A HOLISTIC APPROACH TO PERFORMANCE TUNING ORACLE APPLICATION SYSTEMS

Andy Tremayne, Oracle Corporation (UK)

THE METHODOLOGY

Oracle application systems typically consist of a number of highly integrated and interdependent modules that can be implemented in a number of ways; they are rarely static and no two systems are ever alike. Regardless of how meticulous your planning has been, at some point during the system lifecycle you should expect to encounter an emergency reactive performance situation. When faced with a performance issue, or worse still, a plethora of system problems, it is sometimes very difficult to know how and where to start and perhaps more importantly, identify the resources you need to fix the problems.

As with all reactive performance analysis, time is of the essence. You need an investigative method not only to analyze problems quickly, but also to analyze them when they occur. However, by using this method, you should quickly be able to focus your investigative efforts and identify the expertise or skill-set needed to resolve the performance problems. The increasing breadth and depth of the Applications technology stack means that adopting an 'end-to-end' approach, addressing all system components is even more demanding, especially when dealing with a multifaceted problem that may have several contributing factors.

There are a number of problem definition methods, decision support systems, continuous improvement techniques, and other methodologies that can be usefully employed for troubleshooting Oracle based systems. However, many of these approaches are necessarily abstract and very generic. They can be so complex that they require a specialist with problem-solving skills, making their deployment almost as difficult as the problem they are endeavoring to identify. This paper discusses a proven methodology specifically designed to identify and analyze complex and multifaceted Oracle Applications issues. Although this approach is primarily intended for use with Oracle Applications, it can also be used to identify problems with other Oracle based systems.

In outline, it evaluates all of the interacting components by considering the system in its entirety - or 'holistically'. The problem solving techniques loosely models and incorporates some ideas from more generic approaches such as Kepner Tregoe and those proposed by the University of Maryland. This holistic methodology integrates some of the best features from other techniques whilst maintaining a common approach using five problem definition stages - What, Where, When, Extent and Priority. Which approach you decide to use is personal choice. You may modify or extract stages to align with methods or strategies with which you are more familiar.

This methodology documents an approach based on real world experience that has evolved from numerous performance engagements working on fundamentally disparate problems. It has repeatedly proven to reduce the effort and expense of locating and identifying system problems. Quite some time has been expended on maintaining the speed and simplicity of the approach whilst ensuring it remains accurate and sufficiently comprehensive to identify the most complex Oracle Applications issues. Not only does it formalize the problem definition process, but it also enables you to correlate the subsidiary effects and possible consequences of poor performance from other areas.

Performance is one area where throwing personnel resources at the problem rarely helps. If anything, it can lead to contention between the parties blaming each other with you in the middle trying to mediate. If an on-site expert tells you that there are no problems in their area, while they may not see there is a problem, this methodology can provide a definitive insight. Two common tactics when faced with a performance problem are to either engage a database-tuning expert, as this is where there is the most commonly accessed instrumentation, or secondly to throw hardware at the problem. Neither of these would solve anything in the long term and only address the symptoms - not the causes.

There is a lot that you can do to locate and identify the system problems. The value of this method is while you may need an expert in a particular technology stack to resolve the problem; you do not need to be an expert to identify where the problem is. Your goal is to narrow the scope of the investigation as quickly and cost effectively as possible enabling early identification of the correct resources to solve the problem.

The Three Types of Application Systems

It may help from the outset to understand that like a number of other systems Oracle Applications has three different types of processing. Each has fundamentally different processing requirements:

On-Line Transaction Processing (OLTP)

Systems that are mainly used for OLTP are characterized as high-throughput, insert/update intensive systems. They have a constantly growing database and, usually, a high number of concurrent users. Major concerns for this type of system are indexes, the shared pool, well-tuned SQL statements, and pinned packages.

Batch Data Processing

Most batch data processing modules transfer large amounts of data entered by OLTP users. In Oracle Applications, General Ledger posting processes are a good example of this type of processing. Primary concerns are the amount of memory reserved for sorting records, the temporary tablespace which is used for sorts that do not fit into memory, rollback segments and, the required transaction throughput of either detail or summary data. Batch jobs and reports that do not complete overnight cause on-line performance problems during the day until they complete.

Report Processing

Reports typically sort large amounts of data and exhibit similar requirements to batch processes. Poorly written custom reports can consume huge amounts of memory and CPU. The General Ledger Financial Statement Generator is an example of a very flexible tool that allows code to be written very well, or very badly. Main concerns are temporary tablespaces, the efficiency of the SQL statements and the transaction throughput.

The Five Steps

The approach to defining problems forms a very small but pivotal element of the overall problem resolution process. Regardless of whether you are new to the problem or working with it for some time, use this process to give yourself a fresh perspective, in essence stepping back and examining the whole picture. This process works equally well in problem solving regardless of whether you are an external consultant or an employee.

1. What: What is the exact problem, the nature of the problem and what part is specifically under performing?
2. Where: Where does the problem occur?
3. When: When does the problem occur?
4. Extent: How many users or locations are affected? How much functionality is affected? Is the problem isolated?
5. Priority: What is the relative importance of this problem?

The five steps ensure you fully understand the issue, what has been done to resolve the problem and the outcome of that work. This is especially important for Oracle Applications, as there are not only quite a number of modules, but also a number of ways that they can be implemented and used.

Most people will think of asking these questions and have probably done so a number of times already. However, by extending each question to ask the 'converse', you may reveal clues that have never been apparent before. For example, where else might you reasonably expect the problem to occur but it does not? Using converse questions allows you to narrow down the scope of your investigation by excluding those parts of the system that are not affected. Minimizing the scope of your investigation is of paramount importance when dealing with complex problems. It allows you to quickly disregard those parts of the system that are not a part of this issue or alternatively, it can identify parts of the system that do not need to be investigated.

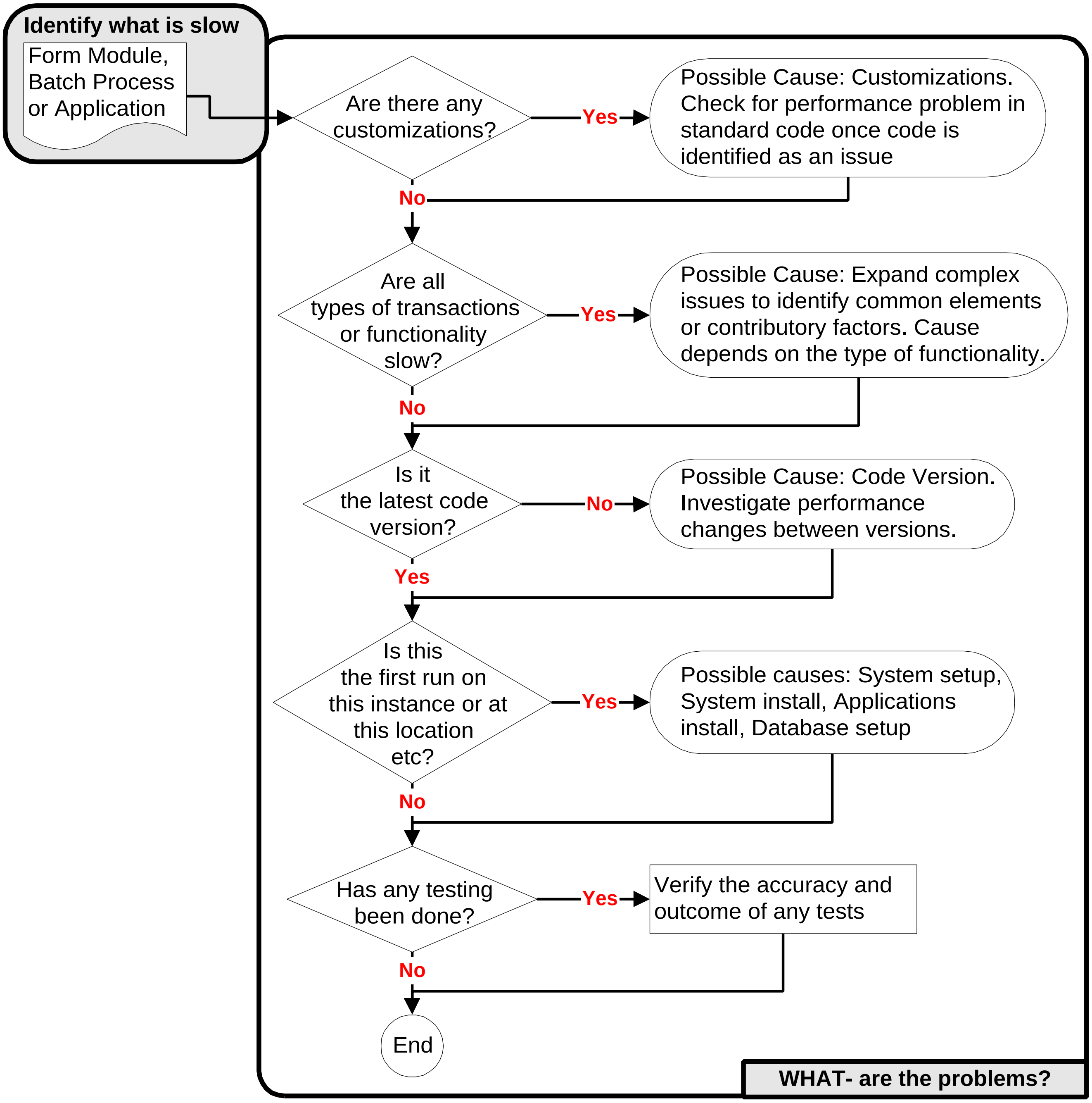
Problem Definition

Before starting any performance review, three essential areas need to be appreciated:

- Accurate problem definition is key to identifying the source of performance issues and will help focus the tuning effort. A host of questions need to be answered not only defining the problem but also the characteristics of the problem.
- Performance target levels to support the critical business functions need to be calculated and agreed. This will provide the definition of success.
- It is essential to identify the different system components and their interaction. There are many aspects to tuning and an appreciation of the whole system ensures the tuning focus is impartial.

