Resource Mapping
A Wait Time Based Methodology for Database Performance Analysis

Prepared for NYOUG, 2005

Presented by
Matt Larson
Chief Technology Officer
Confio Software
Presentation Agenda

- Introduction
- Conventional Tuning vs. Wait-based Tuning
- Foundation: Resource Mapping Methodology
- 5 Key Steps of Applying RMM
Problems with Conventional Tuning Tools:
Like the Drunk Under the Streetlight

You can look where it’s convenient, or look where you will actually find a solution – you choose!
Resource Mapping Methodology

- **RMM is...**
- A rigorous practice for database tuning using *Wait-Event analysis*
- A set of *requirements* defining what you need to know about a database in order to solve the real performance problems
- A tuning approach that focuses on actions yielding *most important business impact*
- A recipe to **Be a Better DBA**
Conventional Tuning

- Art, not a science
- Ratio-based (cache hit ratios, etc.)
- Sometimes fruitless
- It’s “tuned” (I guess?)
- Different tuning/investigation process for each DBA/DBA Team/Company
Problems with Conventional Tuning Tools

- Optimize systems, not business results
- Conventional tools:
  - V$ Views: limited visibility & granularity
  - Statspack: averages across entire database
- Incorrect Data hides real results
  - System-wide averages
  - Event counters
  - Incomplete visibility
What Problems are you Trying to Solve?

• I spend the whole week monitoring and optimizing Oracle configurations, but I have no demonstrable results to show for it - why?
• Will more hardware make my application run faster? By how much?
• Will the new application run efficiently on the production server?
• Why does one application keep impacting my SLA compliance?
• If I could make one (or 2, 3, or 4) changes to my database to have the biggest impact, what would they be?
After spending an agonizing week tuning Oracle buffers to minimize I/O operations, management typically rewards you with:

- A. An all expense paid vacation
- B. A free lunch
- C. A stale donut
- D. Reward? Nobody even noticed!
Visibility problem?

You measure database performance based on:

- A. Increasing trends in user response time
- B. Increasing system down time
- C. Increasing help desk calls
- D. Increasing decibel levels from irate users
Tuning Success (or lack thereof)

Your role in the rollout of a new customer facing application results in:

- A. Keys to drive the CEO’s Porsche
- B. Keys to use the executive restroom
- C. A mop to use in the executive restroom
- D. Your office has been moved to the restroom
Measuring performance

You measure the commute time to work based on:

- A. The time it takes to get there
- B. Counting the times your wheels rotate
- C. Monitoring your tachometer
- D. The number of speeding tickets
Emerging best-practice for database tuning

- “You can’t tell how long something took by counting how many times it happened. … If you’re only measuring event counts, then you’re not measuring what the users care about.”
  — Cary Millsap, Optimizing Oracle Performance

Oracle is starting to build wait-based tuning tools into the database particularly in 10g

Tune by determining where processing time is spent
Oracle 10g - Moving towards wait-based

- Adding wait-based columns to existing views
- New wait-based views in ASH

Example:

$v/session\_wait\_history$

- Provides the last 10 wait events for a session
- Session ID, Username, Event, **Wait_Time**, etc.
- Used to provide wait_time for only a few events
Not all Wait-Event Statistics are Useful

- **Wait Event Analysis is too general**

Example: Sample database-wide statistics (possibly from `v$sysstat,v$latch`)

- db block gets: 53023 seconds
- physical reads: 37734 seconds
- shared pool latch: 694413 seconds
- cache buffers chains latch: 3613269 seconds
Resource Mapping Methodology defines practical requirements to perform Wait-Event Analysis:

- General approach - best practice
- Rigorous, complete requirements
Confio’s Resource Mapping Methodology

- Assembly Line: Data In -> Process -> Results Out
- Observe Assembly Line (SQL Statement) at “Unit of Work” Level (*SQL View Principle*)
- Measure time to complete, not number or occurrences (*Time View Principle*)
- Monitor every resource or suffer blind spots (*Full View Principle*)

Counters

- **CPU**: 74%
- **Reads**: 1789327
- **Time**: 145 seconds

Blind Spot

Follow a unit of work through every operation

Counters

- **CPU**: 38%
- **Reads**: 4955
- **Time**: 8726 seconds
Track SQL Time, Not System Counters

- Watching Counters leads to wrong conclusions: Time is more relevant
- Total System Counters hide information: Need breakdown to individual SQLs

<table>
<thead>
<tr>
<th>Resources</th>
<th>SQL 1</th>
<th>SQL 2</th>
<th>SQL 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30 Minutes</td>
<td>15M</td>
<td>5M</td>
</tr>
<tr>
<td></td>
<td>4 M</td>
<td>200 Minutes</td>
<td>50 Reads</td>
</tr>
<tr>
<td>I/O</td>
<td>80K Reads</td>
<td>200 Minutes</td>
<td>50 Reads</td>
</tr>
<tr>
<td></td>
<td>5K Packets</td>
<td>10 M</td>
<td>100 Minutes</td>
</tr>
<tr>
<td>Network</td>
<td>125 Attempts</td>
<td>35 A</td>
<td>50 A</td>
</tr>
<tr>
<td>Locks</td>
<td>216K Writes</td>
<td>200 Minutes</td>
<td>5M</td>
</tr>
<tr>
<td>Redo</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Two Primary Types of Tools

- Session Specific Tools
  - Tools that focus on one session at a time often by tracing the process
  - Examples: Hotsos Profiler, tkprof

