



Instrumentation

Why You Should Care

Cary Millsap

[@CaryMillsap](#) / cary.millsap@method-r.com

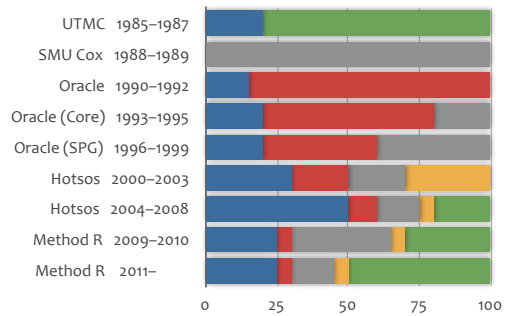
New York Oracle Users Group
New York City
1:30p–2:30p Thursday 22 September 2011

© 2011 Method R Corporation

1

Cary Millsap

■ Teaching ■ Consulting ■ Business ■ Method ■ Software



2

Performance is a feature

3

How many orders did you book between 10am and 11am on November 4?

What percentage of “oe book” clicks have run in less than 1 second since August 17?

Morton’s 1-line “oe book” ran in 1 second. Nancy’s ran in 10 seconds. Why?

Is there a user or location that experiences particularly good or bad response times?

Which tasks have the most execution response time variance?

What is your best hour of “oe book” throughput since January 1? Your worst hour?

What tasks competed against “oe book” during its worst-throughput hour?

99% of “oe book” clicks must run in less than 1 second,
and 99.99% must run in less than 5 seconds. Do they?

When will your 99th percentile “oe book” response time exceed your 1-sec tolerance?

4

You can **know.**

5

But it takes **work.**

6

Performance is a **feature**.

You have to...

Design it.

Code it.

Test it.

Buy hardware for it.

Manage it.

7

If you have a **prepackaged** application,

You have to...

Trust that your vendor did it right.

Or **convince** your vendor to do it right.

Or **retrofit** your vendor's software.

Buy hardware for it.

Manage it.

8

Measuring performance

9

Quiz time

Your system CPU utilization is 96%.

Is that **good performance?**
or **bad performance?**

10

Maybe?

System busy doing work.
Response times are fine.

Maybe not?

Lots of programs using PX.
Response times are horrible.

11

good performance

bad performance

Your system CPU utilization is 96%.

12

The key to measuring performance:

measure **task** executions

The machine can't know what your **tasks** are unless you tell it.

17

What is a **task**?

a click
a report
a batch job
part of a batch job
...

18

performance

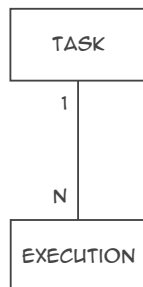
relationship of task **executions** to time

response time

time elapsed to **execute** a task

throughput

task **executions** per unit of time



task

id	mod	act
1	oe	book
2	oe	ship
3	oe	pick
4	pa	mtch
5	pa	corr
6	pa	reco

execution

id	task_id	user	ip	begin	end	error	work
...
189672	1	nwells	10.17.21.71	2011-08-21T08:14:22.307	2011-08-21T08:14:22.807		28
189673	2	rroberts	10.17.21.231	2011-08-21T08:14:23.118	2011-08-21T08:14:23.129		2
189674	5	bhamza	10.17.21.79	2011-08-21T08:14:23.051	2011-08-21T08:14:24.943	1555	3018
189675	3	jping	10.17.21.148	2011-08-21T08:14:23.381	2011-08-21T08:16:01.894		298
189676	1	zmuller	142.19.1.107	2011-08-21T08:14:24.456			
189677	1	nwells	10.17.21.71	2011-08-21T08:15:11.057	2011-08-21T08:15:11.787		42

21

How many orders did you book between 10am and 11am on November 4?

What percentage of “oe book” clicks have run in less than 1 second since August 17?

Morton’s 1-line “oe book” ran in 1 second. Nancy’s ran in 10 seconds. Why?

Is there a user or location that experiences particularly good or bad response times?

Which tasks have the most execution response time variance?

What is your best hour of “oe book” throughput since January 1? Your worst hour?

What tasks competed against “oe book” during its worst-throughput hour?

99% of “oe book” clicks must run in less than 1 second,
and 99.99% must run in less than 5 seconds. Do they?

When will your 99th percentile “oe book” response time exceed your 1-sec tolerance?

...