WHAT

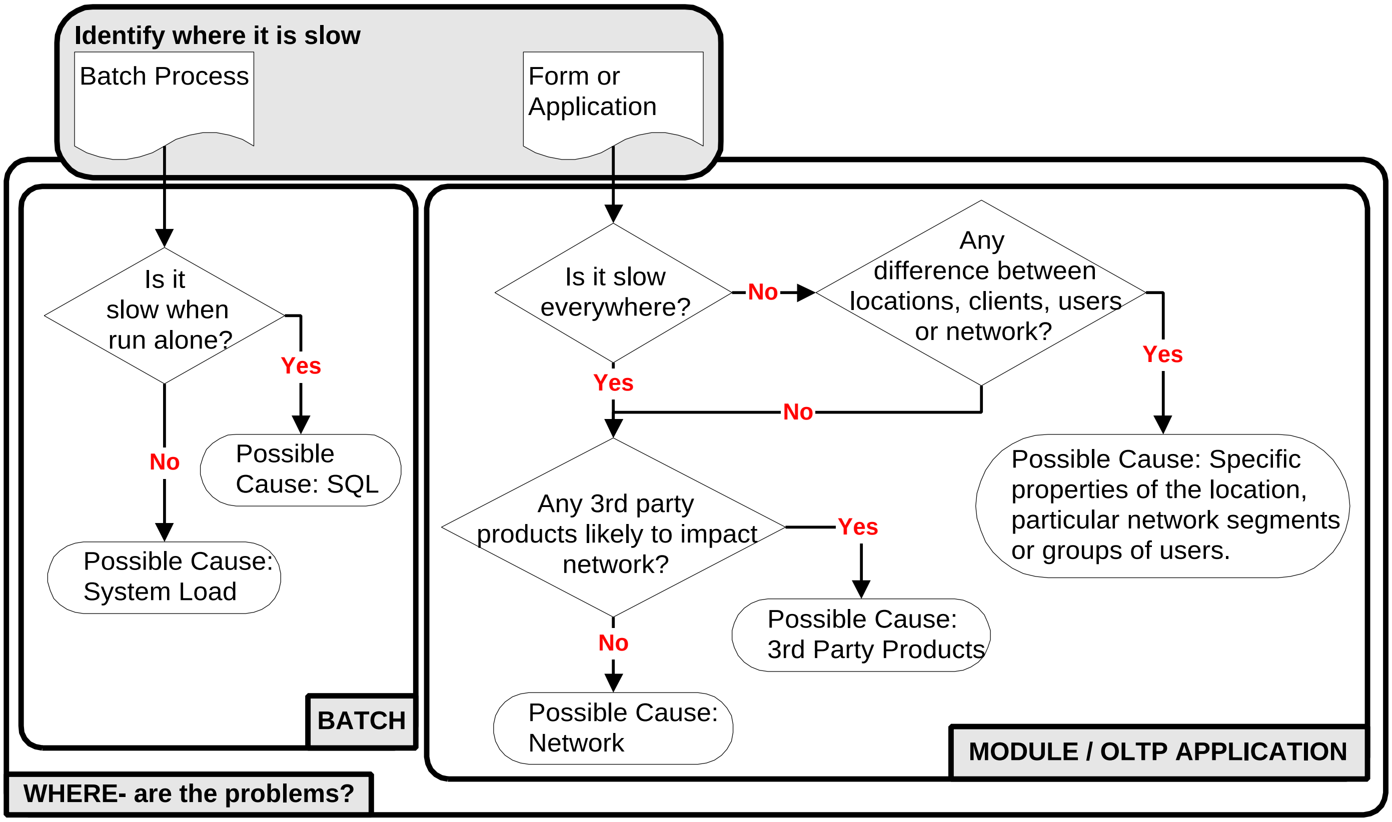
The questions in this step try to specifically characterize the nature of the problem ensure you have a full understanding of exactly what the problem is. What is the problem? What is specifically under performing? What is the nature of the problem? Describe the problem in terms of application modules and/or specific transactions. The answer that the database is slow is not acceptable. The answer that all online applications users in any location using any form experience performance problems during peak load, is much better and would lead to a review of all of the major components - server, database, middle tier, network and clients. State the problem clearly using precise terms and metrics, and if necessary, module version numbers and a characterization of the data that can be used to recreate the problem.


Identify what is slow
Form Module, Batch Process or Application
Are there any customizations?
Yes
Possible Cause: Customizations. Check for performance problem in standard code once code is identified as an issue
No
Are all types of transactions or functionality slow?
Yes
Possible Cause: Expand complex issues to identify common elements or contributory factors. Cause depends on the type of functionality.
No
Is it the latest code version?
No
Possible Cause: Code Version. Investigate performance changes between versions.
Yes
Is this the first run on this instance or at this location etc?
Yes
Possible causes: System setup, System install, Applications install, Database setup
No
Has any testing been done?
Yes
Verify the accuracy and outcome of any tests
No
End
WHAT- are the problems?

Where

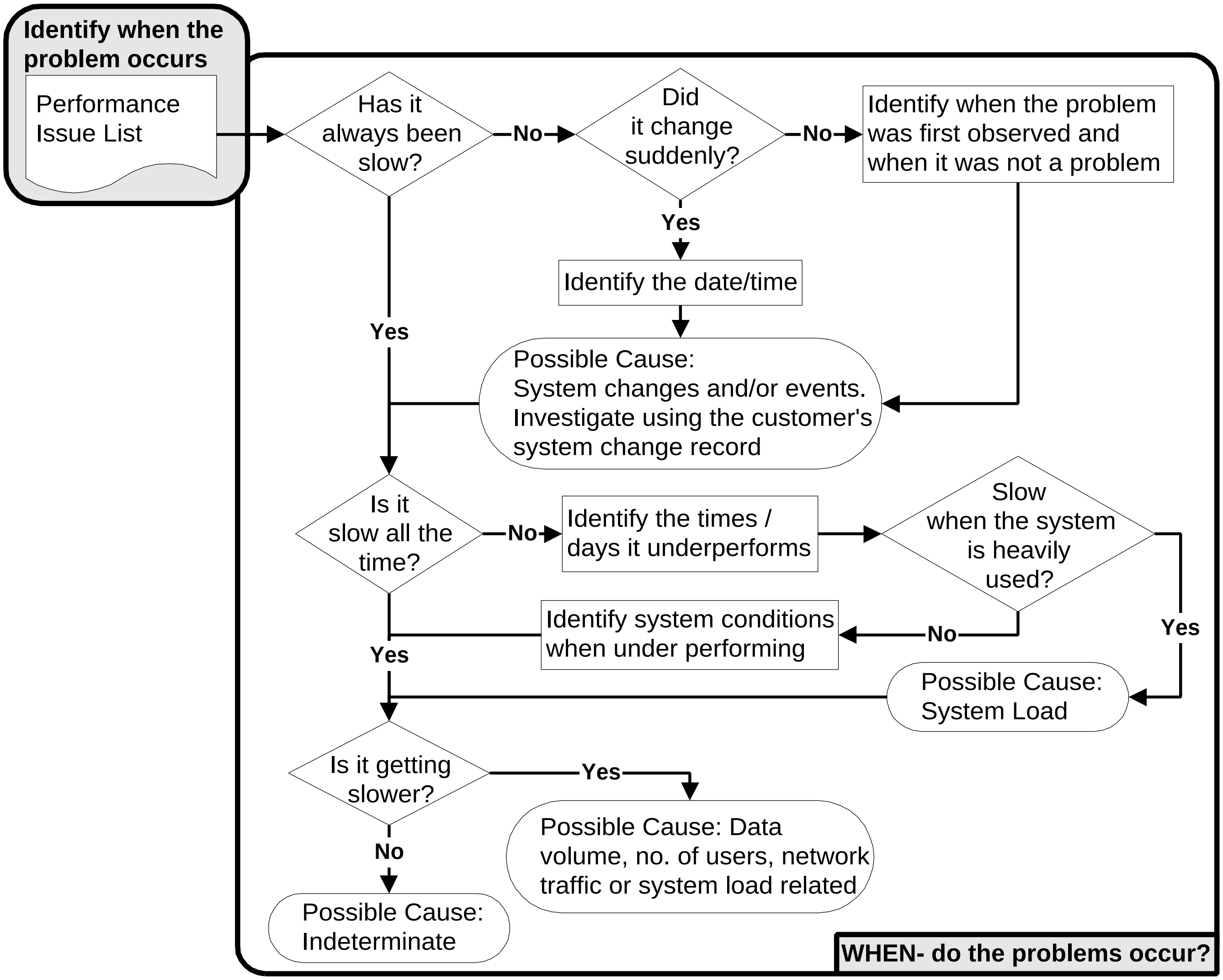
Where does the problem occur? If the problem occurs with specific users, specify the locations and list differences between them. However, this level of detail may not be required if this is a system issue such as on the middle-tier or when running concurrent jobs on the server. Each item in the problem list needs to be assigned to one of the two categories, batch or other, before investigation commences; the system is excluded from this stage as there is typically only a single system. During this stage, the holistic approach ensures that the complete system is evaluated.

It is essential to identify where the problem exists and if there are locations where performance is adequate. It follows that it is important to evaluate not only the users expectations and the relative speeds at which they work, but also the equipment they are using and where they are located. Any variation establishing that performance is satisfactory in some areas completely changes the standard tuning paradigm and fundamental approach to tuning. Although this section specifically excludes instances, if the customer has a number of instances on the same machine, you might want to investigate each environment.


Identify where it is slow
Batch Process
Form or Application
Is it slow when run alone?
Yes
Possible Cause: SQL
No
Possible Cause: System Load
BATCH
Is it slow everywhere?
No
Any difference between locations, clients, users or network?
Yes
Yes
No
Possible Cause: Specific properties of the location, particular network segments or groups of users.
Any 3rd party products likely to impact network?
Yes
Possible Cause: 3rd Party Products
No
Possible Cause: Network
MODULE / OLTP APPLICATION
WHERE- are the problems?

