Project MEGAGRID:
Capacity Planning For Large Commodity Clusters

An Oracle, Dell, EMC, Intel Joint White Paper
December 2004
# Project MEGAGRID:
Capacity Planning for Large Commodity Clusters

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>4</td>
</tr>
<tr>
<td>GRID Infrastructure</td>
<td>6</td>
</tr>
<tr>
<td>Services</td>
<td>7</td>
</tr>
<tr>
<td>Simplification of Capacity Planning and Management</td>
<td>8</td>
</tr>
<tr>
<td>General Recommendations Defining Performance, Resource Consumption and Capacity in a Database Cluster</td>
<td>9</td>
</tr>
<tr>
<td>Description of the Application Workload</td>
<td>10</td>
</tr>
<tr>
<td>Capacity Planning and Management Based on Services</td>
<td>11</td>
</tr>
<tr>
<td>Automating Service Level Objective Monitoring</td>
<td>13</td>
</tr>
<tr>
<td>Defining Services, Modules, Actions and Thresholds</td>
<td>13</td>
</tr>
<tr>
<td>Determining Causes for Service Violations</td>
<td>15</td>
</tr>
<tr>
<td>The Importance of SERVICE, MODULE and ACTION in Large Database Clusters</td>
<td>16</td>
</tr>
<tr>
<td>Performance Statistics and Metrics</td>
<td>17</td>
</tr>
<tr>
<td>CPU Consumption</td>
<td>18</td>
</tr>
<tr>
<td>Service Time</td>
<td>19</td>
</tr>
<tr>
<td>I/O Requirements</td>
<td>20</td>
</tr>
<tr>
<td>Memory Consumption</td>
<td>21</td>
</tr>
<tr>
<td>Network Throughput and Latency</td>
<td>22</td>
</tr>
<tr>
<td>Data Referencing Behavior</td>
<td>22</td>
</tr>
<tr>
<td>CPU Capacity Estimates for Interconnect Traffic</td>
<td>24</td>
</tr>
<tr>
<td>Cdue co</td>
<td>24</td>
</tr>
<tr>
<td>Estimating the CPU Impact of Global Cache Coherence</td>
<td>24</td>
</tr>
<tr>
<td>Server System Configuration and Sizing</td>
<td>25</td>
</tr>
<tr>
<td>Server Options</td>
<td>26</td>
</tr>
<tr>
<td>Memory Addressability</td>
<td>26</td>
</tr>
<tr>
<td>Number of I/O Slots</td>
<td>27</td>
</tr>
<tr>
<td>Network Adapters and Frame Sizes</td>
<td>28</td>
</tr>
<tr>
<td>Networked Storage Capacity Planning</td>
<td>28</td>
</tr>
<tr>
<td>Storage Capacity</td>
<td>28</td>
</tr>
<tr>
<td>Application and Database Size</td>
<td>29</td>
</tr>
<tr>
<td>Backup Requirements</td>
<td>30</td>
</tr>
<tr>
<td>Storage System Requirements Summary</td>
<td>31</td>
</tr>
<tr>
<td>Storage Configuration Decisions for the MegaGrid</td>
<td>32</td>
</tr>
</tbody>
</table>
Additional Sizing Considerations for Adding Cluster Nodes
Storage Infrastructure Issues
Bandwidth
Connectivity
Network Design and Port Density
Port Density for the MegaGrid
Tiered Storage and the Storage Class
Virtualize and Automate Storage Management
Effective Cluster Capacity and Performance
Capacity Management by Expanding and Shrinking of Services
Conclusion
Appendix A – RAID Level Comparison
INTRODUCTION

In order to successfully plan and manage capacity and growth in an Enterprise Grid Computing environment which provides high quality database service to applications and utilizes resources such as servers and disk optimally, a flexible and scalable server, storage and network infrastructure must be built. All components need to be arranged in a topology in which their resources can be provisioned easily to applications needing them to achieve their goals.

In Project MegaGrid, Oracle, DELL, EMC, and Intel are working jointly to design solutions, test and document best practices for Enterprise Grid Computing. For this purpose, a development and test system was constructed in the Oracle Global IT Data Center in Austin, Texas.

At the core, a Grid is an interconnected structure of components providing application services and integrating a variety of resources like databases, application and database servers, storage systems and networks. It is a prime objective to implement the integration in such a way that the resources can be managed as a single system image and provisioned on demand based on the service requirements of each application.

The MegaGrid project environment is composed of standard high-volume servers such as the Dell PowerEdge 1750 or 1850 servers using Xeon processors, Dell 7250s using Itanium 2, SAN and NAS storage components such as EMC Symmetrix DMX series, CLARiiON CX series, and Celera Network Sever series, Oracle Database 10g RAC and Automatic Storage Management (ASM), ORACLE Application Server 10g clusters and management components such as Enterprise Manager Grid Control and EMC Control Center. The following two different clusters were built:

- 24-node cluster with Dell PowerEdge 1750s and
• 12-node cluster with Dell 7250s.

All resources are physically connected in the form of a large server and storage network. Logically, the machines function as a pool of application servers running Apache or ORACLE Application Server 10g, and database servers clustered with ORACLE Database 10g RAC. The storage network provides access to shared file system storage for the application and database server software and data storage. The storage used by the different databases is managed by a single ASM cluster, which maintains disk and recovery groups for each database.