22

Writing the code

23

```
CREATE OR REPLACE PROCEDURE gl_posting AS
  l_sqlcode    NUMBER;
  l_sqlerrm    VARCHAR2(512);
  entries_posted NUMBER;
BEGIN
  ilo_task.begin_task(module => 'GL', action => 'Posting');
  -- Code to execute the business task goes here.
  ilo_task.end_task(widget_count => entries_posted);
EXCEPTION WHEN OTHERS THEN
  l_sqlcode := SQLCODE;
  l_sqlerrm := SQLERRM;
  ilo_task.end_task(error_num => l_sqlcode);
  -- Handle exceptions here.
  raise;
END;
```

<http://method-r.com/software/ilo>

24

```
/* Allocate a connection from the pool. */
Connection conn = this.datasource.getConnection();

/* Set task identification attributes: who is executing what? */
Socket socket = appconn.getSocket();
String ip = socket.getInetAddress().toString();
String uname = appconn.getUserName();
String uuid = UUID.randomUUID().toString();
String cid = String.format("%s %s %s", ip, uname, uuid);
this.metrics[OracleConnection.END_TO_END_CLIENTID_INDEX] = cid;
this.metrics[OracleConnection.END_TO_END_ACTION_INDEX] = appconn.getAct();
this.metrics[OracleConnection.END_TO_END_MODULE_INDEX] = appconn.getMod();
((OracleConnection)conn).setEndToEndMetrics(this.metrics, (short) 0);

/* Enable Oracle extended SQL tracing. */
if (appcon.getTracingIntention) {
    Statement stmt = conn.prepareStatement("begin method_r_trace.enable; end;");
    stmt.execute();
}

/* Code to execute the business task goes here. */

/* Disable Oracle extended SQL trace. */
if (appcon.getTracingIntention) {
    stmt = conn.prepareStatement("begin method_r_trace.disable; end;");
    stmt.execute();
}

/* Give the connection back to the pool. */
conn.close();
```

25

Oracle trace data

26

Tracing gives you *magic*.



Trace data: dbcalls, syscalls

```
*** CLIENT ID:(10.17.22.162 KFERRELL a98b2e0d-56e7-4b83-ab3c-452e121b7f7e) 2011-09-16 19:14:25.746
*** MODULE NAME:(oe) 2011-09-16 19:14:25.746
*** ACTION NAME:(pick) 2011-09-16 19:14:25.746

WAIT #0: nam='pooled connection free' ela= 0 p1=0 p2=0 p3=0
WAIT #4: nam='SQL*Net message to client' ela= 3 driver id=1952673792 #bytes=1 p3=0 obj#=-1 tim=1316218465746795
=====
PARSING IN CURSOR #4 len=33 dep=0 uid=81 oct=47 lid=81 tim=1316218465746850 hv=747926228 ad='28ff6694' sqlid='40kwnn4q98wqn'
begin method_r_trace.enable; end;
END OF STMT
EXEC #4: c=0, e=10647, p=0, cr=0, cu=0, mis=0, r=1, dep=0, og=1, plh=0, tim=1316218465746849
XCEND rlbk=0, rd_only=1, tim=1316218465736472
WAIT #4: nam='SQL*Net message from client' ela= 20814 driver id=1952673792 #bytes=1 p3=0 obj#=-1 tim=1316218465767968
=====
PARSING IN CURSOR #3 len=35 dep=0 uid=81 oct=3 lid=81 tim=1316218465768181 hv=3367355938 ad='27e070e4' sqlid='1wsd8vz4bbjj2'
select * from oe_pick where id = :1
END OF STMT
PARSE #3: c=0, e=124, p=0, cr=0, cu=0, mis=0, r=0, dep=0, og=1, plh=1414767832, tim=1316218465768180
BINDS #3:
Bind#0
  oacdy=02 mxl=22(22) mxlc=00 mal=00 scl=00 pre=00
  oacflg=03 fl2=1000000 frm=01 csi=178 siz=24 off=0
  kxsbbfbp=00a6f6d0 bln=22 avl=02 flg=05
  value=27
EXEC #3: c=0, e=186, p=0, cr=0, cu=0, mis=0, r=0, dep=0, og=1, plh=1414767832, tim=1316218465768559
WAIT #3: nam='SQL*Net message to client' ela= 3 driver id=1952673792 #bytes=1 p3=0 obj#=-1 tim=1316218465768651
WAIT #3: nam='SQL*Net message from client' ela= 1601 driver id=1952673792 #bytes=1 p3=0 obj#=-1 tim=1316218465770345
--
WAIT #3: nam='SQL*Net message from client' ela= 51475 driver id=1952673792 #bytes=1 p3=0 obj#=-1 tim=1316218472888142
=====
PARSING IN CURSOR #2 len=34 dep=0 uid=81 oct=47 lid=81 tim=1316218472888142 hv=1274106519 ad='267d2710' sqlid='c5jt7pp5z2nnr'
begin method_r_trace.disable; end;
END OF STMT
PARSE #2: c=0, e=0, p=0, cr=0, cu=0, mis=0, r=0, dep=0, og=1, plh=0, tim=1316218472888142
EXEC #2: c=0, e=12158, p=0, cr=0, cu=0, mis=0, r=1, dep=0, og=1, plh=0, tim=1316218472908809
```

Group duration by module/action...