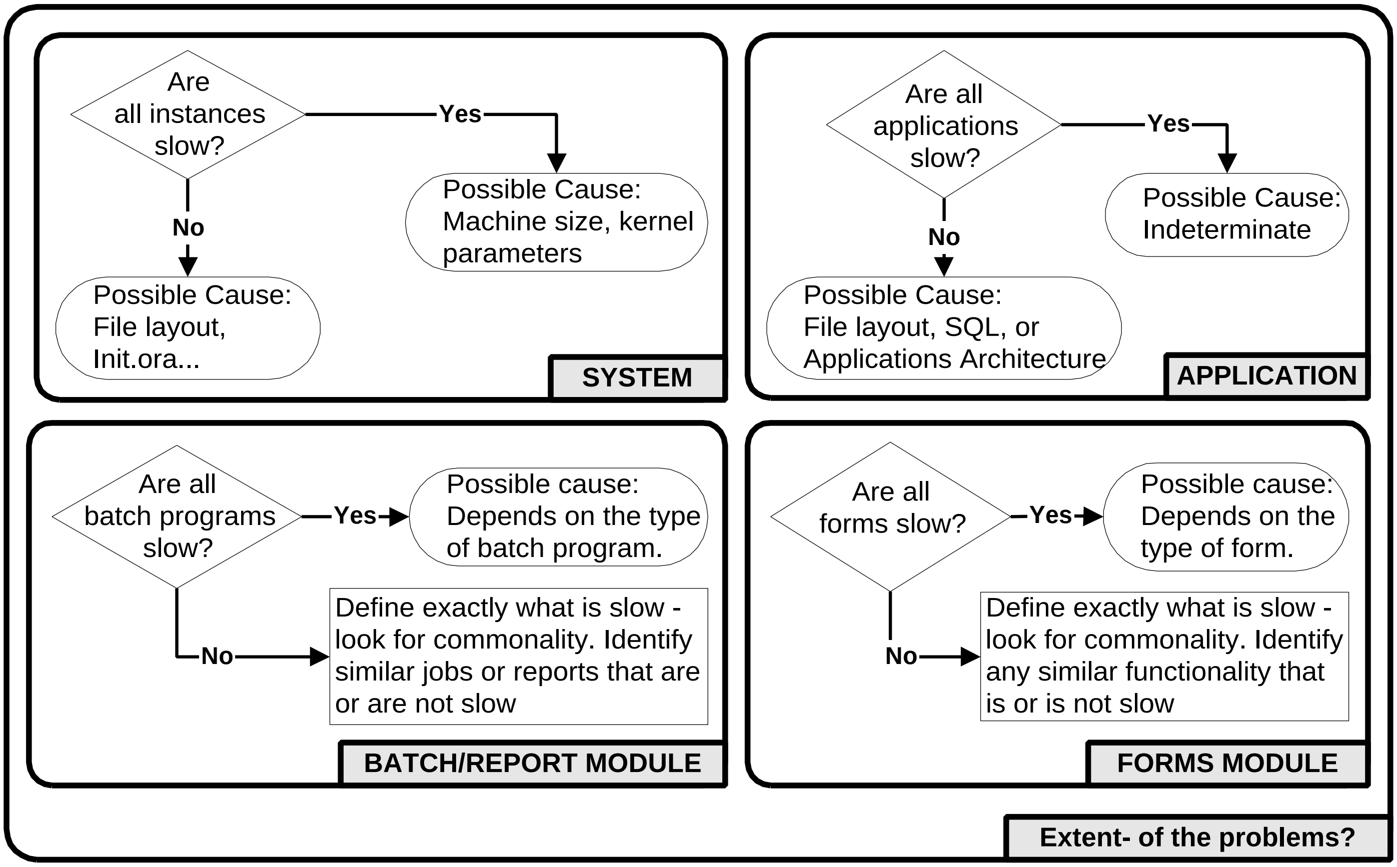
When

When does the problem occur? Note the times of day or dates when the problem occurs. Link the dates to specific events or parts of the Applications or business processing cycle. Identify if the problem has always existed, whether there was a sudden or gradual deviation, and if the problem persists when run alone. Identify other environmental factors and system usage or pattern when the problem is reproducible and possibly relate these to historic events, module versions, and data volumes if the problem has not always existed.


Identify when the problem occurs
Performance Issue List
Has it always been slow?
No
Did it change suddenly?
No
Identify when the problem was first observed and when it was not a problem
Yes
Identify the date/time
Yes
Possible Cause: System changes and/or events. Investigate using the customer's system change record
Is it slow all the time?
No
Identify the times / days it underperforms
Slow when the system is heavily used?
No
Identify system conditions when under performing
Yes
Yes
Possible Cause: System Load
Is it getting slower?
Yes
Possible Cause: Data volume, no. of users, network traffic or system load related
No
Possible Cause: Indeterminate
WHEN- do the problems occur?

EXTENT

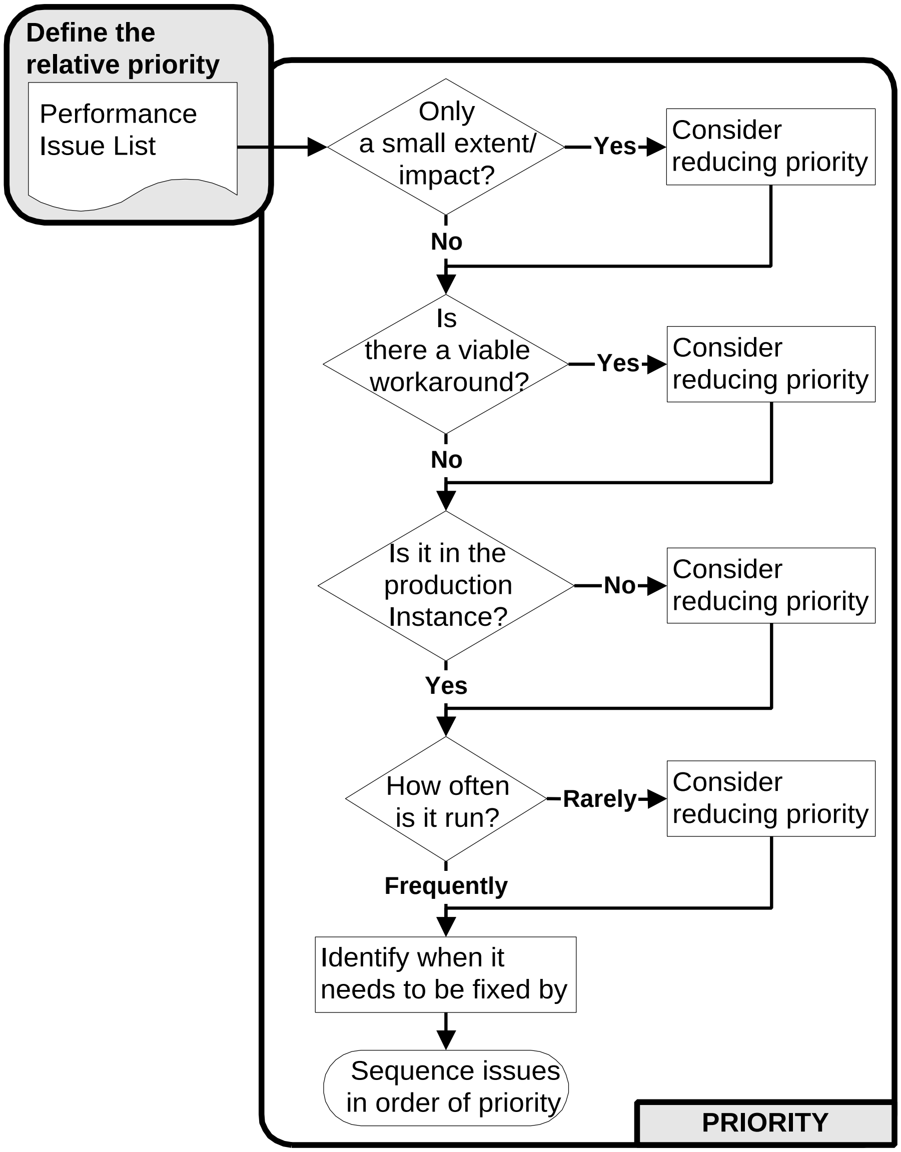
How many users or locations are affected? How much functionality is affected? Is the problem isolated to a single database instance, application, or module? When multiple parts of the system are suspect, look for problem commonality, and try to narrow your investigations to one or two threads. If a problem appears to be isolated, try to establish what makes it different from the remainder of the system.


Are all instances slow?
Yes
Possible Cause: Machine size, kernel parameters
No
Possible Cause: File layout, Init.ora...
SYSTEM
Are all applications slow?
Yes
Possible Cause: Indeterminate
No
Possible Cause: File layout, SQL, or Applications Architecture
APPLICATION
Are all batch programs slow?
Yes
Possible cause: Depends on the type of batch program.
No
Define exactly what is slow - look for commonality. Identify similar jobs or reports that are or are not slow
BATCH/REPORT MODULE
Are all forms slow?
Yes
Possible cause: Depends on the type of form.
No
Define exactly what is slow - look for commonality. Identify any similar functionality that is or is not slow
FORMS MODULE
Extent- of the problems?

Priority

Broadly categorize the issues into three areas such as those critical to sustain critical business functions, those that are essential but won't cause a direct business failure, and the remainder, which are desirable. Consider postponing issues that cannot be recreated or reproduced during the current part of the business lifecycle. Agree a sequence based on the extent, frequency of occurrence and business requirements.

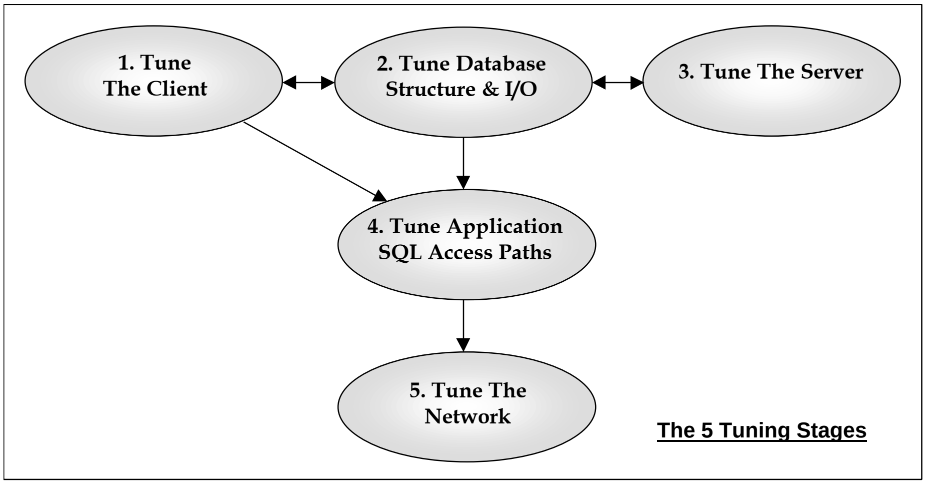
This step may occur during natural conversation for each of the issues but you will need to revisit the step at the end of discussing all of the issues. For this reason, there is an optional link during the What, Where, When and Extent discussion and a mandatory link once you have completed the step.


