_experience the commitment

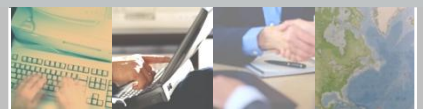
Method R

Performance Optimization the Smart Way

Chad McMahon

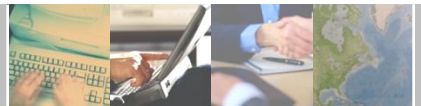
Senior Consultant, Database Services

CGI



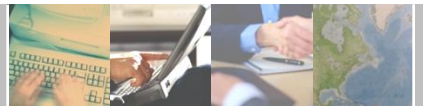
About the Speaker

- **Chad McMahon**
  - **Career:**
    - Database consultant at CGI Database Services for five years.
      - Provide Oracle & SQL Server database support for multiple clients.
      - Various industries – oil & gas, health care, government.
      - Instruct Performance & Tuning for the DBA Fast-Track Program at SAIT Polytechnic.
    - Prior to CGI, a DBA at EPCOR Utilities for three years.
  - **Relevant Area of Expertise:** *Oracle Performance Optimization*
    - Have worked on a multitude of Oracle performance optimization initiatives, ranging from assisting developers with simple query tuning, to resolving large scale mission-critical Oracle performance projects.



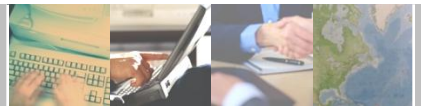
Agenda:

- How do you diagnose performance problems in your day-to-day life?
- Introduction to Method R
- Method R Steps
- Real-life example of bottleneck analysis using Method R
- Method R in action… A Live Demonstration



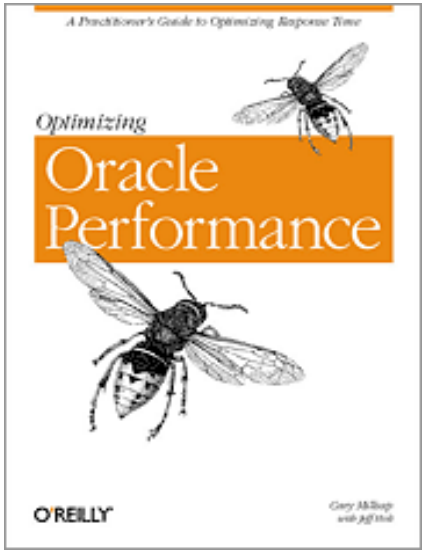
How do you diagnose performance problems in your day-to-day life?

- **You're late for work… How do you prevent it from happening again?**
  - Do you…
    - Remove all your car's surface imperfections?
    - Ensure proper wheel alignment?
    - Ensure the engine is producing ≥ 99.9…% of its rated horsepower?
    - Replace your Ford Taurus with a Ferrari Enzo?
- **Yet it's how most people respond to slow database applications.**
  - Do you…
    - Try to eliminate ALL full table scans?
    - Rebuild indexes on a regular basis, or when performance dips?
    - Ensure ≥ 99.9…% buffer cache hit ratio?
    - Replace your old database server with a new new server?



Introduction to Method R

What is Method R ??



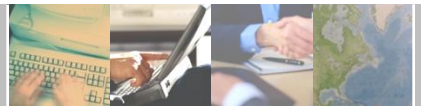
- **Method R is "a response time-based performance improvement method that yields maximum economic value to your business."**

  Source: [Millsap, C.; Holt, J. (2003)]

Method R Steps

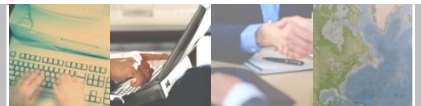
1. **Target the tasks that are critical to the business**
2. **Collect properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.**
3. **React with the candidate repair that will have the greatest net payoff to the business.**
   a. Stop if the cost of the repair exceeds the cost of the problem.
4. **Go to step 1.**

Source: [Millsap, C.; Holt, J. (2003)]



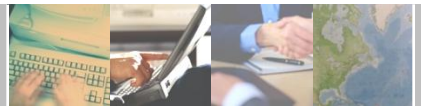
Method R Steps in Detail

1. **Target the tasks that are critical to the business**
2. Collect properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.
3. React with the candidate repair that will have the greatest net payoff to the business.
    a. Stop if the cost of the repair exceeds the cost of the problem.
4. Go to step 1.