The MegaGrid therefore

• Can increases the total computing power available to applications
• Exploits low-cost, high-performance technologies as building blocks in a flexible infrastructure
• Benefits from the easy deployment and integration of different types of storage systems
• Can grow the total computing power and storage infrastructure in small increments
• Provides on-demand adjustments of computing and storage resources based on the short-term or long-term needs of applications
• Allows easy and flexible migration and movement of data between clusters, databases or storage classes
• Covers availability objectives for planned and unplanned outages by rebalancing workloads
• Can be managed using a single set of controls.

It therefore provides business opportunities in terms of cost-reduction, server and management consolidation, and efficient use of available CPU, storage and networking bandwidth.

In this whitepaper we provide some guidance in estimating capacity and performance impact in an end-to-end Grid Computing Environment. In Phase I of the project, the focus lies on the database service. The paper therefore refers primarily to the database tier.

The examples used in the paper are based on the test and capacity data from a real application used in the telecommunications industry and provided by Cramer Systems in the UK. Previously, performance and scalability tests with this application were performed on a 72 CPU SMP machine. The goal was to achieve 550000 business transactions per hour for a transaction mix defined by
a large telecommunications provide in the UK at an average response time of less than 1 second.

In Phase 1 of the Project MegaGrid, the goal was to use the exact application in the Enterprise Grid composed of the two clusters and reach or surpass the service requirements achieved in the tests on a large single machine without little or no changes to the application.

GRID INFRASTRUCTURE

Conceptually, applications implement business functions by using services provided by an arrangement of components such as clusters of application and database servers, disk storage systems and networks. The basic concept of a Grid is to provide these services. To achieve this goal it must implement

- Universal connectivity and resource sharing;
- Virtualization of resources, such as software components, storage types and server models;
- Clustering; and
- Easy management of complexity via self-tuning, automatic monitoring and advisories

so that services can respond flexibly to changing resource demands, allocate sufficient capacity, and maintain high availability and good quality of service.

The following diagram illustrates this concept by showing several applications requesting services, which a Grid environment provides. Groups or clusters of application servers are providing each of the services. Some of the application servers are providing more than one application service. The application servers may depend on a database service. The database kernel virtualizes the corresponding services inside the database. This makes a correlation between the business functions and the database operations possible. To have the flexibility to react to changes in load, the database grid has spare nodes available, which can be assigned to database services as needed.

The database service is provisioning storage from ASM as needed. The ASM Grid virtualizes the different types of storage devices that are available by presenting them as disk groups and recovery groups. Depending on the current resource requirements different storage types can be assigned.
SERVICES

With Oracle Database 10g, Services provide the ability to define workloads that applications use in an Enterprise Grid. Each Service represents an application or part of it and workload attributes, such as service level thresholds and priority. Services enable the workloads to be measured and managed to efficiently deliver the required capacity on demand. Having the ability to deploy and re-deploy different Services based on business polices allows businesses to run smoothly and efficiently. The Database Resource Manager allocates resources to these Services depending on their priority.

From the point of view of performance and capacity management, services

- Align a business model with lower-level, physical implementation, and are therefore abstractions visible in the software layers of the application server and database tiers
• Represent workloads with similar resource demands or performance requirements
• Make applications opaque to the physical implementation of a service, i.e. how many cluster nodes are used and which type of server is deployed
• “Insulate” performance-critical tasks and services from the effects of “perturbation” caused by other workloads executing on the same machines by rebalancing to less utilized or spare nodes
• Constitute an efficient approach for performance management and monitoring.

Assigning business functions to services with clear objectives for performance and availability, and subsequently mapping them to a server and storage Grid is central to capacity management in large clusters. The decision as to the physical implementation of a service: i.e., the type and number of servers and the type of storage from the pool to use, must be made according to the service objectives and estimated or measured demand.

SIMPLIFICATION OF CAPACITY PLANNING AND MANAGEMENT

In general terms, capacity planning and capacity management aim at avoiding system saturation and violation of response times limits, throughput requirements, and the reduction of availability.

A Grid-computing environment combining low-cost and high-performance servers, a network of storage systems, and the software functionality to add and remove server and disk capacity as well as rebalance workloads and disk access to optimize the usage of resources constitutes the basis for a simple and immediate approach to capacity planning.

Owing to the low-cost of the system components as well as their easy procurement, it becomes possible to add bandwidth on-demand or incrementally, using spare servers and disks or by de-provisioning under-utilized or only temporarily utilized resources from other application in the server and storage Grid.

A global, service-oriented view constantly monitors the performance goals and provides indications of capacity saturation. A flexible infrastructure allows for rapid adjustment to changes of usage patterns by workload balancing or rearranging the configuration of services to make use of faster disks, more spindles, more CPU power or more cache memory. There is even more tolerance for errors in initial planning due to insufficient or incomplete data.

In summary, the reality of easy provisioning of bandwidth to an application requiring higher capacity to fulfill its goals, in time and at relatively low cost, simplifies capacity planning and management. Given the fact that the Grid is
built with the idea of rapid connectivity, extensibility and reconfiguration of components in mind and services are fully implemented to achieve abstraction from hardware and location transparency, there is always a server with more CPU power, cache and a storage unit with more disks that can be “switched” from one section of the Grid to another, or simply added and activated as a new element.