Define the relative priority
Performance Issue List
Only a small extent/ impact?
Yes
Consider reducing priority
No
Is there a viable workaround?
Yes
Consider reducing priority
No
Is it in the production Instance?
No
Consider reducing priority
Yes
How often is it run?
Rarely
Consider reducing priority
Frequently
Identify when it needs to be fixed by
Sequence issues in order of priority
PRIORITY

The Five Tuning Areas

Oracle Applications tuning starts with a pre-built application, the design, and SQL optimization have already been completed by the development organization. There are several application tuning methodologies but these are not generally appropriate for dealing with pre-built Applications. The reactive approach below is targeted at the areas that yield the greatest return with the least implementation cost. The exact sequence of steps is flexible depending on the type of problem - the sequence is defined during the audit phase.

Although there are five stages, tuning is not always a sequential process and occasionally areas have to be revisited or cannot be fully investigated as there are other bottlenecks in the system that need to be dealt with first. Ideally, one or two people will perform the entire investigation together. This approach has the benefit of knowledge transfer, quick impact analysis and problem consolidation across their particular areas of expertise. There is an inherent problem if several people are involved as each is usually convinced that the problem is not in their area and the whole process only achieves limited improvement or tends to fail.



Tune the Client

The underlying assumption in performance tuning is that the equipment is adequate to achieve an acceptable performance given some patches or major tweaks. However, it is not possible to make a highly tuned 486 perform as well as a 300MHz Pentium but one way around the problem, which is typically overlooked, may be reallocation of fast PCs to power-users.

There are several different versions of Windows and they may need to be configured differently on different clients. Tuning the client is a relatively quick, easy, and definitive step. Patches are available for all but Windows 3.x and a tuned client can return up to 66% performance increase for Smart Client and around 40% for internet computing. General Protection Faults (GPFs) can be reduced by 80% using an appropriate boot configuration. Not only does tuning the client remove a major bottleneck but it also ensures you have a test machine that can be used with confidence for benchmarks. Plugging this directly into the server eliminates any network bottlenecks, immediately isolating the tuning exercise to the database, application, and server.

Smart Client Benchmark Information

It is impossible to establish if there is a performance problem without some benchmark information. The table below shows 'real-world' benchmark figures measuring the time it took from clicking the applications icon to the time it was possible to enter the username in the logon screen; these timings are similar for opening a large workbench form. There is minimal database access during this operation and therefore only the client is tested.

It is important to note that these are the worst timings that are achieved anywhere in Oracle Applications and that most small screens only take a few seconds to open. Also, purchasing clerks, for example, will only login once or twice during the day. Typically, the screen opening times only affect management who visit many screens in their normal duties.

Smart Client (Tuned)				
166MHz	**32M**	**32M + SP**	**64M + SP**	**64M + SP + FA**
Windows 3.x	24 secs	not available	not available	23 secs
Windows 95	28 secs	18 secs	16 secs	14 secs
Windows NT4	28 secs	16 secs	12 secs	9 secs 7 secs* 4 secs*

- SP=Service Pack 1 for Windows 95, Service Pack 3 for Windows NT 4 (*SP 5 is available)
- FA= Removal of Forms Font Aliasing and other general tuning as defined below

- 7 seconds using a parallel enabled network card
- 4 seconds using a 300MHz PC with a parallel enabled network card
- Similar results are achieved when reducing memory from 64M to 48M for Windows 95

Figures scale linearly for different processors. The timings demonstrate the importance of memory and Service Packs for the Smart Client environment. The latest service packs are available from http://www.microsoft.com:

- Win 95 - Service Pack 1
- Win 98 - Second Edition
- Win NT 4 - Service Pack 5
- Win NT 3.51 - Service Pack 5

Service pack 5 is available for NT and is in use at a number of sites. It fixes a number of issues introduced by Service Pack 4 that has been reported as the cause of a number of severe problems. You need to consider applying the hot-fixes to Service Pack 5, especially if you are using it on a middle-tier. There are a number of known issues with some makes of computer and you should check with the manufacturer for known issues. Service pack 6 will be available soon. If you use Microsoft Word 97, apply Service Packs 1 and 2a fixes to prevent it hogging the CPU.

INTERNET COMPUTING BENCHMARK INFORMATION

This table provides figures for the time it takes from clicking the AppletViewer to being able to login.

10.7 internet computing	**Latency**		
Application Startup	**6mS**	**300mS**	**1400mS**
133MHz Win 95 48M	66	67.7	80
233MHz Win 95 128M	30	36.5	53
300MHz NT 128M	25.5	29.4	35
400MHz NT 128M	21.4	26.5	35

Note that there were a significant number of retries on the 1400mS network.

In these tests, the Oracle Forms client Java applets were loaded from the remote Forms Server rather than a local file server. Downloading from a local file server would have consistently achieved the best times. It is apparent that CPU speed compensates for high-latency situations and there is only a minimum of performance degradation over very high latency networks when used with 300MHz clients and above. Although large amounts of memory were used on the clients, this was not necessary. Finally, Jinitiator 1.1.7x produces substantially better figures.

DISK FRAGMENTATION

Hard disk fragmentation occurs over a length of time in all Windows environments and in fact upgrading from Windows 95 to NT possibly starts life with a fragmented file system due to the structure and placement of the Master File Table. Fragmentation means the files are stored in non-contiguous sectors on a disk and so it takes longer to scan the hard disk to locate and assemble all of the file fragments. Severe fragmentation has quite an impact on all applications on the client. Experience shows 4% is measurably detrimental.

There are two main considerations. Firstly, defragment before installing Oracle Applications Smart Client code or the AppletViewer. Secondly, disable the swap file before defragmenting as the swap file fragments are immovable and will result in fragmentation as the swap file shrinks and expands. Many windows defragmenters do not consider the optimal positioning of files and will simply defragment based on the directory structure rather than frequency of file access. This in itself can introduce a significant delay in a file system suffering from 'file sequence fragmentation'.

VIDEO

Video cards are typically restricted by the available video bandwidth. This relates not only to the type of bus, but also the video drivers that govern the data transfer rates, throughput, and transfer modes. The speed of the Peripheral Component Interconnect (PCI) and Advanced Graphics Port (AGP) bus are faster than Industry System Architecture (ISA) but to make this change may mean changing the motherboard. However, up-to-date video drivers can provide around 20% video improvement, even on video cards that are only a few months old. Using 256 colors, instead of more, can save around .5 to .75 seconds when opening a form with Smart Client, even on PCI cards.

SCREEN SAVERS & WALLPAPERS

Screen Savers, including some supplied with Windows, use up to 8M of system memory that is then not available for applications, or causes an additional 8M of paging to the swap file. This is also true of wallpapers and high-quality graphics/pictures. Ideally, use the blank screen saver or none at all and use a plain wallpaper or 16-colour picture, both of which use the same amount of bandwidth and memory.

INSTALLED FONTS

When Windows starts up, it attempts to load all of the fonts located in the Windows font directory. Then, when they are used, they are rendered as bitmaps, further increasing the size of the font cache. It follows that more fonts require a larger cache but the affect of the font cache with Applications appears to grow exponentially after about 50 fonts are installed causing unnecessary swapping. Reduce the number of working fonts to 50 or less or use a font manager so that only a sub-set of fonts is used at any one time. This is most important for Smart Client but also affects internet computing on a machine with little memory.

CPU INTENSIVE TOOLBARS

Several office-type application suites have floating menu bars containing application icons - typically known as shortcut toolbars. Not only are they always in the way, slowing productivity, but they also use an enormous amount of CPU as they are frequently checked. This is easily demonstrated by playing pinball or any fast game with and without them enabled. It is actually quicker and easier to use keyboard shortcuts to activate applications. Other associated items, such as the 'fast-find' and 'fast start' programs are also rarely used and should be removed.

SPEEDING UP DATA ENTRY

A 10%-15% increase in data-entry throughput can be achieved by speeding up the keyboard response speed. This can be done in two ways. Firstly, in the BIOS, ensure the Keyboard Rate is set to 30 characters per second and Keyboard Delay is set to a minimum. If these settings are not available, investigate obtaining a full-bios setup program. Secondly, in the Windows Control Panel / Keyboard, set the Repeat Delay and Repeat Rate to optimal.

FONT ALIASING

Oracle Applications Smart Client forms use font-aliasing controlled by the *uifont.ali* file in the Applications\Tools\Common folder. Typically, this is very small file of only two lines that lists alternative fonts by style, weight, width, and character set. The reason for using font-aliasing is that some forms were developed with Helvetica (on Motif) and some with MSSansSerif (on Windows). If both of those fonts are installed, then you may see a slight difference between some of our forms, but it is very slight and you would have to look very hard.

Deleting or renaming this and any others files with a .ali extension can speed up the start-up time of a form, or changing between screens - as much as one or two seconds. Oracle's internet computing does not have a toolkit and therefore this file is never read.

TWEAKUI

Microsoft PowerToys is a free suite of utilities that extends many of the standard Windows features. It enables you to make some dramatic performance improvements to the Windows interface, such as increasing the data-entry and menu speeds without needing to directly manipulate the registry. In Windows 95,it will record General Protection Fault errors automatically. This is discussed further in the section on GPFs. Another very worthwhile option is to increase the menu speed. This, with the keyboard speed improvement above gives the impression of a much faster client.

What Next?

Having tuned the PC, it should be directly connected to the same switch or device as the database server. Once the database, server, and applications SQL access paths have been tuned, the benchmark timings achieved using this approach, will be the optimum for the installation. If the timings are within targets then it is still worth pursuing the investigation. If the timings are outside the acceptable tolerance range for the business to function, then an alternative solution should be found. If the timings are acceptable, then moving the client away from the server one hop at a time will establish where any network problem is.

The remaining investigation areas are the server, the database, and the application SQL access paths. Once these components have been tuned, the client effectively becomes a network benchmark tool.

Tune The Database Structure & I/O

If not testing against the production system, then the test system should be representative of the production or target implementation. Even tuning application SQL can be dependent on realistic data volumes and data-distribution. Several factors are critical for Oracle Applications. For example, the analyze strategy, block size, hit-ratio, pinning strategy, and adherence to Oracle Flexible Architecture (OFA) have a major effect on I/O bottlenecks and resource contention.

Information Sources

The UTLBSTAT.SQL and UTLESTAT.SQL scripts provide some useful views of the whole of the system and in particular, how the internal Oracle memory is working. The v$parameter table and database alert log provide sources for standard database information and the concurrent manager and process log files also contain a lot of information for Oracle Applications. There are also several instance level and session level views, which are useful during the tuning process.