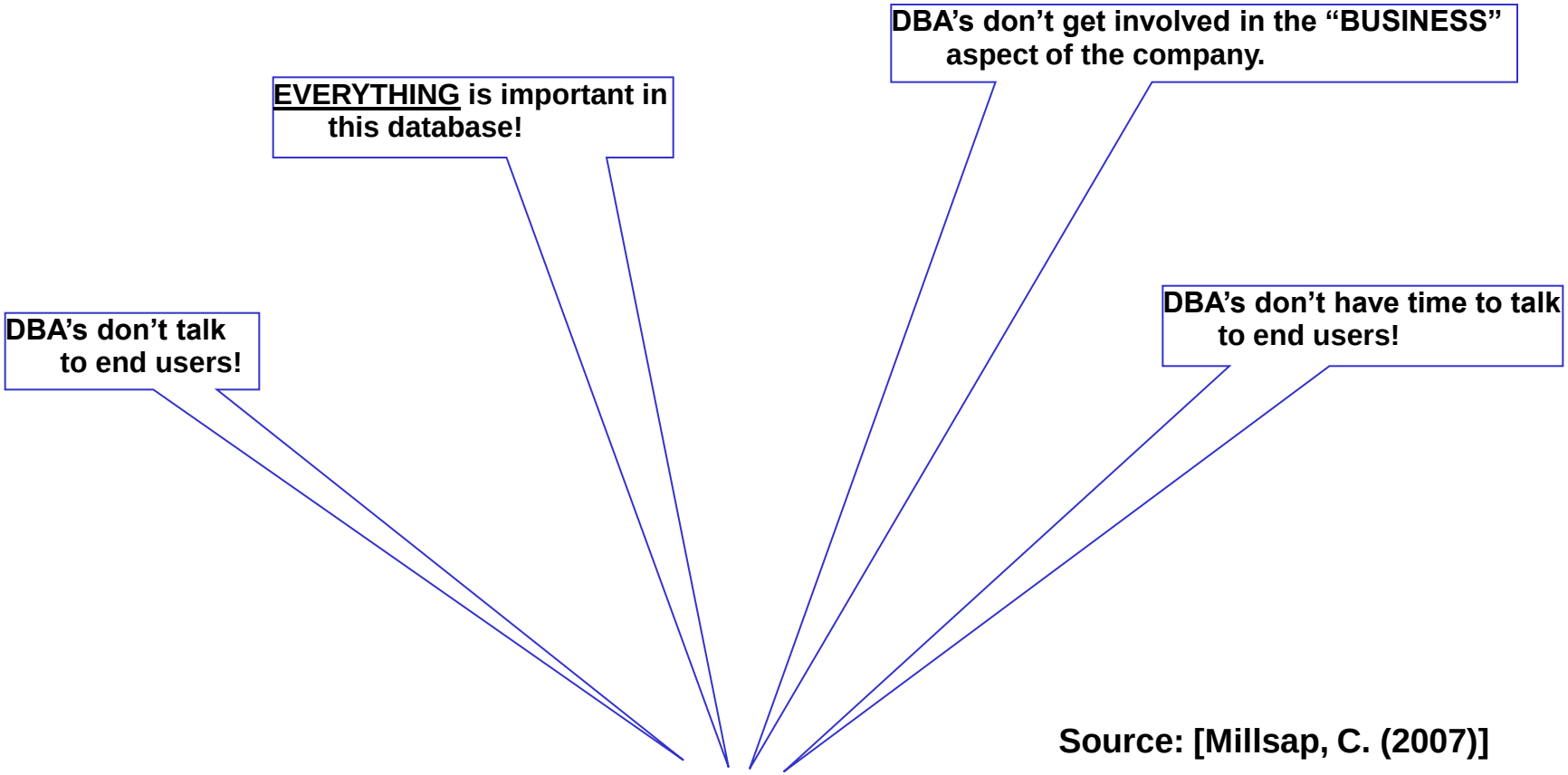
Targeting tasks that are critical to the business

- Talk to the end users to find out which business process is the bottleneck.
- DBA's don't decide which tasks are most critical to the business, the business does!
- DBA's don't like to take this step of Method R… Why?



Targeting tasks that are critical to the business

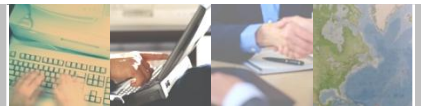
Why DBA's don't like to take the first step of Method R…



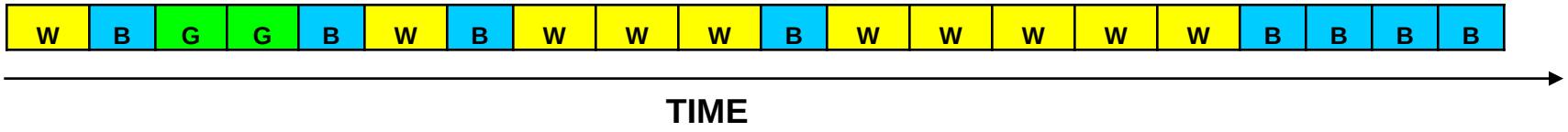
Source: [Millsap, C. (2007)]

Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

1. Target the tasks that are critical to the business
2. **Collect properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.**
3. React with the candidate repair that will have the greatest net payoff to the business.
   a. Stop if the cost of the repair exceeds the cost of the problem.
4. Go to step 1.



Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

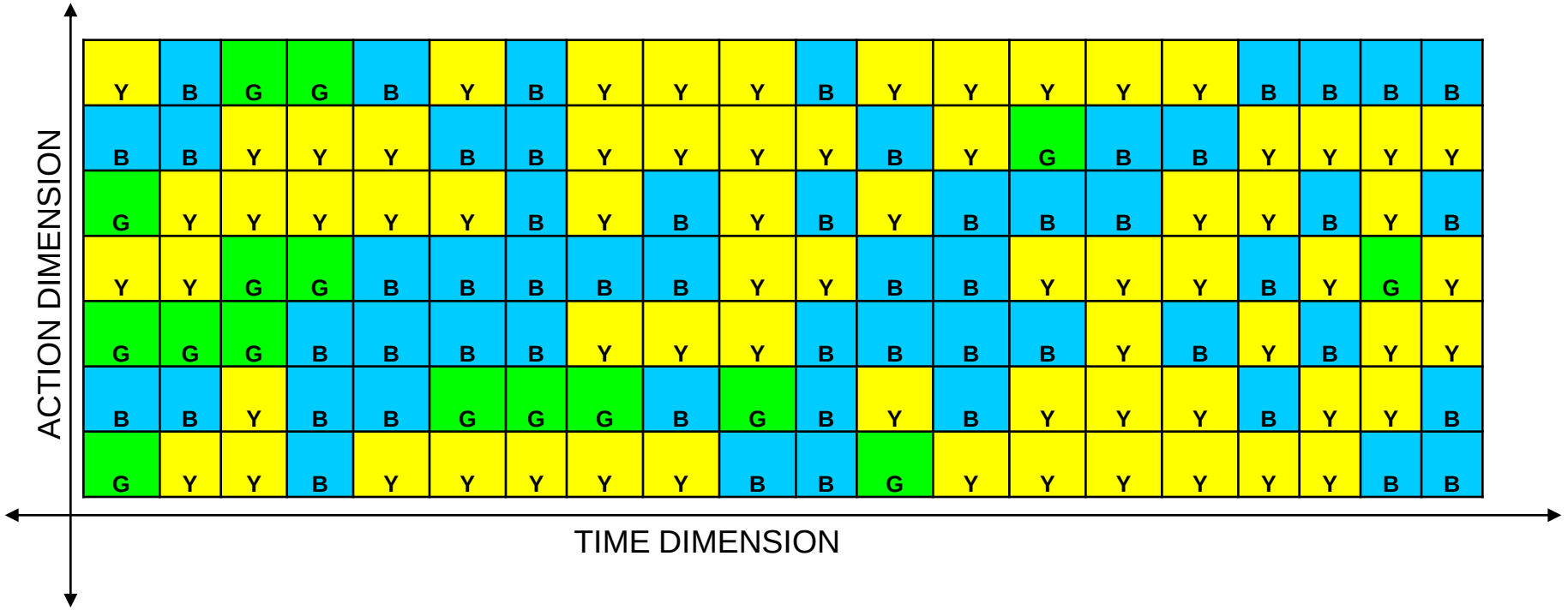


- Imagine the drawing above describes a user action's response time consumption.
- Our imaginary system consists of three types of resources, called W, B, and G.
- The time dimension extends in the horizontal direction from LEFT to RIGHT.

Source: [Millsap, C.; Holt, J. (2003)]

Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

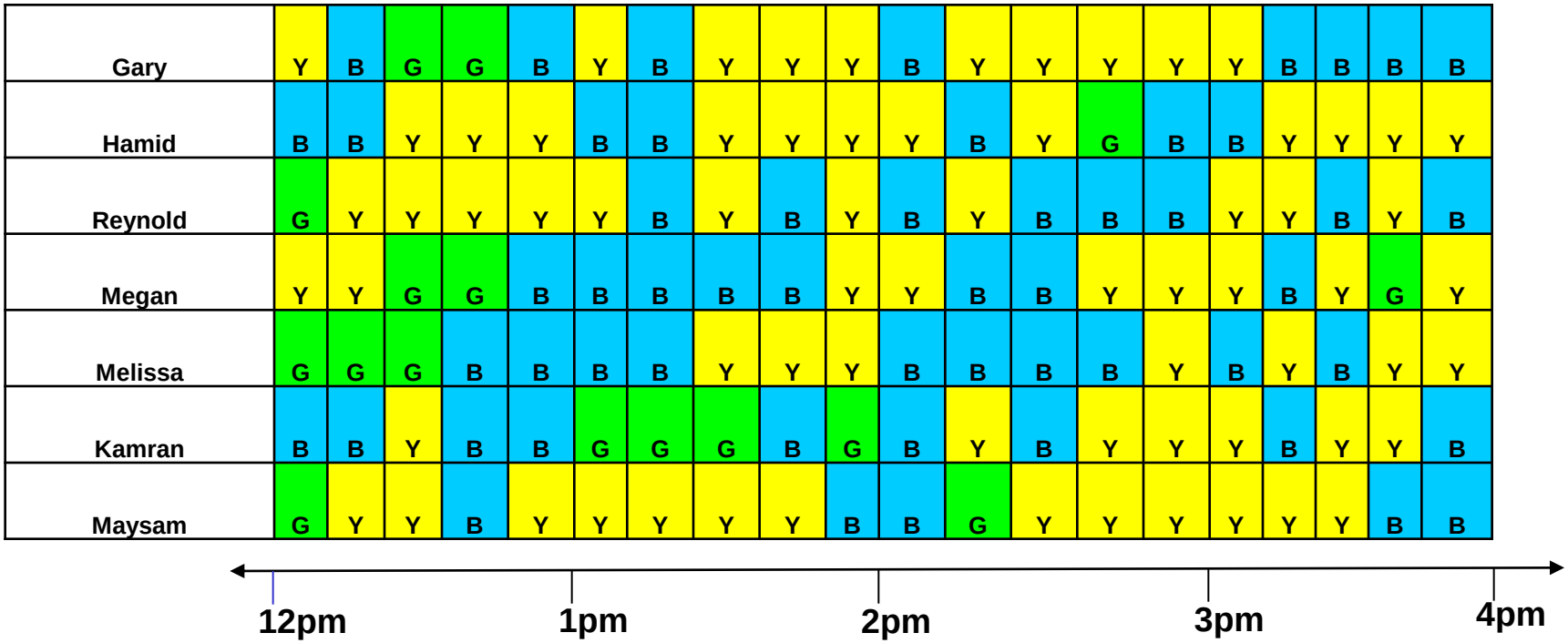
- We can denote a system that is executing several user actions at the same time by stacking such drawings vertically