Generally, capacity estimations are required when

- Applications are “downsized” from a powerful server with a large number of CPUs and many Gigabytes of memory supporting a large number of users
- A new application is developed
- Applications are consolidated

In all cases, there will be more or less complete operational data to facilitate an accurate estimation. For the consolidation of existing applications, technology trends and the lack of uniformity among systems from different hardware vendors may pose difficulties in making predictions as to how a given application will perform. The detailed characterization of the workload can vary from one environment to the next. Most notable, the distribution of an application or workload over multiple machines in a cluster incurs synchronization cost with a certain impact on CPU consumption and network latency, the magnitude of which is entirely dependent on data access patterns and is difficult to predict with great accuracy.

However, due primarily to the arrangement of all components in an Enterprise Grid structure allowing resource provisioning on-demand, the level of abstraction given by the software in terms of service placement, and the relatively low-cost of the components, the risk of miscalculation can be mitigated and response to critical shortage is faster and more flexible.

**General Recommendations Defining Performance, Resource Consumption and Capacity in a Database Cluster**

Generally, the following recommendations are important:

- Maintain a repository of Statspack and AWR statistics from legacy systems: i.e., export or back up existing statistics to highly available file systems or ATA storage,
- Identify the business functions and operations for the application
- Produce a clear definition of an application’s performance requirement, based on business functions or operational data and the service levels that need to be achieved, described in terms of the important metrics and units of work, such as the CPU consumption, response time and I/O demand per user, session or transaction,
• Map the business functions and operations to services and modules
• Instrument applications to use SERVICE names and identify portions of the user code by registering MODULE and ACTION
• Describe the workload classes based on their resource usage patterns (e.g., CPU-intensive, high I/O content), or data access characteristics (e.g., volume and rate of inserting new data, reporting queries using unique or sequential scans, etc.)
• Characterize the I/O workload based on access patterns (e.g. random or sequential reads or writes) and media recovery requirements and assign specific storage to these usage classes, supported by standard sizing of LUNs
• Estimate the cost of the Global Cache based on best and worst case scenarios
• “Overprovision” the Grid with spare resources such as servers for application and database instances, disk storage, IP addresses, ports, cables etc

It is important to realize that ORACLE Database 10g, ASM and the storage management software provided by EMC allow for adaptive improvement by rebalancing of data distribution on disks as well as transparent re-distribution of user connections. The clear structuring of application services and the exploitation of service-related monitoring and notification support the concept of resource-on-demand by differentiation.

The following practice-oriented approach to capacity planning follows a top-down approach from the definition of application and business requirements and their service requirements in terms of response times, throughout and availability, to the selection of components that comprise attributes matching the required service demand, and their grouping.

Description of the Application Workload

For illustration, a real-world application model will be used:

• A network provisioning system by Cramer Systems in the UK

In the introduction it was mentioned that the application tested in the Phase I of the MegaGrid was chose for various reasons, namely because

• It includes a real-word transaction mix
• There were baseline data, OS statistics and Statspacks from a previous test on a large single SMP machine for comparison and planning
• The workload generator does not consume much resource, is easy to manage and produces repeatable results
• The size of the database can be scaled up easily.

At a high level, the application workload is described as follows:

• HTTP requests are submitted to an HTTP server
• The business logic executes mainly in the database server and uses PL/SQL and XML.
• The transaction mix consists of short queries (< 30 ms response time), more complex queries (< 13 secs response time) and DML operations referencing small numbers of rows
• The CPU consumption and I/O volume is high and most of the work is done in the database rather than the application server
• The application servers are collocated with the database server.
• The service requirements were defined in terms of throughput, response time and availability, namely 550000 business transactions per hour at a response time of < 1 second with and availability of 24 x 7

CAPACITY PLANNING AND MANAGEMENT BASED ON SERVICES
A top-down approach to implementing services in a server and storage grid would start with the applications to be consolidated, their business functions, service requirements and other characterizing attributes:

• List the business functions and transactions, frequency of execution, their type and importance
• Define their requirements for performance in terms of response time, transactions, number of users or any other significant work unit
• Group the business functions and transactions based on similar resource usage, similar SLAs for performance or availability or similar business logic
• Create services for these “workload classes”
• Within each service, determine MODULES and ACTIONS for better monitoring of service quality and resource usage and enable monitoring for important modules
• Estimate the required capacity to achieve the desired goals
• Assign servers and storage
The last item refers to the assignment of services to application and database servers. A critical service may want to be deployed on a different set of nodes than other services to avoid “perturbation;” batch or database maintenance jobs could be assigned to powerful nodes from which online services have been migrated away.

It is strongly recommended to deploy the SERVICE, MODULE and ACTION functionality provided by ORACLE Database 10g and enable statistics gathering for important modules and actions. The monitoring of important service statistics and metrics facilitates early detection of capacity limits, which could be easily resolved by redistribution of resources in the Grid.

