019 22:15:06	DEMO	HibernateTest/findAllUsers		0.05s		

Real-time SQL Monitoring

```
SELECT *
FROM v$sql_monitor WHERE dbop_name = 'findAllUsers';
```

KEY	REPORT_ID	STATUS	USER#	USERNAME	MODULE	ACTION	SERVICE_NAME	CLIENT_IDENTIFIER
273	0	EXECUTING	109	DEMO	HibernateTest	findAllUsers	orcl	8e314bdf-28d4-4b93-b462-354c1f44832f
38654706118	341	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	6d1f9a2e-df6f-42d3-9c6f-8f2076513f92
38654706044	288	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	14d1b1ca-5cc6-4d36-b2de-c7ceb01d85c9
64424509989	0	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	34bd6114-a1d7-4838-9ef9-67f1fea96d3a
68719477095	275	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	077e2d50-cb21-465a-84fe-226e632d2bfa
120259084671	289	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	bcf4220b-d7c3-4db2-a0f2-63305b0e93da
124554052052	355	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	42d54186-60f1-4b37-9688-5435bdec798c
128849019317	332	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	246d2dbb-82c6-405a-b42c-d7d77f541d19
150323855783	312	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	b1c612fb-599a-4b6b-8072-e346068b344a
326417514957	351	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	e0ca7cab-d4f8-499f-882c-390e6dbbb51c
335007449358	238	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	175d3c15-0b24-45ba-affe-1b6847b5291e
343597383956	244	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	4f89f20d-9757-40fc-9355-63d4d6597225
403726926268	331	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	c431b2a9-1291-403d-9de6-9bec167c1f13
433791697326	323	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	05877d94-6292-4c3f-8842-e218a74ee096
438086664554	278	DONE	109	DEMO	HibernateTest	findAllUsers	orcl	fc59befe-988d-49c4-8f55-707fdf72fe8f

Real-time SQL Monitoring

```
DECLARE
   l_dbop_eid number;
BEGIN
   l_dbop_eid := DBMS_SQL_MONITOR.begin_operation (
       dbop_name => 'testDBOP', dbop_eid  => l_dbop_eid,
       forced_tracking => DBMS_SQL_MONITOR.force_tracking );

   for rec in (SELECT /*+ MONITOR */ FROM dual) loop
      null;
   end loop;

   DBMS_SQL_MONITOR.end_operation (
       dbop_name => 'testDBOP', dbop_eid  => l_dbop_eid );
END;
/
```

/*+ MONITOR */ hint

REPORT_ID	USERNAME	CLIENT_IDENTIFIER	DBOP_EXEC_ID	DBOP_NAME	IN_DBO..	IN_DBOP_NAME	SQL_TEXT
526	DEVDATA	80A30388289AB0CBE053020011AC4E63	0	(null)	174	testDBOP	SELECT /*+ MONITOR */ * FROM DUAL
525	DEVDATA	80A30388289AB0CBE053020011AC4E63	174	testDBOP	0	(null)	(null)
524	DEVDATA	80A303882899B0CBE053020011AC4E63	0	(null)	173	testDBOP	SELECT /*+ MONITOR */ * FROM DUAL
523	DEVDATA	80A303882899B0CBE053020011AC4E63	173	testDBOP	0	(null)	(null)

/*+ MONITOR */ hint

ElasticSearch

LogStash

Kibana

Instrumentation 3.0

"Performance is a feature." – Cary Millsap

HTTP server

Oracle Weblogic

Oracle Database

elasticsearch

https://www.elastic.co/blog/visualising-oracle-performance-data-with-the-elastic-stack

ASH detail

1 2 3 4 5 ...10 »

Time	machine	sql_opname	sql_plan_operation	sql_plan_options	wait_class	session_state	program	module	action
February 4th 2016, 23:05:48.420	asgard.local	INSERT	-	-	-	ON CPU	JDBC Thin Client	New Order	-
February 4th 2016, 23:05:48.420	asgard.local	SELECT	TABLE ACCESS	BY INDEX ROWID	-	ON CPU	JDBC Thin Client	Browse Products	getProductDetails
February 4th 2016, 23:05:44.410	asgard.local	-	-	-	Commit	WAITING	JDBC Thin Client	-	-
February 4th 2016, 23:05:43.410	asgard.local	INSERT	LOAD TABLE CONVENTIONAL	-	-	ON CPU	JDBC Thin Client	New Order	-
February 4th 2016, 23:05:38.400	asgard.local	-	-	-	-	ON CPU	JDBC Thin Client	-	-

“Performance is a feature!”

Instrument your code:

- Oracle End-to-end Metrics
- Logging (both application and database)

Need to decide:

- What should always be logged (application, errors)
- How to turn on logging when needed (database logging)

lasse.jenssen@eritec.no
Twitter: @lasjen
http://www.eritec.no