Source: [Millsap, C.; Holt, J. (2003)]

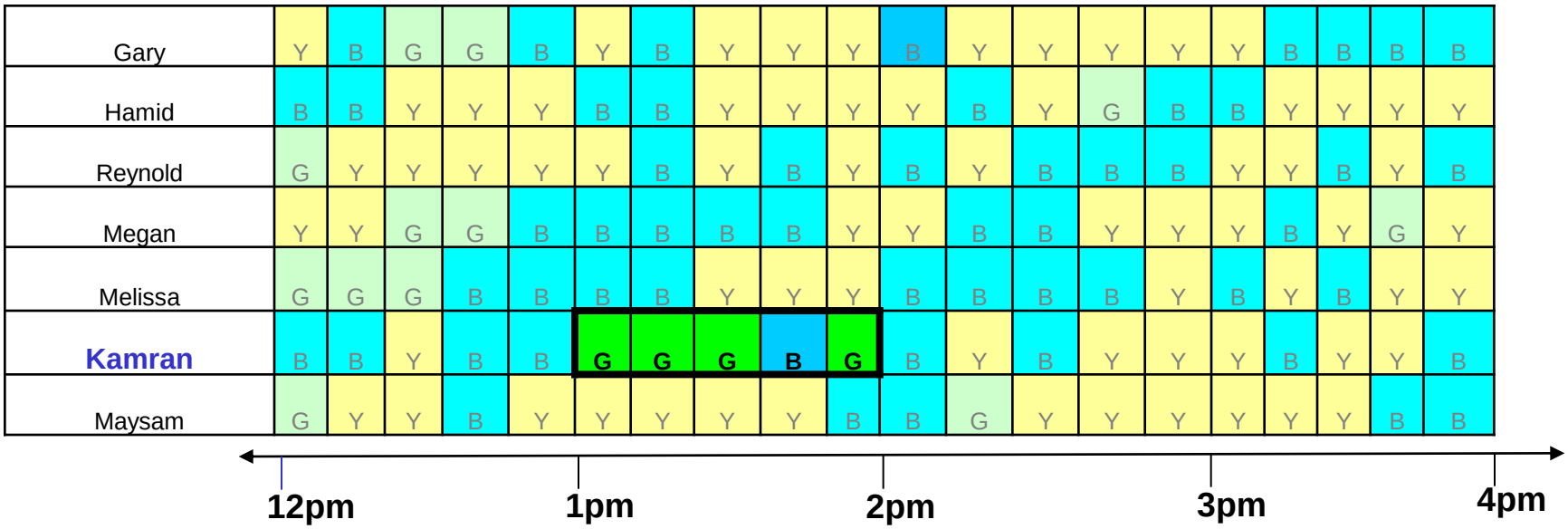
Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

Kamran is experiencing poor performance between 1pm and 2pm



Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

Gather properly scoped, un-aggregated profile data for Kamran's process between 1pm & 2pm