In the following table, two applications are represented. Note that the structure of the table is entirely dependent on the definition and breakdown of the business functions. The table lists the weight or frequency of the business function, the unit of work, the growth in work units and the processing mode:

<table>
<thead>
<tr>
<th>Application</th>
<th>Business Function</th>
<th>Pct</th>
<th>Unit of Work</th>
<th>Growth per year</th>
<th>Schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMS Inventory</td>
<td>Route Report</td>
<td>46%</td>
<td>Trans/h</td>
<td>50%</td>
<td>24h/7d</td>
</tr>
<tr>
<td>BMS Inventory</td>
<td>Status Change</td>
<td>16%</td>
<td>Trans/h</td>
<td>24%</td>
<td>24h/7d</td>
</tr>
<tr>
<td>BMS Inventory</td>
<td>Node query</td>
<td>12%</td>
<td>Trans/h</td>
<td>25%</td>
<td>24h/7d</td>
</tr>
<tr>
<td>BMS Inventory</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>ERP Accounts</td>
<td>Accounts/h</td>
<td></td>
<td>Accounts/h</td>
<td></td>
<td>Monthly</td>
</tr>
<tr>
<td>ERP Receivable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERP Self Service</td>
<td>User</td>
<td></td>
<td>User</td>
<td></td>
<td>24h/7d</td>
</tr>
<tr>
<td>ERP</td>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Example of Applications, Business Functions and Load Profile

An application defines work in terms of users, transactions, requests, jobs or other significant units. Here the percentage is defined as an estimate of the units of work executed as part of the total application and the growth is also given as the percentage growth for units of work; i.e., an increase in the number of users, the number of transactions or requests to be processed per unit of time.

It is possible to define a service for each business function listed, but it may be expedient to wait until detailed characteristics have been provided.

The focus of capacity planning may fall on the Route Report function, as it not only constitutes the largest workload component of the Inventory application, but also accounts for the biggest growth in terms of transaction volume. All business transactions are online; consequently the service quality depends on the responsiveness of the system. It is also clear that the Inventory business is
supposed to be available for 24 hours a day and 7 days a week and that a highly available solution is therefore indicated.

The goal of the Payroll batch job is apparently to execute with high throughput rates within a defined period of time for month-end processing. The service it requires will probably have to provision CPU and I/O bandwidth. It should also be considered that the resource-intensive execution of this business function might affect other applications executing in the same environment. These considerations may make it expedient to provision more powerful machines to Payroll once monthly, and perhaps isolate the jobs to a subset of the entire topology in order to minimize the impact on other applications or business functions within the same application. A nightly, off-peak schedule may be indicated so that resources can be moved from low-intensity workloads to the demanding and schedule-driven Accounts receivable job.

On the basis of the simple table of business functions, some assumptions have been made regarding service goals, significance and priorities. For more accurate planning, it would be logical to detail the objectives and priority for each function so that they can be translated into thresholds, alerts and monitoring units supported by ORACLE Database 10g.

**Automating Service Level Objective Monitoring**

The capacity of a system reaches saturation if any of the service levels is violated. In ORACLE Database 10g, the manageability framework monitors elapsed time and CPU time consumed by a service on each instance and will alert when thresholds are defined.

**Defining Services, Modules, Actions and Thresholds**

When the workload has been categorized in terms of business functions, its service requirements should be stated. The major metrics such as maximum response time or transaction throughput should be specified for each important transaction. In addition to that, the time to handle failures and the mean time to recover ought to be stated.

It is then possible to group business functions with similar performance and availability requirements into classes. These classes then represent components of the workload. Depending on the homogeneity, these functions may then be given a service name and can be further subdivided into modules and actions for accurate accounting and monitoring.

The subsequent mapping of services to nodes should take the specific objectives into account so that servers and storage with sufficient capacity to maintain the service levels can be provisioned. However, for the physical implementation and the estimated capacity of a service, additional data should be procured.
The table below specifies the response time and throughput thresholds for all business functions. Let us assume that the relevant metrics of the entire workload for “BMS Inventory” are the average response time and the aggregate throughput. Moreover, the requirements for availability are the same for all functions.

<table>
<thead>
<tr>
<th>Application</th>
<th>Business Function</th>
<th>Resp. Time</th>
<th>Trans/h</th>
<th>Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMS Inventory</td>
<td>Route Report</td>
<td>850ms</td>
<td>254840</td>
<td>N/A</td>
</tr>
<tr>
<td>BMS Inventory</td>
<td>Status Change</td>
<td>1100ms</td>
<td>88640</td>
<td>N/A</td>
</tr>
<tr>
<td>BMS Inventory</td>
<td>Node query</td>
<td>65ms</td>
<td>66480</td>
<td>N/A</td>
</tr>
<tr>
<td>BMS Inventory</td>
<td>Summary</td>
<td>780ms avg</td>
<td>554000</td>
<td>N/A</td>
</tr>
<tr>
<td>ERP Accounts</td>
<td>Accounts</td>
<td>8 hours</td>
<td>10000</td>
<td>N/A</td>
</tr>
<tr>
<td>ERP Self Service</td>
<td>&lt; 4 secs</td>
<td>TBD</td>
<td>3000</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Application and Business Function Service Level Requirements

In this example a service will be created for the application “BMS Inventory” and its business transactions defined as modules. Assigning module names to the business transactions and setting the module/action info by calling the DBMS_APPLICATION_INFO.SET_MODULE() procedure has the advantage of being able to monitor each component of the Inventory service separately:

execute DBMS_APPLICATION_INFO.SET_MODULE('Route Report', 'Display Route');

The module and action name would appear in dynamic views related to sessions, statistics and SQL and can be queried ad-hoc or via Enterprise Manager.

In order to guarantee capacity for sub-second average response times, it is a good practice to monitor each module in order to find out which one degrades under load and may cause the threshold for the service to be violated.