Analyze The Database

The two available database optimization modes are rule-based (RBO) and cost-based (CBO). Rule-based optimization uses a fixed set of ranked rules to choose an execution plan based on the syntax and structure of the SQL statement whereas the cost-based optimizer uses statistics that reflect the data in the tables. In general, using the cost-based optimizer safeguards against reindexing when data volumes change. Oracle Applications mandates the optimizer_mode initialization parameter is set to RULE as the majority of the SQL statements have been tuned for this optimizer. However, this optimizer is automatically overridden by the cost based optimizer when Applications SQL statements include cost-based hints - do not change the optimizer_mode.Hints are especially prevalent in 10.7 /11 Receivables and Project Accounting and correctly analyzing these schemas typically returns a 40% performance increase in many areas.

In the examples below, the first may be beneficial in a report, the second when returning information to a single-record form.

```
SELECT /*+ ALL_ROWS */ empno, ename, sal, job FROM emp
SELECT /*+ FIRST_ROWS */ empno, ename, sal, job FROM emp
```

For cost-based optimization to be effective, the necessary statistics for the tables and indexes must reflect the exact nature of the data distribution. The statistics may be based on computation of the whole table or an estimation based on a sample of the rows. Estimation has the benefit that it is faster than computation and uses less space to sort the information. The statistics need to be periodically compiled using the ANALYZE command and are stored primarily in user_indexes and user_tables in the data dictionary.

Example analyze command syntax:

```
ANALYZE TABLE gl.gl_je_lines COMPUTE STATISTICS;
ANALYZE TABLE gl.gl_je_lines ESTIMATE STATISTICS SAMPLE 20% PERCENT;
```

Always use the full-compute method unless there are space or time constraints when working with large tables. If an inadequate sample is used, then performance may reduce by up to 40% and so it is imperative to ensure that the sample-size is sufficient to reflect the total data distribution. Many argue that a sample-size of 20% is usually almost as accurate as doing a full table scan. The main problem using the estimate approach is that a skewed data distribution will reflect in the statistics; this is quite common in Applications.

To determine how much of a large table needs to be analyzed, consider an incremental approach starting at 20% and increasing by 10% each time, taking a snapshot of the statistics at each level. Comparing the variance in the snapshots will indicate when an appropriate sample-size has been analyzed. Note that a full compute is automatically invoked when using a sample of 50% or above. If the data is analyzed at the correct level, where hints have been used the use of the cost-based approach should be at least as good or better than with rule-based optimization. The question of how often to analyze is more difficult as it requires a level of understanding of how quickly the data distribution changes within the business. Again, comparing snapshots for volatile tables will determine not only the frequency at which to analyze, but also 'sets of tables' to be analyzed during each scheduled run.

The 10.7 Applications script $AD_TOP/sql/ADXANLYZ.sql by default uses an estimated sample of 20%; in R11 it uses 50,000 rows. In order to measure a performance change, or to address a performance problem in a particular model, Trace can be used to identify candidate tables. Incidentally, Trace will also show that the cost-base approach is used for some recursive SQL; however, you should never analyze the SYS schema. Note that all tables in a SQL statement using hints need to be analyzed to enable the cost-based approach. If problems ensue, then the statistics can be deleted immediately. 10.7 and R11 contain a smattering of hints so analyzing all modules may be beneficial. Finally, from the sql command prompt in 7.3 or higher, you should alter the session to set the sort_area_size to 5MB or more and set the db_file_multiblock_read_count to 32 (Oracle will reduce this value if it is too high).

Block Size

For best performance, all the data to satisfy a query would exist in a single block or consecutive blocks that had been read into the read-ahead I/O buffer during a sequential read operation, commonly known as a full-table scan. This implies that ideally the database block size should be equal to, or a multiple of, the operating system block size.

There is a tremendous amount of speculation regarding block size and there are many factors involved with data storage. Many discuss record chaining, which occurs when a record is too big to fit into a single block. Excessive row chaining indicates that the block size is too small but this is not the main reason why such a dramatic effect occurs when increasing the block size for Oracle Applications.

Changing the block size from 2K to 8K can increase performance by 40% for batch programs and 'large data-sort queries', and by 4% to 5% for 'light data-sort queries'. Why do we get these improvements? Although there is a reduction in chaining of large data rows, the overriding factor is table reorganization and, the reduction in height of B-Tree indexes and associated searches, which are used with almost every database access. There are several examples where a customer has rebuilt the database and reaped the benefit.

An inextricably linked topic is the maximum number of database blocks read during each I/O read-operation as defined by the db_file_multiblock_read_count initialization parameter. Invariably this is set incorrectly and there appears to be confusion regarding the small, medium, and large settings in the init.ora. If this parameter is inadvertently set too high, Oracle will reset the value and will use the value from v$parameter. The maximum number of blocks that can be read in one I/O request is defined by the system: max_io_size/db_block_size.

Setting that value correctly based on the max_io_size returns a performance improvement for all aspects of Applications performing sequential read operations where the table size exceeds the _small_table_threshold initialization parameter (default 5 blocks).

Another interesting point regarding the max_io_size is the ideal system stripe size. Consider a system with a system read capability of 64K, an 8K-block size, and db_file_multiblock_read_count of eight. This means that oracle would request 64K in a single I/O read-operation. This being the case, using a stripe size of 16K would perform four system I/O for each Oracle sequential read request. It follows that not only should db_file_multiblock_read_count * db_block_size = max_io_size but also the stripe size should equal the max_io_size.

Package Pinning Strategy

Each Application has several packages that can be loaded into the shared SQL and PL/SQL memory areas. It can take a considerable time to load and parse a package when it is referenced but having packages resident in memory yields the best performance for OLTP users. The time to load and parse packages is proportional to their size.

Theoretically, pinning all packages improves performance and a pinned package cannot be swapped out of memory unless they are explicitly unpinned at which time they will be normally aged out of the buffer. Pinning all packages is not always possible due to the amount of available memory. It is not necessary to pin small infrequently used packages or those used by long-running batch processes or reports, as these will be implicitly pinned for the duration of the task. However, if packages are not explicitly pinned, very large packages may not be able to load due to insufficient or fragmented memory within the shared buffer pool.

Further, it has been found that pinning too many packages in an uncontrolled manner can degrade system performance by as much as 50%. Importantly, when the amount of shared pool memory falls below a critical level, the server has to dedicate resources to managing the limited space available. When a very large package is referenced, a huge amount of flushing and buffer coalescing takes place to create a contiguous area of free memory known as a 'chunk'. If the chunk is not large enough and there is still insufficient memory, the package will fail to load, an ORA-4031 error will occur and the failure will be recorded in the x$ksmlru table.

Pinning rarely used packages is very wasteful of the shared memory area and it is better to only pin a few packages - those that are used almost constantly as defined by the highest number of executions in the table v$db_object_cache. Fragmentation is reduced when packages are pinned in descending size order (determined from dba_objects) - this is vital for Applications.

It is essential to ensure that there is sufficient memory to load packages and this is achieved by setting the shared_pool_reserved_size initialization parameter. This specifies the amount of the shared pool to be reserved for SQL statements requiring large memory allocations - including packages. In order that not every request is automatically obtained from the pool, setting the shared_pool_reserved_min_alloc parameter prevents small statements below a certain size being allocated space. The shared_pool_reserved_size should be set to approximately 10% of the shared_pool size and shared_pool_reserved_min_alloc should be set to 5000 bytes if this is different to the system default. Some performance gain has been reported when increasing this to 10000 bytes, but you will need to test this on your system. Monitoring any package load failures in the x$ksmlru table will enable a more scientific approach but this has not been found necessary.

Hit Ratio

There do not appear to be any guidelines for the buffer cache hit-ratio specifically for Applications. From investigations on several systems, users appear to be dissatisfied with hit ratio less than 94%. However, there is also an upper limit to this figure, not the law of diminishing returns but by having a buffer cache that is so large that it takes an inordinate amount of time to search for free space within the buffer. This provides the foundation for the statement that the buffer hit-ratio for Oracle Applications must be between 94.0% and 97.0% on all systems for best performance. Theorists will discount the upper limit, and this may not apply to more recent versions of the database, but this has not been proven by anyone to date. Customers within this range are generally very happy.

The buffer cache hit-ratio is tuned by changing the db_block_buffers initialization parameter. An estimate of the performance increase for a particular number of buffers can be gained by setting the db_block_lru_extended_statistics initialization parameter to the number of db_block_buffers you are thinking of adding. For example set db_block_lru_extended_statistics = 500 and restart the database. The statistics are computed and stored in x$kcbrbh. A similar approach can be used to determine a reduction of the hit-ratio. When this value is set other than zero, there will be a drop in performance and so it should not ordinarily be set.

Database Triggers

Many customizations are done using database triggers, which are easily coded and understood. For versions before Oracle 7.3, unlike procedures, triggers are compiled at run-time and many have caused a significant loss of performance. In general, if more than a few lines of code are used, it is far better to move the code into a procedure (which remains compiled) and call it from the database trigger. Incidentally, triggers are defined as large anonymous PLSQL blocks and can be identified by periodically querying v$sqlarea where command_type=47 and length (sql_text) > 500.

Tune Resource Contention

Multiple concurrent users may create contention for Oracle resources causing processes to wait until resources are available. Block, shared pool, lock and latch contention should all be minimized. They are monitored using standard Oracle database scripts.

Latching

If you are encountering a performance problem which only shows after a period of database activity, and is partially cleared by flushing of the shared pool and completely cleared by restarting the database then it is likely that you are experiencing contention on the library cache latch or shared pool latch. This can be verified by obtaining a Bstat/Estat report and checking the library cache and shared pool latch waits.

In Oracle Applications, these high impact problems have been identified.

Related To Views

Selecting from views uses too much shared cursor memory.

On most platforms, this bug (724620) is fixed in Oracle Release 7.3.4.4, 8.0.4.3, and 8.0.5.2.

Non-Sharable Cursors And Parsing Of Bind Variables

This affects many aspects of Oracle Applications fixes include PL/SQL packages, procedures, Oracle Forms and Reports. To alleviate these issues, a number of application patches must be applied.

If you are not able to upgrade immediately to the appropriate RDBMS version or apply the application patches, then changing the database initialization parameters may alleviate the problem. As there are several permutations of database setup and factors that contribute to the latching problem, it is not possible to apply a standard set of rules to correct the problem.