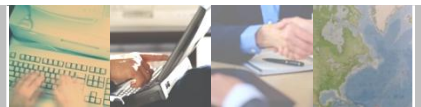


Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

Resultant Resource Profile for Kamran's process between 1pm & 2pm

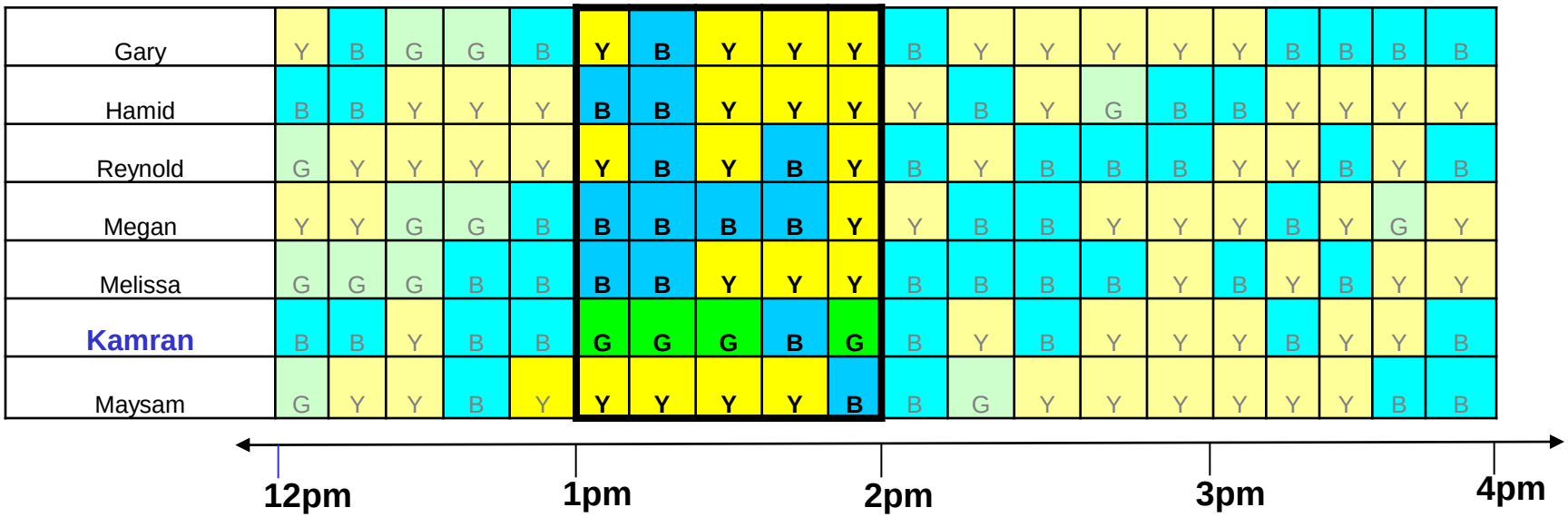
Resource	Elapsed time	Percentage of total time
G	***4***	**80.0%**
B	1	20.0%
Total	***5***	***100.0%***

- "G "consumes 80% of Kamran's response time.
- Reducing "G" will have the greatest performance benefit for Kamran.



Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

Example of how aggregated system-wide data can skew the result



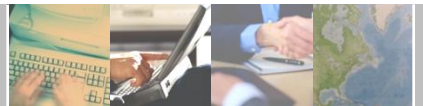
- **Examples:** Timed-snapshot tools such as Statspack / AWR / ADDM
- Will find the entire **SYSTEM's** bottleneck.
- Great for investigating overall system performance.
- Not great for investigating individual business task performance.

Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

Resultant Resource Profile for the entire system between 1pm & 2pm

Resource	Elapsed time	Percentage of total time
Y	**18**	**51.4%**
B	13	37.1%
G	**4**	**11.5%**
Total	***35***	***100.0%***

- The SYSTEM's bottleneck is Y with 51.4% of the response time.
- Kamran doesn't spend any response time on Y between 1pm & 2pm!
- Improving Y will have 0% performance improvement for Kamran!
- Remember Kamran's bottleneck is G, consuming 80% of his response time.



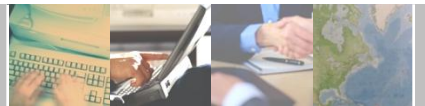
Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

Example of how incorrect time scope can cause grief…

Gary	Y	B	G	G	B	Y	B	Y	Y	Y	B	Y	Y	Y	Y	Y	B	B	B	B
Hamid	B	B	Y	Y	Y	B	B	Y	Y	Y	Y	B	Y	G	B	B	Y	Y	Y	Y
Reynold	G	Y	Y	Y	Y	Y	B	Y	B	Y	B	Y	B	B	B	Y	Y	B	Y	B
Megan	Y	Y	G	G	B	B	B	B	B	Y	Y	B	B	Y	Y	Y	B	Y	G	Y
Melissa	G	G	G	B	B	B	B	Y	Y	Y	B	B	B	B	Y	B	Y	B	Y	Y
Kamran	**B**	**B**	**Y**	**B**	**B**	**G**	**G**	**G**	**B**	**G**	**B**	**Y**	**B**	**Y**	**Y**	**Y**	**B**	**Y**	**Y**	**B**
Maysam	G	Y	Y	B	Y	Y	Y	Y	Y	B	B	G	Y	Y	Y	Y	Y	Y	B	B

12pm 1pm 2pm 3pm 4pm

- **Remember, Kamran is having performance problems between 1pm & 2pm**
- **What happens if our time scope is incorrect, (12pm – 4pm)?...**