A threshold can be defined for the service, which uses the average response time requirement stated in the table above as a critical value to implement monitoring of response time violations by the database:

execute DBMS_SERVER_ALERT.SET_THRESHOLD(-
The system will now warn when it does not have sufficient capacity to maintain average response time for the Inventory service.

In this particular case a threshold alert is set based on an end-to-end response time metric, but the manageability framework will actually monitor the elapsed time spent in the database kernel. However, by tracing individual requests in the server, it was established that most of the end-user response time is spent there rather than in-flight or at an application server. Consequently, setting the threshold based on server elapsed time will be reasonably accurate.

The alert will warn if

- Database response time for the service exceeds the threshold due to increases in the workload volume
- Database response time for the service exceeds the threshold due to reduce capacity when a node fails

In both cases, the manageability framework will indicate that there are insufficient resources to meet the objectives.

**Determining Causes for Service Violations**

If the SLA for a service includes a response time goal and the goal is to minimize the average business transaction response time associated with its service, then it is useful to determine violating components and the reason why violations occur. If a service is instrumented to register module and action, the work done or wait time incurred can be aggregated by them. The monitoring is enabled either via Enterprise Manager or by calling a DBMS package API; e.g.,

```sql
execute DBMS_MONITOR.SERV_MOD_ACT_STATS_ENABLE(
    SERVICE_NAME => 'BMS Inventory',
    MODULE_NAME => 'Route Report',
    ACTION_NAME => null);
```

This enables statistics collection for the business transaction 'Route Report' which was declared as a module by the application.
The performance statistics can be queried in the view

\texttt{V$SERV\_MOD\_ACT\_STATS}

or in the corresponding screens offered by Enterprise Manager.

A service can also be monitored in terms of the wait time for a particular class of events in the database:

<table>
<thead>
<tr>
<th>SERVICE_NAME</th>
<th>WAIT_CLASS</th>
<th>TIME_WAITED/secs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complex Routing</td>
<td>User I/O</td>
<td>837</td>
</tr>
<tr>
<td>Complex Routing</td>
<td>Cluster</td>
<td>437</td>
</tr>
</tbody>
</table>

\textbf{Table 3: Example of Wait times by Service}

by querying the view \texttt{V$SERVICE\_WAIT\_CLASS}, which in this case shows us significant time accumulated by waiting for I/O and due to global cache operations.

These classes of events can be narrowed down further by querying the service event statistics in \texttt{V$SERVICE\_EVENTS} for a more detailed profile of where time is spent.

The same profiles can be obtained in the appropriate screens of Enterprise Manager.

Note that characterizing a business function or transaction within a service by module and action gives the opportunity for accurate drill-down of the work done on behalf of a transaction, and its cost.

The Automated Workload Repository in ORACLE Database 10g maintains a history of detailed snapshots of many statistics and metrics for objects, and SQL, synchronized on all instances of a cluster. An ADDM gives advice about performance tuning based on the snapshots in the repository and can identify the major bottlenecks and contention.

Ultimately, the violation of service level threshold can be traced to its bottleneck. The advisory framework in ORACLE Database 10g will attempt to self-tune the system to adjust the capacity based on demand, or will provide tuning recommendations.

Any service time violation may be due to a problem to which tuning can be applied. However, many performance problems develop over time, with increasing load and may have a capacity issue as the root cause.

\textbf{The Importance of SERVICE, MODULE and ACTION in Large Database Clusters}

In the previous sections, we have been trying to show that working with the concepts of SERVICE, MODULE and ACTION in ORACLE Database 10g is
not only an important virtualization technique, but also enables automatic features such as service time thresholds, alerts and aggregation methods which facilitate performance diagnostics, and eventually simplify capacity planning and management.

Planning is facilitated via trending and forecasting by service based on automatically collected statistics, while management is driven by alerts and threshold violations, which can be fixed in the short-term by load-balancing.

Moreover, services, modules and actions associate database workloads with applications and, ultimately, business logic.

The monitoring features built into these concepts are visualized and managed by ORACLE Enterprise Manager for all nodes in a cluster, but are also available to ad-hoc queries and other monitoring tools.

It is therefore strongly recommended to use the features in applications, which are intended to share the resources provided by a large cluster or Grid system.

PERFORMANCE STATISTICS AND METRICS

Making accurate predictions about the capacity in the database server tier required to support the SLAs for an application not an easy task. There are four main reasons for large errors in estimations:

- There are few or no operational statistics from a production or test system available,
- The baseline statistics were collected on a legacy system with a different architecture and there are no comparative benchmarks or workload independent capacity measures,
- The contribution of different components of the workload to the overall system performance is not sufficiently described,
- The impact of remote, global cache memory references cannot be estimated for lack of traces and understanding of their impact.

In ORACLE 9i, customers were encouraged to take snapshots of instance performance and load data on a regular basis as scheduled jobs and archive the tables in which the data is stored.

In ORACLE Database 10g, the Automatic Workload Repository is maintained by a background process called MMON which snapshots the instance statistics in regular intervals.

Using the collected instance statistics to describe a workload for a single instance is a good start when downsizing to a grid model with low-cost, high-
performance servers. The challenge lies in distributing work done on a large, single computer, over multiple servers sharing data.

The following system or component level statistics and derived metrics can be used in the estimation of capacity. Generally, the statistics will be normalized over business transactions.