MOD/ACT	DURATION	%	CALLS	MEAN	MIN	MAX
oe / book	232.433195	31.3%	324,109	0.000717	0.000000	0.439000
pa / mtch	159.855674	21.6%	190,551	0.000839	0.000000	0.293000
oe / ship	113.720698	15.3%	129,670	0.000877	0.000000	0.394000
pa / corr	94.278553	12.7%	130,234	0.000724	0.000000	0.301000
pa / reco	74.152792	10.0%	83,006	0.000893	0.000000	0.413000
oe / pick	67.108856	9.0%	72,625	0.000924	0.000000	0.517000
TOTAL (6)	741.549768	100.0%	930,195	0.000797	0.000000	0.517000

Group task execution duration by client_id...

CLIENT_ID	DURATION	%	CALLS	MEAN	MIN	MAX
10.17.22.12 XPHELPS eec4c72f-b685-4b5b-8447-688b8aecbc6f	49.699960	6.7%	74,351	0.000668	0.000000	0.220486
10.17.22.76 VSAUNDERS ab9fdc58-6c09-4fe2-b253-8b2c191e4671	48.977182	6.6%	108,760	0.000450	0.000000	0.154742
10.17.23.174 PDRAKE ab8e9d2c-8ba6-4c5f-8673-13a3c958af7a	47.520700	6.4%	49,059	0.000969	0.000000	0.439000
10.17.24.138 VMICHAEL ced4bddee-f0d8-44cd-a878-f0b938802bc0	43.597261	5.9%	43,983	0.000991	0.000000	0.189084
10.17.22.115 MSTANLEY 5c01b8e5-e0d3-44c2-ada8-c8b1ed17abee	23.714088	3.2%	21,701	0.001093	0.000000	0.151083
10.17.22.249 TIRWIN 2161468f-5504-419a-8226-f5ee0ad39b73	22.946626	3.1%	23,882	0.000961	0.000000	0.132008
10.17.21.167 ESANDERS e463a90b-32a8-49fc-81e8-0c2559269e1e	22.607869	3.0%	24,175	0.000935	0.000000	0.128223
10.17.22.98 DHALEY 8a41c465-e00f-43b2-afcd-ba4edd7df899	21.967559	3.0%	22,522	0.000975	0.000000	0.159246
10.17.21.239 UDECKER 052b3d18-e992-46a2-8ae3-87427c78598e	20.312153	2.7%	19,922	0.001020	0.000000	0.137756
10.17.21.14 LDILLARD 2270f4b4-1c46-4ad7-ac6a-2ed1fc13c283	20.024742	2.7%	19,818	0.001010	0.000000	0.138701
158 others	420.181628	56.7%	522,022	0.000805	0.000000	0.517000
TOTAL (168)	741.549768	100.0%	930,195	0.000797	0.000000	0.517000

Group task execution duration by call-name for a specific task execution (profile)...

CALL-NAME	DURATION	%	CALLS	MEAN	MIN	MAX
SQL*Net message from client	40.507440	81.5%	18,585	0.002180	0.000000	0.220486
EXEC	6.684420	13.4%	18,585	0.000360	0.000000	0.128008
FETCH	2.388145	4.8%	18,583	0.000129	0.000000	0.120008
SQL*Net message to client	0.094487	0.2%	18,585	0.000005	0.000000	0.018525
cursor: pin S wait on X	0.025463	0.1%	4	0.006366	0.000000	0.011166
cursor: pin S	0.000005	0.0%	5	0.000001	0.000000	0.000005
PARSE	0.000000	0.0%	2	0.000000	0.000000	0.000000
pooled connection free	0.000000	0.0%	1	0.000000	0.000000	0.000000
XCTEND	0.000000	0.0%	1	0.000000	0.000000	0.000000
TOTAL (9)	49.699960	100.0%	74,351	0.000668	0.000000	0.220486

Group call duration by duration for a specific call type for a specific task execution...