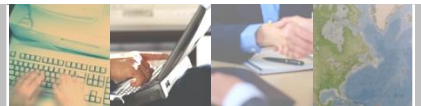


Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

Resultant Resource Profile for Kamran's action between 12pm & 4pm

Resource	Elapsed time	Percentage of total time
B	**9**	**45.0%**
Y	7	35.0%
G	**4**	**20.0%**
Total	***20***	***100.0%***

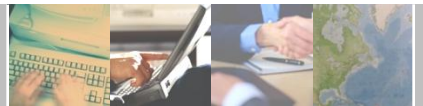
- Apparent "evidence" that B and Y are the culprits, yet we already know that:
  - Reducing B will only have 20% performance improvement for Kamran.
  - Reducing Y will have 0% performance improvement for Kamran.
- G is Kamran's real problem, consuming 80% of his response time between 1pm & 2pm.



Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

Ok that's great, BUT …

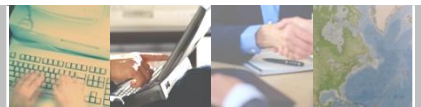
How do we ACTUALLY gather this properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior we want to record ???



Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

Answer …

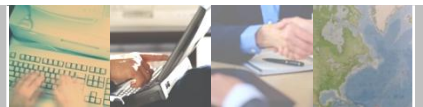
Oracle Extended SQL Trace



Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

The Oracle Extended SQL Trace gives us everything we need to:

- Create the resource profile which gives the breakdown of timed events in order of greatest response time duration to least.
- View exactly which Oracle statements are being executed.
- Wait events, and wait event response times for each statement being executed.
- SQL execution plans for each statement executed.
- Logical I/O (buffer cache blocks read)
- Physical I/O (OS reads/writes to Oracle data/temp/redo/archive/control files)
- Number of parse calls, executes, fetches.
- Bind variables values.
- Etc.



Overview of how to use the Oracle extended SQL trace:

1. **Prepare your session for the extended SQL trace:**

```
alter session set max_dump_file_size=unlimited;
alter session set timed_statistics=true;
alter session set statistics_level = all;
alter session set tracefile_identifier = <identifier>;
```

2. **Enable the Oracle Extended SQL Trace in your session:**

```
execute DBMS_MONITOR.SESSION_TRACE_ENABLE(waits=>true, binds=>true);
```

3. **Execute the business task in question:**

```
execute <stored_procedure>;
SELECT <blah> FROM <blah> WHERE <blah>;
INSERT ...;
UPDATE ...;
```

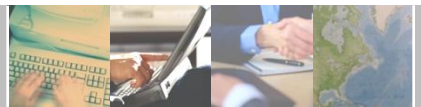
The business task can be anything from database calls from the application, execution of stored procs, dbms_jobs, etc., etc.

3. **Logout of the database and/or stop the Oracle Extended SQL Trace:**

- Logout of the session.

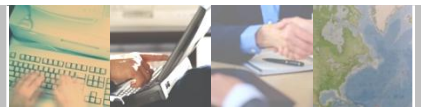
  OR

- `execute DBMS_MONITOR.SESSION_TRACE_DISABLE(session_id=>NULL, serial_num=>NULL);`



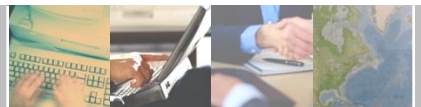
Oracle Extended SQL Trace

- **Extended SQL Trace files are placed in:**
  - In the `USER_DUMP_DEST` directory for Oracle 10g and earlier.
  - In the Automatic Diagnostic Repository (ADR) in Oracle 11g:
    - *<ADR-home>/trace*
- **There are many ways to start/stop the Oracle Extended SQL Trace depending on your scenario and/or version of Oracle:**
  - ALTER SESSION SET EVENTS '10046 trace name context forever, level 12'
  - DBMS_SYSTEM.set_ev
  - DBMS_SUPPORT.start_trace_in_session
  - DBMS_MONITOR.session_trace_enable
  - DBMS_MONITOR.client_id_trace_enable
  - DBMS_MONITOR.serv_mod_act_trace_enable
  - etc.



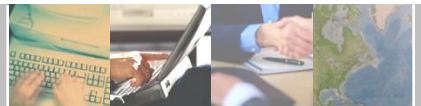
Collecting properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.

- **Raw trace files can be formatted into a more readable format using tools such as:**
  - **tkprof**
    - Included in Oracle installation, and located in `$ORACLE_HOME/bin`
  - **Oracle Trace Analyzer**
    - Downloadable from Oracle Support Note: 224270.1 (**http://support.oracle.com**)
  - **Hotsos Profiler**
    - Developed by the team who pioneered Method R (**http://www.hotsos.com/profiler.html**)
  - **OraSRP**
    - Free downloadable tool developed very “eerily” similar to the Hotsos Profiler (**http://www.oracledba.ru/orasrp**)
  - **etc.**



React with the candidate repair that will have the greatest net payoff to the business.