**CPU Consumption**

The database server CPU consumption of an application is significant to determine which types of processors and architecture to use and how many CPUs each machine should have. It will also determine what fraction of work can be performed by one node in the cluster and therefore how many of these processing units must be combined to support the scale of the application.

In order to achieve this, one should compute the following metrics:

- The total CPU power used at a steady state per unit of time
- The unit-of-work CPU consumption per unit of time (e.g., CPU sec/transaction, CPU cycles/transaction etc.)

If these metrics are not directly available from external Performance Tools, they can be approximated by

- CPU time statistics in Statspack or AWR for an instance and/or SQL execution

or converted from utilization and CPU clock rate.

For the Inventory system, the average business transaction can be used as a unit of work and the CPU consumption estimated by the number of CPUs, the clock rate, and the peak utilization of the system and then converted in the following way:

- Total CPU consumed/sec = #CPUS * CPU clock rate * pct utilization = 72 * 1.2Ghz * 0.91 = 78.624Ghz
- (Total CPU consumed/sec) / (unit of work/sec) = 73.44 / ((554000 transactions/hour)/360) = 0.5109 GHz/transaction or 0.4258 secs/tx

<table>
<thead>
<tr>
<th>Application</th>
<th>CPU% idle</th>
<th>#CPUs</th>
<th>Clock rate</th>
<th>Trans/h</th>
<th>CPU/Trans</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMS Inventory</td>
<td>9</td>
<td>72</td>
<td>1.2Ghz</td>
<td>554000</td>
<td>0.5109Ghz</td>
</tr>
</tbody>
</table>

*Table 4: CPU Requirements per Business Transaction*

Note that the conversion from CPU cycles per transaction to wall-clock time would result in 0.426 secs of CPU.
Obviously, the aggregated CPU power of the cluster must support the transaction rate.

**Service Time**
The database service time is the total time spent executing in the database, processing or waiting for a resource, and thus includes

- Service Time = CPU Time + Wait Time

and can therefore be estimated from the event wait time reported by AWR. For the Inventory system, the top wait events reported by the baseline data as the following chart describes:

Figure 2: Service Time per Transaction

Therefore, the total estimated service time per business transaction is

- \( \frac{\text{Total wait time/sec}}{\text{Transactions/sec}} = 0.8356 \text{ sec} \)

In summary, based on the database statistics, the estimated times for a business transaction are as follows:

<table>
<thead>
<tr>
<th>DB Service Time</th>
<th>DB CPU Time</th>
<th>Total CPU Time</th>
<th>DB IO wait time</th>
</tr>
</thead>
<tbody>
<tr>
<td>836 ms</td>
<td>323 ms</td>
<td>426 ms</td>
<td>390 ms</td>
</tr>
</tbody>
</table>

Table 5: Estimated Service Time for a Business Transaction

Note that the CPU time estimated for the entire system above is higher than the DB CPU time, by approximately 103 ms. This can be explained by the fact that the Web Server used by the application is located on the same machine as the
database instance, and that the difference would have to be added to the DB service time, so that the total estimated response time for the system per transaction would be

- 939 ms

Please note that this is an approximation. It will be useful when evaluating the performance of the new system. It also gives us a baseline and an upper bound for the application to normalize the end-user response time.

The derived average server response time per business transaction for the Inventory system was based on observed operating system metrics and ORACLE statistics from a 9i system. It includes the service time for the database and the web application server. The metrics can later be used to estimate the server capacity to be provisioned in the server grid.

Newer processor technology such as Xeon and Itanium 2 are likely to reduce the CPU cycles due to efficient branch prediction, pipelining and large caches whereas the estimates above are derived from performance data gathered on a different architecture. The benefit of migrating to standard, high-volume servers is likely to have a higher impact on CPU efficiency and the CPU time consumed for each transaction.

I/O Requirements

In order to size the I/O capacity of a cluster of low-cost, high-performance servers and to determine the number of nodes required to support a given I/O load and the choice of storage systems, the usual metrics are

- I/O rates in operations per sec
- I/O throughput in MB/sec
- I/O sizes for reads and writes
- Average disk service times
- Redo write rate and size

The redo write rate and size of the logs are used to derive the volume of redo that would have to be archived in the event that archiving to a recovery area was required. If the redo size and the log switch interval are known and the required I/O throughout has been derived, then the throughput of the disk system used for a recovery area needs to be configured to sustain the sequential I/O demand so that archival is finished before the next log switch.

Here are the I/O requirements for the Inventory production system, measured at the host operating system level of a 72 CPU SMP machine. These are operational statistics:
Table 6: I/O Baseline Statistics from the Operations History

Most of the disk I/O is random, single block I/O, since the size computed from the statistics above approaches the database block size (8K).

The shared disk storage system would have to sustain an I/O volume of about 5700 random IOPS of about 8K-block size at an average service time of 6ms. As we have seen above, the IO wait time constitutes a considerable amount of the service time in the database, so that it is of some importance to provision a storage system and layout, which can minimize the IO service time. Note that the 6ms service time for an I/O is averaged over read and write operations and that the distribution of controller cache hits and misses is not known.

The aggregate HBA and I/O bus capacity of the nodes in the cluster must – for this system - allow a throughput of 50MB per sec with added peaks when archival starts.