Cursor_Space_For_Time

When set to TRUE, this pins SQL areas in the shared context area and prevents deallocation of private SQL areas until the application cursors are closed. When set to FALSE, both the shared SQL area and the library cache need to be checked when attempting to find a SQL statement. This parameter has not been established as having a major affect on Oracle Applications performance but clearly, setting this value to true, without increasing the size of the shared pool, compounds the latching problem and you should consider setting it to FALSE. You will need to allocate a shared pool large enough to prevent parse errors whilst ensuring that there is sufficient free memory to prevent extensive swapping.

- If TRUE:

  Increase shared pool by 100%

  ```
  Set ROW_CACHE_CURSORS> = 200
  ```
- If FALSE:

  Increase shared pool by 50%

  ```
  Set ROW_CACHE_CURSORS> = 200
  Set SESSION_CACHED_CURSORS> = 200
  ```

Session_Cached_Cursors increases the number of SQL statements that can be moved, when parsed repeatedly, into the session cursor cache. Subsequent parse calls will find the cursor in the cache and need not reopen the cursor, which is highly beneficial in this case for users who frequently switch between the same group of forms.

Shared_Pool_Reserved_Size

This parameter and shared_pool_reserved_min_alloc are discussed in the section discussing the Package Pinning Strategy. The shared_pool_reserved_size should be set to 10% to provide best performance for all Oracle Application systems.

When the shared pool fragments or if the 'small chunk freelist' is growing excessively, flushing the shared pool will cause a short-term peak in CPU and I/O resources but should provide relief from contention. The frequency of flushing depends on the system usage but if it is necessary to flush during the working day, pin commonly used packages, functions and triggers that execute more than 100 times (in descending size order) will help. This though, may result in insufficient deallocation, which will result in manual unpinning or, in the worst case, a database restart.

TUNE THE SERVER

Server upgrades are generally very expensive and tuning Oracle can resolve many problems. For example, changing the database architecture to comply with the Oracle Flexible Architecture methodology can resolve problems by moving the database files to even out the disk activity. It is essential to know the memory, disk types, configuration, file layout, and system I/O amongst other things. The database and applications need to be tuned before it is possible to determine whether the servers have sufficient capacity for the implementation. Advice from the Oracle platform specific manuals is invaluable. Rather than deal with standard tuning issues, some hot topics are discussed.

DISK I/O: OFA VS RAW PARTITIONS

Oracle Flexible Architecture (OFA) defines a default directory structure and file system layout for database installations on Unix. It is generally accepted that indexes and tables are separated into different tablespaces. If both tables and indexes are all located on one disk, the head would be constantly moving between the two. Locating the tablespaces on separate disks minimizes disk head movement. A useful extension of this approach, which is commonly overlooked, is that it is better to ensure the disks are on separate controllers enabling the operating system to concurrently perform input/output operations through a number of channels.

The increasing use of RAID arrays and logical volume managers does not really make this advice obsolete. While these provide good automatic load balancing across disk devices, some control is lost and if a table tablespace and its associated index tablespace are located in the same RAID disk, an indexed access to a table may produce activity on the same physical disk. The most important point for Oracle Applications is to use several mount points with raid arrays to reduce I/O queuing.

A few customers have decided to use an exclusively raw devices rather than a UNIX file system. Theoretically, using raw devices instead of UNIX file buffering improves disk I/O. A few have done the conversion and noticed a gain in performance of around 10%-15%. However, a database export and import may have achieved the same return, as row chaining is eliminated and the indexes are rebuilt and balanced. Raw partitions also have the disadvantage that they are significantly more difficult to administer. Ideally, there should be as many raw sections as there are data files/tablespaces making changes easier. Having said this, there are situations in which raw devices are the correct choice, however, the only reason for exclusively using raw devices is that disk I/O is the system performance bottleneck that cannot be resolved by tuning the database, server or application.

MEMORY BOTTLENECKS

Since memory access is much faster than disk access, it is desirable for data requests to be satisfied by access to memory rather than access to disk. Tuning memory allocation and management involve distributing available memory to Oracle memory structures. This is done by correctly sizing the SGA, by optimizing the number of database buffers and the data dictionary cache.

A word of warning: If after increasing the amount of Oracle memory there is insufficient for the operating system, the SGA may be paged to disk. If this occurs, trace files will still report logical reads although the operating system has had to fetch the page from disk. Importantly, on a system with limited memory, balance increasing the size of the buffer with the degradation likely and swapping.

PAGING AND SWAPPING BOTTLENECKS

Insufficient paging space can dramatically affect a system's performance. The system can demonstrate such problems as slow response times, failure to spawn sub-processes, or complete system hanging. Most UNIX systems should have total swap space between two and five times their physical memory size. The minimum is twice physical memory; Oracle Applications requires paging space of at least 3-4 times of the amount of physical memory.

CPU BOTTLENECKS

If the system is found to have a CPU bottleneck, the workload should be examined to determine if any of the work could be moved to off-peak periods. For example, batch processes or non-critical reports could be run overnight. By moving less critical jobs to periods of the day when the system is lightly loaded, you can improve throughput for users who run during those peak-demand periods. If there is no scope for rearranging the workload, investigate and tune the applications code before considering a hardware upgrade.

Tune SQL Access Paths

When dealing with system performance, application SQL tuning should be considered a special case when processes do not either complete within a required period or place a significant load on the system to the detriment of other activity. Tuning activity in this area generally deals with isolated module problems.

Expensive SQL

The first stage of any Application performance analysis has to be to check that the code is efficient. The most expensive system SQL activity can be determined using buffer_gets/executions in v$sqlarea. This only contains the first 1000 characters of the SQL statement and is usually linked using the hash_value to v$sqltext to retrieve the complete statement. Finding the source code is still a lengthy process though v$sess_io may provide a useful link to the process that executed the statement.

The normal approach would be to resolve the statement bind variables, trace the execution in SQL*Plus and review the TKProf version of the trace file. However, Oracle 7.3 includes the autotrace function in SQL*Plus which immediately displays the explain plan and other information about a SQL statement on screen, in the SQL session, making tuning statements a very quick process. Be aware that statements using the like operator may never be run exactly the same way again.

Assuming the SQL code is efficiently written, the next step is to look at the anomalous data volumetric and data distributions identified during the investigation or audit stage. For example, 1,000,000 customers would be considered a large number in any implementation. Additional indexes may help but should be very carefully considered before they are added. One of the biggest problems in Applications is finding other modules that are affected by the change. There are several papers and books that discuss standard indexing practice but it is worth remembering that not all full table scans are noteworthy. For example, scans of very small tables, smaller than the _small_table_threshold initialization parameter, or those that are read in a single system I/O read-operation should not be considered as candidates for optimization. The utloidxs.sql script may be used to analyze possible new candidate columns for indexing.

Information Sources

Trace may be used at instance or session level to generate useful information whilst only minimally affecting performance. Trace may also be enabled for a forms session or report, within a stored procedure or in a concurrent program. For any of these, the use of the initialization parameter timed_statistics = true is highly recommended as this information is usually requested by development when bugs are logged. Whenever you generate a TkProf file, you must check the elapsed trace time is the same as the real elapsed time. This may not be the case where packages are called and the raw trace file should be used to provide the complete picture. You will generally find that the 'unrecorded' TkProf time is spent in packages.

For more advanced troubleshooting, it is possible to increase the level of information recorded in the trace file by setting event 10046 using various levels.

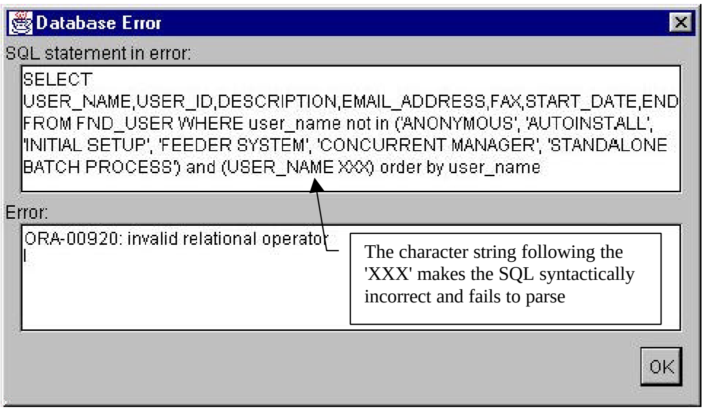
Level	Included Information
null or 1	Same as standard SQL_TRACE functionality
4	As Level 1 + Bind variable information
8	As Level 1 + Session Wait Information (can be used to identify full table scans)
12	As Level 1 + Bind variable + Session Wait Information

Some important points to note are that when the instance is running normally, the data dictionary cache, library caches and Shared SQL Area will contain data. When the database is restarted, these are effectively empty and this affects how you should interpret trace file information. For example, immediately after a database startup, increased I/O from physical disc reads for the data dictionary and data access may have higher physical reads as even small commonly accessed, normally cached, tables will have to be read into memory. Initially following startup, there is more shared memory available. This means that sorts and other operations, which may result in quite high traffic to the Temporary data file may actually be all done in memory and so the statistics may not be representative of what happens on the system when it is running normally.

FINDING DYNAMIC FORM SQL: HOT TIP!

Occasionally, when dealing with development issues or support questions, it is useful to find the dynamic SQL being executed when a form performs a query without having to use trace. This works in all applications versions and not only lists the tables, columns, constants, and ordering, but also includes restrictions (including folder conditions) on records that are queried. The procedure is very simple:

- Put the form into query mode (Query Enter)
- Enter #XXX in any 'database' field
- Execute the query - This will create an error
- Acknowledge the error, cancel the query, and select Help / View Database Error
- Review the database error which will show the query being executed at the time the form 'crashed'



SQL statement in error:

```
SELECT
USER_NAME,USER_ID,DESCRIPTION,EMAIL_ADDRESS,FAX,START_DATE,END
FROM FND_USER WHERE user_name not in ('ANONYMOUS', 'AUTOINSTALL',
'INITIAL SETUP', 'FEEDER SYSTEM', 'CONCURRENT MANAGER', 'STANDALONE
BATCH PROCESS') and (USER_NAME XXX) order by user_name
```

Error:

```
ORA-00920: invalid relational operator
```

The character string following the 'XXX' makes the SQL syntactically incorrect and fails to parse

The select statement lists all form fields being retrieved from the database. The where-clause conditions apply regardless of whether an 'open query' was executed or any additional query criteria were used. For example, on the Purchase Order lines Zone, you would find that cancelled lines are not displayed. This can make answering support questions very quick.

This only works if the field containing the '#XXX' is a base-table database field. The error 'Query caused no records to be returned' indicates the field used was not a base-table field. Quantity and date fields are good candidates.