1. Target the tasks that are critical to the business
2. Collect properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.
3. **React with the candidate repair that will have the greatest net payoff to the business.**
    a. Stop if the cost of the repair exceeds the cost of the problem.
4. Go to step 1.

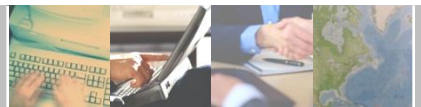


React with the candidate repair that will have the greatest net payoff to the business.

- This is where some DBA's start their tuning projects.
- How do you know the fix if you don't know the problem in the first place!
- You might solve your performance problems by jumping directly to this step, but it's usually luck!
- You will only achieve this step successfully and consistently by following the first two steps of Method R.

REMEMBER: Stop if the cost of the repair exceeds the cost of the problem.

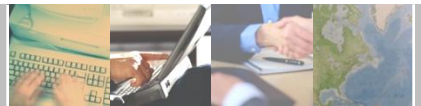
Don't come down with a case of Compulsive Tuning Disorder (CTD)



Real-life example of bottleneck analysis using Method R

1. **Target the tasks that are critical to the business.**
   - ie. What is the business's bottleneck?

     - A process that audits medical record updates is performing very poorly. As a result, audit logging queues are consistently growing over time.

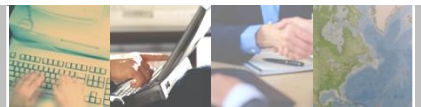


Real-life example of bottleneck analysis using Method R

2. **Collect properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.**
   - ie. Capture and analyze diagnostic data of ONLY the audit logging process, while it is exhibiting poor performance.

     Analysis of an Oracle extended SQL trace file of the audit logging process revealed that:

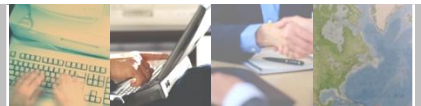
     - **74% of the task's response time is consumed by the "log file sync" Oracle wait event.**



Real-life example of bottleneck analysis using Method R

Resource Profile generated by OraSRP of 2,077 Records Audited

Event Name	% Time	Seconds
log file sync	74.0%	125.4837s
SQL*Net message from client [idle]	7.2%	12.2265s
EXEC calls [CPU]	7.1%	12.0800s
log file switch (checkpoint incomplete)	6.8%	11.5903s
SQL*Net message from client	2.7%	4.6452s
FETCH calls [CPU]	1.3%	2.2800s
log file switch completion	0.7%	1.1369s
PARSE calls [CPU]	0.1%	0.1600s
SQL*Net message to client	0.0%	0.0259s
SQL*Net more data from client	0.0%	0.0182s
db file sequential read	0.0%	0.0003s
Total	**100.0%**	**169.647s**



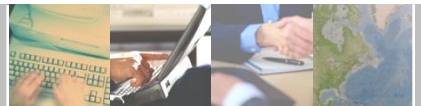
Real-life example of bottleneck analysis using Method R

3. **React with the candidate repair that will have the greatest net payoff to the business.**

   - ie. How can we make the audit logging process consume less time waiting for “log file sync”?

   - **“log file sync” is incurred when a session commits/rollbacks a transaction.**

   - **“log file sync” is accumulated while a session is waiting for committed/rollbacked data to be written from the log_buffer to the online redo logs by the LGWR background process.**

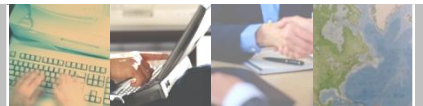


Real-life example of bottleneck analysis using Method R

- **Fix:**
  - **Batch several audit logging updates together in a single transaction, therefore reducing the commit frequency.**
  - **Move online redo logs to faster disks to increase the LGWR I/O throughput.**

Resource Profile generated by OraSRP of 2,083 records Audited

Event Name	% Time	Seconds
EXEC calls [CPU]	64.6%	12.0800s
SQL*Net message from client	16.9%	3.1586s
FETCH calls [CPU]	12.2%	2.2800s
log file sync	5.4%	1.0040s
PARSE calls [CPU]	0.9%	0.1600s
SQL*Net more data from client	0.1%	0.0156s
SQL*Net message to client	0.1%	0.0149s
db file sequential read	0.0%	0.0006s
Total	**100.0%**	**18.7137s**



Real-life example of bottleneck analysis using Method R

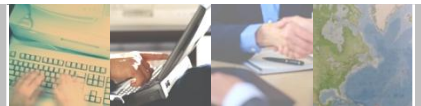
Comparison of Resource Profiles

Before

Event Name	% Time	Seconds
log file sync	74.0%	125.4837s
SQL*Net message from client [idle]	7.2%	12.2265s
EXEC calls [CPU]	7.1%	12.0800s
log file switch (checkpoint incomplete)	6.8%	11.5903s
SQL*Net message from client	2.7%	4.6452s
FETCH calls [CPU]	1.3%	2.2800s
log file switch completion	0.7%	1.1369s
PARSE calls [CPU]	0.1%	0.1600s
SQL*Net message to client	0.0%	0.0259s
SQL*Net more data from client	0.0%	0.0182s
db file sequential read	0.0%	0.0003s
Total	**100.0%**	**169.647s**