Assuming that media recovery is a priority and archiving should be enabled, we can calculate the redo log switch rate and log write I/O rate from the redo size per second in the database statistics, so that

- Log file size / redo size per sec = 500MB / 956 KB = 523 secs or approx 1 log switch every 8 mins
- About 1MB/sec of log writes
- 500MB of redo log must be archived within less than 8 minutes

The sizing of the recovery area and the choice of storage should therefore provide the capacity to sustain the service required by the application.

Memory Consumption

The total buffer cache and process memory will be distributed over multiple, smaller machines with 4 – 8 GB of main memory. This may be a limitation on a 32-bit platform and therefore the SGA and process memory consumption should be taken seriously. If ORACLE statistics are available, focus on the

- PGA advisories
- SGA and buffer cache sizing and advisories

to obtain actual usage hints from a production or test systems.

For the Inventory system, the memory sizes are listed below:
<table>
<thead>
<tr>
<th>BMS Inventory</th>
<th>90GB</th>
<th>6GB</th>
<th>31GB</th>
<th>53GB</th>
</tr>
</thead>
</table>

**Table 7: Memory Consumption from Operations History**

Note that although there is some additional cost to maintain global memory structures, the cost is not significant to be taken into account for our estimates. However, when inter-node Parallel Query is used, one should follow the documentation to size the shared pool.

**Network Throughput and Latency**

Usually, there are no interconnect statistics available when trying to plan the capacity of the private networks in large database clusters. As a rule,

- One GB Ethernet NIC can support up to 800Mb/sec or 12200 8K blocks of effective capacity and is usually sufficient for OLTP large size clusters,
- The latencies for remote block accesses, measured in roundtrip times, are usually in the range of a few hundred microseconds, but may increase even before bandwidth saturation by a factor of 3 due to
  - System load and increased scheduling latencies
  - Faulty network configurations and buffer overflows
  - Block contention.

and runtime configuration and load balancing should ensure that run queues remain short, socket receive buffers are sized to accommodate at least 256KB of data and large frame sizes (Jumbo frames) are supported and configured,

- For DSS or DW applications using PQ, the bandwidth may have to be increased by adding NICs

The fact that multiple clusters or database are sharing the same physical interconnect should be taken into account. However, it is recommended to configure different physical interconnects or VLANs for each cluster managed in the server Grid.

**Data Referencing Behavior**

Data referencing patterns and distributions can be an important issue for globally shared data. While a single system with large memory support may cache the entire working set of an application, the same cache may have to be distributed over multiple machines in a commodity cluster and data may have less locality because it is shared by multiple machines and must therefore be globally synchronized.
This involves interconnect messaging and therefore CPU and latency overhead for remote cache reads, even if these are only in the range of a few hundred microseconds to a millisecond.

In some special cases, write contention on a small working set of data blocks can result in significant additions to the elapsed time of transactions.

In general, it is hard to determine data access patterns with tracing. The interesting categories are

- Randomly distributed uniform read access: if it is prevalent, it is unlikely to be problematic in terms of cache locality because the cluster systems will behave very much like single instance systems; if the working set is large and does not fit into a single cache, then disk I/O will be the prevalent form of I/O; if the working set is small, then the instances will not exchange a lot of message

- Randomly distributed uniform write access: if it is prevalent, there can be a considerable amount of remote cache referencing depending on the probability with which blocks are accessed, and the size of the data set; as a rule, it will rarely cause any severe contention

- Frequent write access to a small set of blocks, aka “hot” blocks, may cause contention when the cluster becomes larger and transactions may incur some delay because the cache reads must wait for data from a remote instance; the contention for the small working blocks reduce locality of access when referenced by processes distributed over the cluster

In general, data referencing behavior is very hard to predict on the basis of regular statistics collection.

However, the workload repository maintained by the database collects statistics for cursors. With a bit of effort, the cursors can be matched to business transactions, especially when services are used and applications have been instrumented with MODULE and ACTION. Any cursors for insert statements consuming a lot of time, are executed frequently and thus constitute a large part of a workload with critical service levels should be examined carefully for the possibility of a small working set.

In the case of the Inventory application, the application designers specified that the database read and write access be distributed mostly randomly and uniformly over the entire database.
CPU CAPACITY ESTIMATES FOR INTERCONNECT TRAFFIC

In ORACLE Database 10G RAC, a global cache of shared data is maintained coherently by the so-called Cache Fusion protocol.

Consequently, some memory references may be non-local due to contention or a large shared working set and database blocks are read through the private interconnect via memory-to-memory copies.

Additionally, every block read from disk into a local buffer cache must be globally managed; i.e., the Global Cache Service must know in which instance the block is cached.

Memory transfers and global cache coherence for disk reads have a cost in terms of CPU cycles and latency. The cost for each operation is known, but can be different on each platform. Therefore it is difficult to provide all the reference data.

Moreover, the volume and rate of operations is difficult to estimate in advance because it is specific to the data accesses of each particular application.

Therefore, one needs to rely on estimations. For capacity planning, we model the CPU cost by adding a fractional cost to the metric for CPU per transaction derived above.

Estimating the CPU Impact of Global Cache Coherence

The impact of the global cache processing is not known in advance. Therefore,

- Scenarios of 5, 10 and 20% CPU overhead per transaction are modeled as best and worst case scenarios.

In order to estimate the cost, we simply

- Adjust the CPU per transaction by an added, assumed percentage increase and then
- Compute the transaction rate with the adjusted metric for different system utilizations.