HIGH-WATER MARK

Applications interface tables are prone to index stagnation and a High-Water Mark problem. When data is written to a table, the highest block used, known as the High-Water Mark is recorded in the header block. When data is loaded, the High-Water Mark is used to determine the start point for a data load and records are loaded above this mark, even if the previous set of records has been deleted. The high-water mark is not used during index scans but when a full table scan occurs, the whole table is read up to the High-Water Mark regardless of the number of records in the table.

There are two main methods for detecting a high-water mark. Firstly, the number of empty data-blocks is recorded in dba_tables when the table is analyzed. Secondly, the dbms_space package available in Oracle 7.3 and later can be used. Clearing the problem is easy using the TRUNCATE command when the table is empty.

This is a very important problem area for Applications and should be the first thing to be checked when an interface becomes slower over time. The performance return can be huge. One notable scan of gl_interface took over 20 minutes to return a count of zero records. This was found when average size postings started taking nearly 40 hours to complete. The posting time reduced to about 2 hours after the problem was rectified.

Customized Views

Quite often new views are created for customized work or to simplify complicated expressions used in Oracle Data Query. It is very important when creating views that will be heavily used to ensure that they are based on the base-tables and not on other views. This type of tuning has returned 30% performance increase across the parts of the system where it was referenced. Further, many developers forget to consider the cost-based approach when writing SQL to access the views.

Tuning The Concurrent Manager

When the number of requests exceeds 5000, performance of the concurrent managers and especially the query screens may slow down depending on the system. Normally, to improve performance you should schedule running the purge concurrent requests program to reduce the amount of information held in this table. You may need to maintain many thousand requests on-line during each business cycle. In this case, you need to look elsewhere for performance gains.

The sleep time is the number of seconds that the internal manager waits between looking for new concurrent requests to run. The optimal value is system dependent and is based on the number of worker queues and available CPU. Polling of fnd_concurrent_requests, which uses a full table scan, is wasteful of resources. You should be wary of reducing the sleep time below 30 seconds and should measure the impact on performance.

When you monitor the top 10 SQL statements from v$sqlarea, you may find that 70% are connected with the concurrent requests table and may also associate these with a number of lock issues. Analyzing the fnd_concurrent_programs, fnd_concurrent_requests and fnd_oracle_userid tables using a full compute will enable use of the fnd_concurrent_requests_n2 index, substantially improving performance. For release 11, you should check for the existence of an index on the fnd_conc_release_classes table for columns release_class_id and owner_req_id.

If you have a high number of small concurrent requests and experience the problem that other requests are being queued for long periods you should consider creating a separate manager to process these small requests and exclude them from other queues.

Tune The Network

Protocols, topologies, switches, hubs, routers, and bridges make networking a complex area. The use of a compression technology can also introduce problems. To identify a performance problem, only a rudimentary knowledge is required, not detailed information regarding every minute detail of the OSI model. Resolving the problem is significantly more complex but it is also noteworthy that not all problems in this area are related to performance but may be the cause of GPFs.

A thorough understanding of the whole network is essential to resolving problems. An accurate, detailed network diagram showing size and latency of all links, location and number of all oracle users and every network device crossed between the desktop and target database is invaluable. It should show bridges, routers, switches, hubs, topologies, bandwidth, and protocol distribution as well as any firewalls or known bandwidth limitation problem areas and indicate whether any WAN usage is domestic or international. If the Maximum Transmission Unit on any network device has been changed, this should be noted together with the default and new value. Any Network Management Systems, traffic management, or priority queuing set on any device should also be noted.

Generally, network characteristics are analogous to I/O constraints. The amount of data per second that the network can transfer is known as the bandwidth; latency is the time delay to transfer the information. Bandwidth can be checked by comparing the available network bandwidth with the network segment loading which is determined by multiplying the transactions by the average number of bytes for each transaction and by the number of users on the network segment. Latency may be influenced by both device latency and waiting for the network to be available because the bandwidth has been exceeded or collisions have occurred.

When evaluating performance, it is important to obtain a network latency map for a full size packet. The maximum transmission unit size can be determined using the '-f' ping option to prevent packet fragmentation, increasing the packet size whilst there are no errors, and noting any timeouts. For 10SC, this should include the time from the client to the database server. For internet computing this should include the time from the client to the various servers and from these to the database server. The internet computing architecture has overcome the Smart Client sensitivity to latency. Acceptable internet computing performance has been achieved on networks of up to 1000mS latency.

The client, connected to the same device as the server, should be used to produce best-case benchmarks. Once the figures have been derived, it is an easy task to move the client around the network to identify problem areas. If this approach is not used, and access to the network is restricted, a test 'area' should be agreed and used where performance targets have to be achieved. The test location should not have an unacceptable amount of latency or suffer bandwidth problems.

Diagnostic Tools

Details of packet retries, collisions and bandwidth utilization may be obtained by placing routers, bridges, switches and hubs in diagnostic mode where possible. Network monitors and probes are useful in distributed systems, primarily to check that no network resource is being exceeded.

In the absence of a sniffer, most operating systems have commands such as ping, which are very useful for establishing that packets are not being fragmented and for perhaps tracing the actual route, a packet takes. You should always use a full size packet when measuring latency. For example, to send an unfragmented full-size Ethernet packet, use the command: ping -f -l1472 -n50 <host>. The priority of the ping command may have been reduced on some network devices and this should be checked especially where measurements do not appear sensible.

Trace-Route is a route tracing utility that uses the IP TTL field and ICMP error messages to determine the route from one host to another through a network. The problem with this utility is that it tends to show the optimal route rather than the actual route taken. There is no substitute for a network sniffer, preferably, that decodes SQL packets, to determine what is really happening on the network. The most useful module decodes the SQL*Net protocol in addition to measuring SQL*Net response time and tracing packets across the network.

Applications Patches

Applications development has specifically addressed network traffic issues in Smart Client. In prod 10SC prod 15, two missing profile options, jgzz_country_code, and jgzz_product_code, caused excessive message dictionary traffic. This was resolved in bug 556926. Some prod 16 sites to not appear to have these defined. Performance of windows caching generally was improved by amendments to FND75WIN.dll.This issue is dealt with in bug 558704.

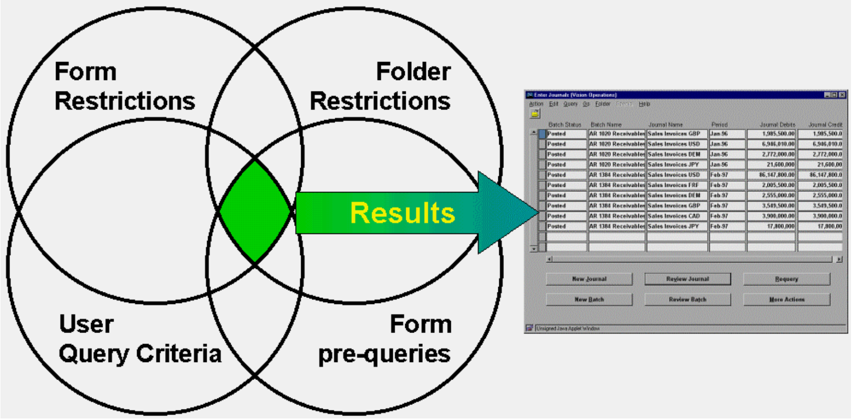
Tuning the Users

Inefficient user queries are an aspect of performance that is commonly overlooked. Teaching a user to query efficiently and effectively reduces the load on almost every component in the system and returns information more quickly. Such techniques are also useful in extending the functionality of screens and screen-reports when using the 'export' function. In general, this type of training should be delivered as a special short course in isolation from other topics.

In one particular Smart Client engagement, the users of a data-processing department were forced to use the keyboard when their mice were taken away. After an initial drop, there was roughly a 15% rise in productivity. This shows that looking at areas other than the Application system can be worthwhile. In this case, the high performance gain was mainly due to the limited number of screens operations that were used.

Querying Forms

All user queries in forms may be subject to restrictions built into the form, folder, or pre-queries in the form code that may be conditionally applied. Simple queries can be performed by entering any query criteria whilst in query-mode. The rows returned to the screen are the intersection of the database records, any in-built form restriction and any query criteria entered by the user.



Batch Status	Batch Name	Journal Name	Period	Journal Debits	Journal Credit
Posted	AR 1020 Receivables	Sales Invoices GBP	Jan-96	1,985,500.00	1,985,500.0
Posted	AR 1020 Receivables	Sales Invoices USD	Jan-96	6,946,010.00	6,946,010.0
Posted	AR 1020 Receivables	Sales Invoices DEM	Jan-96	2,772,000.00	2,772,000.0
Posted	AR 1020 Receivables	Sales Invoices JPY	Jan-96	21,600,000	21,600,00
Posted	AR 1384 Receivables	Sales Invoices USD	Feb-97	86,147,800.00	86,147,800.0
Posted	AR 1384 Receivables	Sales Invoices FRF	Feb-97	2,005,500.00	2,005,500.0
Posted	AR 1384 Receivables	Sales Invoices DEM	Feb-97	2,555,000.00	2,555,000.0
Posted	AR 1384 Receivables	Sales Invoices GBP	Feb-97	3,549,500.00	3,549,500.0
Posted	AR 1384 Receivables	Sales Invoices CAD	Feb-97	3,900,000.00	3,900,000.0
Posted	AR 1384 Receivables	Sales Invoices JPY	Feb-97	17,800,000	17,800,00

Most users know to enter query reduction criteria in fields. Quite often, they will execute multiple queries, or perhaps swap between forms to retrieve the information they require. Better ways of querying can not only speed up data entry for users, but also reduce the load of multiple queries on the system by eliminating the requirement for them. These methods are discussed in the next few examples.

CLEVER QUERIES

This method explains how to query back rows using parameters that are more complex than normal including where fields are specifically null. For example, finding journal headers starting with 'ABC', together with those starting with 'DEF' - all in one query rather than two.

To create a clever query:

- Enter Query Mode
- Enter ':a' in the field to be used in the query
- Execute Query
- A query window will appear in which the search criteria is entered. Click 'OK' to execute the query.