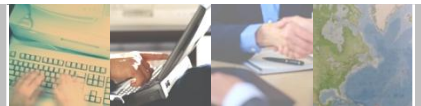
After

Event Name	% Time	Seconds
EXEC calls [CPU]	64.6%	12.0800s
SQL*Net message from client	16.9%	3.1586s
FETCH calls [CPU]	12.2%	2.2800s
log file sync	5.4%	1.0040s
PARSE calls [CPU]	0.9%	0.1600s
SQL*Net more data from client	0.1%	0.0156s
SQL*Net message to client	0.1%	0.0149s
db file sequential read	0.0%	0.0006s
Total	**100.0%**	**18.7137s**



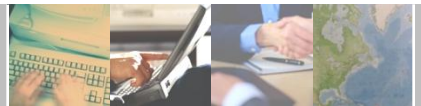
Live Demonstration

- **Company Background**
  - Company Name: **Chadazon**
  - Application Type: **Order Entry**
  - Database: **Oracle 11gR2 (11.2.0.1) on Oracle Enterprise Linux 5**
  - Application users vary from customers purchasing merchandise, to distribution center staff processing orders, to company management running reports.



Live Demonstration

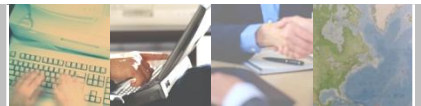
- **Explanation of Performance Problem**
  - The DBA has received a high priority helpdesk request from the distribution center staff
  - They are indicating that the entire system is slow and getting worse over time.
  - As a result, they can’t produce time-sensitive reports within the required time.



Live Demonstration

Step 1: Target the tasks that are critical to the business:

- An interview with key distribution center staff reveals:
  - The poorly performing business task is a report showing a count of un-processed orders.
  - In order to maintain service level agreements, this report must take no longer than 10 seconds.
  - It is currently taking anywhere from 30-300 seconds.

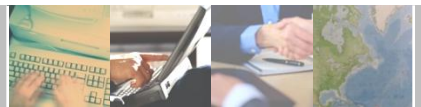


Live Demonstration

Step 1 Continued:

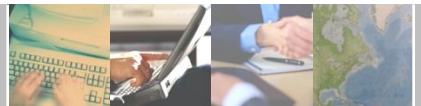
- A brief discussion with the application support team reveals that the SQL for the report is:

```
SELECT COUNT(*) "UN-PROCESSED_ORDERS"
  FROM ORDERS
 WHERE ORDER_STATUS = 3;
```



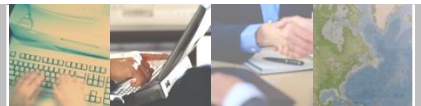
Live Demonstration

- **<u>Step 2:</u> Collect properly scoped, un-aggregated profile data for each task while the task is exhibiting the behavior you want to record.**
  - **Process:**
    - *From SQL*Plus:*
      1. Enable the Extended SQL Trace
      2. Execute the query.
      3. Log out of the session to end the Extended SQL Trace.



Live Demonstration

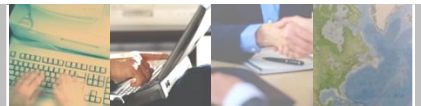
- **<u>Step 3:</u> React with the candidate repair that will have the greatest net payoff to the business.**
  - I will demonstrate on my laptop!



Live Demonstration

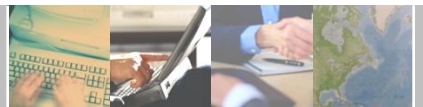
How will I emulate this environment on my PC???

- **Use OraSRP Extended SQL Trace Profiler to analyze and create resource profiles from the Oracle extended SQL traces files.**
  - http://oracledba.ru/orasrp/
  - **Syntax:**
    - `> orasrp [trace_file_name] [output_file_name].html`



Outside the scope of this presentation

- **What if you don't know the SQL and/or PL/SQL statement(s) being executed??**
- **What if an external application is executing the problem code??**
- **Starting/stopping a trace in someone else's session ??**
- **Database connection pooling??**
- **All application users log into the database as a common username.**
  - How can I trace a particular session??
  - How do I know which one to trace ??



References

- **Millsap, C.; Holt, J. (2003)** *Optimizing Oracle Performance***. Sebastopol CA: O'Reilly & Associates. ISBN 0-596-00527-X**

- **Millsap, C. (2007)** "Preventing Performance Problems" **from the 2007 Hotsos Symposium Training Day:** *https://portal.hotsos.com/events/SYM07*

- **Millsap, C. (2006)** *"Questioning Method R"* **Hotsos Article Library:** *http://hotsos.com/library.html*