RANGE {min ≤ e < max}	DURATION	%	CALLS	MEAN	MIN	MAX
0.000000 0.000001	0.000000	0.0%	3	0.000000	0.000000	0.000000
0.000001 0.000010	0.000000					
0.000010 0.000100	0.000358	0.0%	5	0.000072	0.000041	0.000092
0.000100 0.001000	8.391000	20.7%	13,947	0.000602	0.000104	0.000999
0.001000 0.010000	9.307749	23.0%	3,822	0.002435	0.001000	0.009955
0.010000 0.100000	20.877546	51.5%	793	0.026327	0.010001	0.099419
0.100000 1.000000	1.930787	4.8%	15	0.128719	0.103157	0.220486
1.000000 10.000000	0.000000					
10.000000 100.000000	0.000000					
100.000000 1000.000000	0.000000					
1000.000000 Infinity	0.000000					
TOTAL (6)	40.507440	100.0%	18,585	0.002180	0.000000	0.220486

Group call duration by call execution id for a specific call type for a specific execution...

LINE#	DURATION	%	CALLS	MEAN	MIN	MAX
146783	0.220486	0.5%	1	0.220486	0.220486	0.220486
82770	0.181826	0.4%	1	0.181826	0.181826	0.181826
121473	0.164216	0.4%	1	0.164216	0.164216	0.164216
102868	0.128376	0.3%	1	0.128376	0.128376	0.128376
120701	0.126663	0.3%	1	0.126663	0.126663	0.126663
246050	0.120756	0.3%	1	0.120756	0.120756	0.120756
74221	0.119724	0.3%	1	0.119724	0.119724	0.119724
120861	0.119231	0.3%	1	0.119231	0.119231	0.119231
181952	0.109707	0.3%	1	0.109707	0.109707	0.109707
192950	0.109635	0.3%	1	0.109635	0.109635	0.109635
18,575 others	39.106820	96.5%	18,575	0.002105	0.000000	0.108649
TOTAL (18,585)	40.507440	100.0%	18,585	0.002180	0.000000	0.220486

Group call duration by call execution id and sql id for a specific call type for a specific execution...

LINE#:	SQLID	DURATION	%	CALLS	MEAN	MIN	MAX
146783:	6m3a3v3yjavbh	0.220486	0.5%	1	0.220486	0.220486	0.220486
82770:	6m3a3v3yjavbh	0.181826	0.4%	1	0.181826	0.181826	0.181826
121473:	6m3a3v3yjavbh	0.164216	0.4%	1	0.164216	0.164216	0.164216
102868:	6m3a3v3yjavbh	0.128376	0.3%	1	0.128376	0.128376	0.128376
120701:	6m3a3v3yjavbh	0.126663	0.3%	1	0.126663	0.126663	0.126663
246050:	6m3a3v3yjavbh	0.120756	0.3%	1	0.120756	0.120756	0.120756
74221:	6m3a3v3yjavbh	0.119724	0.3%	1	0.119724	0.119724	0.119724
120861:	6m3a3v3yjavbh	0.119231	0.3%	1	0.119231	0.119231	0.119231
181952:	6m3a3v3yjavbh	0.109707	0.3%	1	0.109707	0.109707	0.109707
192950:	6m3a3v3yjavbh	0.109635	0.3%	1	0.109635	0.109635	0.109635
18,575 others		39.106820	96.5%	18,575	0.002105	0.000000	0.108649
TOTAL (18,585)		40.507440	100.0%	18,585	0.002180	0.000000	0.220486

Performance feature wish list

37

Performance features wish list

Make it easy for developers to get trace data

Trace $p\%$ of executions of a given task

Create profiles from trace files

Aggregate trace files by arbitrary dimension

Show profile differences side-by-side

Predict when p th percentile R exceeds threshold

Report tasks with excessive R variance-to-mean ratio

Let me know, at info@method-r.com.

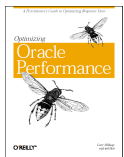
38

References

39



Ron Crisco, Robyn Sands, et al. 2011. [Expert PL/SQL Practices](#). Apress
Detailed information about instrumenting your Oracle application code.



Cary Millsap, Jeff Holt. 2003. [Optimizing Oracle Performance](#). O'Reilly
Detailed information about Oracle trace data and what to do with it.



Cary Millsap. 2011. [Mastering Oracle Trace Data](#). Method R Corporation
One-day course including software that teaches you how to master Oracle trace data.

40

Discussion



Cary Millsap



*Me*THOD R

<http://method-r.com>

<http://carymillsap.blogspot.com>

[@CaryMillsap](#) [@MethodR](#)

 *Me*THOD R™