The following operators may be used:

Operator	Description	Comment
= xyz	Equals xyz	xyz can be a number or a word or date enclosed in single quotes
!= xyz	Not Equals xyz	As above
< xyz	Less than xyz	As above
> xyz	Greater Than xyz	As above
like xyz	Similar to xyz	like 'xyz' may contain the wildcard %
Between x and y	Between x and y	a and b may be numbers, words or dates
in (x,y,z,...)	exists in list	As above
is null	is empty	for example printed_date is null

This method may be used to return two specific invoices on the same query on Invoices Entry form using invoice_num in (1234, 5678) or to return a list of Approved Purchase Orders that have not been printed using (printed_date is null) and (approval_flag = 'Y'). The more complex functions of SQLPlus can also be used. For example, the AND and OR operators can be used to put a number of criteria together into a single query.

Extending the Queries

This method can be extended enabling a 'clever query' in multiple fields using: a, b, c and so on. For example, use the Employees form to query back all employees with the name 'SMITH' or 'KING' and salaries between 1000 and 3000 by entering ': a' in the name field and ': b' in the salary field. Execute the query, then in the query window use: a in ('SMITH', 'KING') and: b between 1000 and 3000.

Record Retrieval

When users query and review multiple records in a multi-row block, they tend to scroll down a record at a time. The cumulative time for scrolling down say six records individually is far greater than pulling the scroll bar down to retrieve the next set of records. A form is usually set to buffer a certain number of rows. Unless a record is still cached, each time a record is scrolled, extra round trips are made across the network, the database has to retrieve and sort rows and the system has to access the associated data block. It is far more efficient to pull the scrollbar down and retrieve the next set of records.

General Protection Faults

Whenever you are faced with a random General Protection Fault problem in Windows 95, you need to obtain the Microsoft TweakUI utility, which is available for most Windows versions. Check the option to 'Log Application Errors To Faultlog.txt' in the Paranoia Screen. This utility will then record the date, time, and other information as shown in the example below:

```
Date 08/06/1999 Time 18:38
NETSCAPE caused an invalid page fault in module KERNEL32.DLL at 014f:bff782ae.
Registers:
EAX=726f2e6b CS=014f EIP=bff782ae EFLGS=00010202
EBX=00001004 SS=0157 ESP=009ff480 EBP=00000a64
ECX=752e3731 DS=0157 ESI=00a63658 FS=107f
EDX=c10e3e20 ES=0157 EDI=01615b70 GS=0000
Bytes at CS: EIP: 89 41 08 8b 56 04 8b 46 08 89 50 04 8d 04 1e 50
Stack dump:
01615b70 00000000 00a63658 00a10000 00000000 00a1000c 0000207a
00000000 00000000 bff78482 00a10000 00a63658 00001004 00000000 009ff554 00000000
```

All you have to do is to record the module, what you were doing, and the date and time so that the events can be correlated. When you log a GPF bug, you will still need the full drwatson log file.

The following two checklists catalogue all known issues that affect and correct GPF problems. The first applies to all versions whilst the second only applies to Oracle Applications Smart Client. You may only need a combination of these, or all of them. They are listed in the order they should be applied with the most likely or easiest listed first.

Generic Checklist

Check the Hard Disk For Errors

Check file and disk integrity using ScanDisk. Cross-linked files will cause a number of problems.

Defragment your hard disk especially if fragmentation is greater than 4%.

Ensure a Clean Boot

Aim to minimize environment settings, interacting drivers, and software in:

```
Config.sys        Minimize device drivers and use 32-bit device drivers wherever
possible.
                        Remove third-party memory managers Remove extraneous and
non-essential settings.
Autoexec.bat      Remove Terminate and Stay Resident Programs.
                        Remove extraneous and non-essential settings.
                        If the TMP and TEMP variables are set, check the directory
exists and there is space on the disk.
Win.ini           Remove programs from the run= and load= lines.
StartUp Folder    Remove all programs.
Registry          Remove the entries under the following keys:

      HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run,
                        'RunOnce', 'RunServices', and 'RunServicesOnce'.
```

Delete Temporary Files

Delete all files in the C:\Windows\Temp folder and directory pointed to by the TEMP variable.

Using 'Find all Files' and delete (or archive) all files named *.tmp and ~.*

Check for Conflicting Devices

In Control Panel | System |Device Manager check that there are no devices with a yellow circle and black exclamation mark. Resolve any other device annotations that and check for duplicate devices.

Check for Duplicate Dynamic Link Library Files

Use Filemon (from www.sysinternals.com) or some other utility to list the libraries loaded into memory.

Check the version of the DLL is the same as the one shipped with the product.

Check DLL files do not have the read-only flag set, which they will have if, copied from CD-ROM.

Apply Windows Patches

Apply *W95oleupd* patch for stability of 16-bit applications.

Apply *Winsock2* (W95ws2setup) for increased stability of communications.

If you experience problems running programs after running the *Winsock2* update, consider applying the *Wsipx update to Winsock 2* and *Windows Library Update.*

If you are not running OSR2, apply the *W95Krnlupd* patch for increased stability of memory.

If using a Cyrix processor:

```
Copy WB16off from the Windows CD to your Windows directory.
 Add the line C:\Windows\Wb16off to your Autoexec.bat file.
```

If using an AMD-K6®-2/350 or above:

```
Apply the Amdk6upd patch.
```

On Windows NT, try reinstalling the service pack.

If running in a Novell environment ensure the latest client patch is installed.

CHECK THE NETWORK

Check for saturation/timeout errors or lost/runt packets:

- **Lost packets**, retransmissions, network storms, and saturation problems are very typical in Novell environments and Service Advertising Protocol (SAP) and printserver packets can flood the network, mostly due to general topology configuration issues. Loosing three sequential packets typically causes a GPF in Oracle Applications. To detect timeouts and lost packets, without a LAN Analyzer, loop the command: ping -l -n 100 <host>
- **Runt packets** are information packets that are an incorrect size or do not follow the standard protocol framing rules. Whenever Applications is running and windows receives a runt-packet it appears to GPF. Nothing has been found yet that will prevent this. A runt-packet can be caused by hacking, power spikes, or perhaps an incorrectly shielded cable.

FIX THE SWAP FILE SIZE: VIRTUAL MEMORY

Firstly, check there is at least 100M free space on the drive hosting the swap file.

Secondly, fix the swap file size at its optimal size: (Physical memory - memory used by windows) *3.

For example, on a PC with 32M, 16M is used on boot so set the swap minimum and maximum to 48M.

Thirdly, some combinations of hardware and I/O controllers do not appear to handle serialization across multiple IDE channels very well. If you only use a single channel, you could try disabling the second channel in the BIOS.

INSTALL THE EXPANDED MEMORY MANAGER

This is thought to increase Windows breakpoints that help when the network quality is less than optimal. Add these lines to the top of the Config.sys file:

```
Device=C:\Windows\Himem.sys
Device=C:\Windows\Emm386.exe NOEMS
Stacks=9,256
```

Note the stacks command is the same as the default Windows environment from *IO.SYS*. If this helps, but does not solve the problem, increase it further to 18,512.

If you suspect a problem with network or video drivers in the Upper Memory Area (640K-1M), exclude it using the command: Device=C:\Windows\Emm386.exe NOEMS X=A000-FFFF. If this resolves the problem and you have checked your video and network documentation, trying reducing the amount of excluded memory using a reduced range such as A000-AFFF.

CHECK VIDEO

Try changing the number of colors from 256 to 64,000 (also known as High Color 16-bit) and vice versa.

Install the latest or tested video drivers checking for known issues.

Disable screen savers and use a plain background/wallpaper.

Reduce Video Acceleration to a minimum to prevent problems with specialized video functions.

Monitor the GDI and USER heap using the Windows resource meter ensuring either resource remains above 50%.

EXCLUDE THE ROM ADDRESS SPACE

Start Windows using Win /D: S. If this fixes the problem, something other than permanently available ROM is using the memory between F000:0000 and 1MB preventing Window from finding a breakpoint. You will need to add *SystemROMBreakPoint=FALSE* to the [386Enh] section of the System.ini file.

Additional Smart Client Checklist

Check The Environment

- Check the FORMS45_PATH points to the latest AU\resource directory.
- Check the oa_config= line of your oaconfig.ora file specifies the appldev directory.
- Check the oracle.ini for duplicate entries.
- *SPECIAL WARNING*: Check your autoexec.bat file does not set oa_config, as this is no longer used.

Disable Global Libraries

Comment the 'FORMS45_APPSLIBS' setting in oracle.ini i.e. add a leading semicolon.

Apply Applications Updates

You can test upgrade your GUI common tools from the certified version 2.3.6.11.0 to a later uncertified version 2.3.6.18. If this resolves the problem, contact your account manager regarding support issues.

Apply FND75WIN.dll of at least version 75.16.1.46 or later from ARU, which will also improve the Windows caching. This only ever used to be deployed as a last resort, long before the full nature of application GPFs was understood. You should eradicate the problem by using one of the previous steps.

All patches and utilities are available from the Microsoft Web site. This document does not cover every known issue but addresses at least 98% of random GPF problems to date.

RevealNet

Apart from the many tuning books and Oracle courses available, one of the tools that provide an invaluable reference is the RevealNet Oracle Administration Knowledge Base. Not only is it useful for tuning, but it is also used by Oracle developers and database administrators. It uses a text search engine and menu system that provide quick access to over 3,400 topics that include insights, explanations, examples, and database scripts. It covers general database tuning and administration, network and SQLNet management, a visual data dictionary and SQL reference. In general, it has proven to be very accurate and reliable. In daily use, it claims to reliably answer 70% to 80% of your questions and comes highly recommended. Product trial versions and discussion groups are available at www.revealnet.com.

Conclusion

An accurate problem definition is key to identifying the source of performance issues and will help focus the tuning effort to achieve the performance targets. The best benchmark figures for the system are obtained by initially tuning the client, then the database and server. Batch jobs and reports should be rescheduled to minimize the affect of on-line users before tuning application SQL access paths or using the tuned client to benchmark the network.

Throughout the tuning exercise, make a change and measure the affect. Once the entire system been checked and tuned, re-benchmark. If targets are still not met, stop and possibly reconsider the targets before recommending a hardware upgrade. Always investigate every area. Look for novel ways to tune such as the way that forms are used and queried or perhaps by grouping jobs, which access the same tables, to run at the same time.

Although tuning is a science, there is a lot of common sense involved.

About the Author

Andy Tremayne is a Senior Principal Technical Specialist working in Applications Performance Group of Oracle. He has worked with applications since 1992, holds an honors degree in computing and infomatics, and is a qualified electrical and electronic engineer. Andy has been involved in around 200 customer engagements in the past 2 years. The methodology section was co-developed with Jim Viscusi of the CoE.