- Continuous DB Wide Monitoring Tools
  - Tools that focus on all sessions by sampling Oracle
  - Examples: Confio DBFlash, Veritas Indepth

- Both tools have a place in the organization
Tracing

- Tracing with wait events complies with RMM
- Should be used cautiously in non-batch environments due to session statistics skew
  - 80 out of 100 sessions have no locking contention issues
  - 20 out of 100 have spent 99% of time waiting for locked rows
  - If you trace one of the “80” sessions, it appears as if you have no locking issues (and spend time trying to tune other items that may not be important)
  - If you trace one of the “20” sessions, it appears as if you could fix the locking problems and reduce your wait time by 95+%
Tracing (cont)

- Very precise statistics, may be only way to get certain statistics
- Bind variable information is available
- Different types of tracing available providing detail analysis even deeper than wait events
- Ideal if a known problem is going to occur in the future
- Difficult to see trends over time
- Primary audience is technical user
Continuous DB Wide Monitoring Tools

- Allows DBA to go back in time and retrieve information if problem was not expected
- Not the level of detail provided by tracing
- Most of these tools have trend reports that allow communication with others outside of the group
  - What is starting to perform poorly?
  - What progress have we made while tuning?
Example 1: Problem Observed

- Critical situation: Secure Service Center application performance unsatisfactory
  - Response time between 2400 and 9000 seconds
  - Very high network traffic (3x—4x normal), indicating time-outs and user refreshes
  - “CritSit” declared: major effort to resolve problem
Observations using Resource Mapping Methods

- 1: Identify accumulated Waits
- 2: Identify specific resources used

Notice scale: > 8000 secs
Results

Library cache pin nearly unobservable

Library cache load lock no longer observable

Notice scale: < 1400 secs max vs. 8000 previously
Results

- Response time improvement from 8000 seconds (worst case) to 900 seconds

- Variance improvement:
  - *Before:* response time 2400 - 8000 sec
  - *After:* response time 800 - 900 sec
Example 2: Performance Drain – Identify the Source

- Slow response reported
- DBA and database focus of delays
- Database problem?
- No – SQL*Net Message identified as source of delay
- 2nd highest wait event
RMM Drill Down identifies source of problem

- Single application generates all SQL*Net Messages
- App on same server as Oracle!
- Answer:
- Misconfiguration – TCP/IP used within server
- Change to IPC, eliminate NIC traffic and 30% of wait time

Solution requires knowing: Which SQL, What Wait Time, Which Resource
Example 3: Scattered Reads

- Situation: LINS06 database - Hourly profile identifies high wait anomaly
- 3-10x higher than other periods - requires investigation

Wait time: 42,000 seconds
10:00-11:00
Drill Down to Key RMM Parameters

Notice scale: > 6000 secs

Individual SQL statements – No database averages

Db file scattered reads

Db file scattered reads
How do you Use it?

Applying RMM for Business Results

1. Identify
2. Allocate
3. Quantify
4. Prioritize
5. Assign
Step 1: Identify

- Find your pain points
- Identify highest impact SQLs *(SQL View and Time View principles)*
- Longest wait times = most significant “pain points” for customers
- Conversely, low cache hit ratios or high latch usage may not impose high wait times for users (so why fix them?)

SQL statements prioritized by Total Wait Time
Step 2: Allocate

- Fix the problems you get paid to resolve
- Allocate impact to real customers (internal or external)
- Allocate wait time to Program, Session, Machine
  - **SQL View** principle makes this connection

Programs Prioritized by Total Wait Time
Step 3: Quantify

- Show the $ benefit
- Enabled by Full View and Time View principles
- Soft dollar savings
  - Data entry clerks
  - DBA time spent in problem resolution
- Hard dollar savings
  - Reduce hardware upgrades
  - Meet SLA’s avoiding penalty
  - Ensure business isn’t lost due to poor performing or unavailable system

Quantifiable benefit of
Tuning a specific statement

![Statistics](image)
Quantify your ROI: Hard Cost Example

Input Data

H/W cost (per server) $60K
Oracle S/W license cost (per server) $80K
IT Expense (per server) 50%
   (Admin, facility, maintenance, project mgt)
Total Cost (Year 1) $170K

ROI Results

Eliminate 35% capacity requirement $59.5K
Reduced External Consulting $16K
RMM Value to Customer (per server) $75.5K
RMM Cost of Implementation (per server) $8K
Generated ROI - 4 months 943%
Step 4: Prioritize

- Pick the right projects
- Cut through the clutter of potential new projects, investigations, and trials.
- Justify your priorities
  - (e.g. We aren’t working on your problem since this other has a higher demonstrable business impact)
Step 5: Assign

- Assign the right people to the problem
  - DBA / Developer / Network Admin / SysAdmin…
- Enabled by Full View principle

- Avoid finger pointing – show the evidence
Conclusion

- Look for what has an impact
- Resource Mapping is more than Wait Time – it must include:
  - SQL level granularity
  - Full Resource granularity
- Isolating the SQL and Resource allows you to find and fix the Root Cause
- DBAs can have an impact and be heroes!
About Confio Software

- Developer of Performance Tools
- Dedicated to helping customers get more out of their existing IT infrastructure
- Oracle product is DBFlash
- Packaged, easy-to-use implementation of RMM
- Based in Denver, customers worldwide
- Free trial at [www.confio.com](http://www.confio.com)
Thank you for coming

Matt Larson

Contact Information

- mattlarson@confio.com
- 303-938-8282 ext. 110
- Company website
  www.confio.com