Then the number of nodes required to support the target transaction rate is calculated by

- Dividing the target transaction rate by the adjusted transaction rate.

For example, assuming an added cost of 5% of CPU per transaction, the adjusted transaction rate at 75% node utilization for the Dell PowerEdge 1750 is 9 per second per node. The target throughput is 544000 transactions per hour.
or 153.9 per second divided by the adjusted transaction rate per node at 75%, 
153.9 / 9 = 17.1, or 17 nodes.

In summary, in order to support a transaction rate of 540000 per hour at 75% 
utilization per node and 5% overhead per transaction, 17 clustered nodes of 
Dell PowerEdge 1750 servers are required.

The following table summarizes the results for the different cost models:

<table>
<thead>
<tr>
<th>CPU cost per Transaction</th>
<th>CPU/Transs</th>
<th>Dell 1750 Trans/sec @ 75% utilized</th>
<th>Dell 1750 Trans/sec @ 100% utilized</th>
<th>Dell 7250 Trans/sec @ 75% utilized</th>
<th>Dell 7250 Trans/sec @ 100% utilized</th>
</tr>
</thead>
<tbody>
<tr>
<td>5%</td>
<td>0.5364</td>
<td>17</td>
<td>13</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>10%</td>
<td>0.5619</td>
<td>17</td>
<td>14</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>20%</td>
<td>0.6131</td>
<td>19</td>
<td>15</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 8: Comparison of Different Cost Models

It should be noted that despite the fact that we calculate for 100% utilization of 
the nodes it is recommended to always

- Allow for at least 25% headroom for CPU capacity on average 
and accommodate growth and load fluctuations by adding nodes and re-
balancing

The estimates show the number of servers of each type required to sustain the 
service level of 540000 transactions per hour of the Inventory system with 
revisioning additional CPU cost and at different server utilizations. The cost 
calculation is based on the CPU per transaction derived from a reference system
with a different processor and system architecture. We did not factor in any 
coefficients derived from comparative processor benchmarks, assuming that 
these are not available.

However, the assumption that the clock rate is the only quantifier to distinguish 
one CPU from another could result in a large error. In reality, as we shall see 
later, fewer servers of each kind were needed to sustain the throughput and the 
overcapacity resulting from the initial estimate could be used to scale up the 
workload and achieve higher throughput.

SERVER SYSTEM CONFIGURATION AND SIZING

After estimating the CPU capacity required to accommodate the load and 
sustain the response time and throughput of the production system and
deriving the database cluster size, a choice of the server configuration can be made.

Based on the above estimates, the database clusters(s) could be composed of

- 17 Xeon-based Dell PowerEdge 1750 servers
- 7 Itanium 2-based Dell PowerEdge 7250 servers

if the individual nodes are to be restricted to 75% usage of their CPU capacity to allow for headroom.

For the Inventory system, the choice would also be predicated by the ability of the system to reduce the CPU time per transaction, support the I/O rates and possibly maintain a large buffer cache, as we earlier established that the main constituents of the transactions service time are CPU and I/O wait time.

The following considerations are guidelines, but do suggest a minimum configuration based on the workload requirements and capacity estimates.

Server Options

In our hypothetical case, the possible choices for the server configuration consisted of Dell PowerEdge 1750 2-way Intel Xeon-based machines and Dell PowerEdge 7250 4-way Itanium 2-based nodes.

The Dell 1750 Xeon-based system is a rack-dense server providing affordable and demonstrated grid scalability. This configuration delivers solid performance per node, allows for higher granularity of scaling, and enables new customers to start small and grow their grid by small increments.

The grid composed of the Dell PowerEdge 7250 4-way Itanium 2-based servers provides impressive performance per node, requiring half the physical nodes to achieve almost twice the performance. Larger L3 caches on the Itanium 2 processor may help CPU bound applications by reducing cache misses and cycles for memory accesses.

Memory Addressability

The original system had 90 GB of memory and used a 36GB buffer cache. As there is no data regarding the efficiency of the buffer cache usage, but historical data indicated that there was free main memory at peak times, one should provide for the possibility of having to use a cache larger than 4GB.

However, the memory size and the workload of the original system will be distributed over multiple machines, so that – assuming 80% utilization of main memory in the Inventory production system – the required memory per node would be
• Memory used in the original system / estimated number of cluster nodes; e.g.,
  o Dell PowerEdge 1750: 90GB \* 0.80 / 17 = 4.2GB
  o Dell PowerEdge 7250: 90GB \* 0.80 / 7 = 10.3GB.

For the lower bound, one can use the SGA size and compute
• SGA size of the original system / estimated number of cluster nodes, e.g.
  o Dell PowerEdge 1750: 36GB / 17 = 2.1GB
  o Dell PowerEdge 7250: 36GB / 7 = 5.1GB.

In conclusion, the systems used in the cluster should be fitted initially with
• 4GB of memory when the Dell PowerEdge 1750 is used
• 5-10 GB of memory when the Dell PowerEdge 7250 is used.

To put the calculations performed into perspective,
• More memory and a larger buffer cache may reduce I/O and therefore increase the transaction rates and lower response times if the working set fits into the cache,
• A uniform, randomly accessed large database which does not fit into a large cache will still be I/O intensive
  o The database size of the Inventory production system was 1.2TB and the buffer cache used amounted to 30GB, i.e. only about 3% of the database fit into the cache, i.e. with 17x2GB for Dell 1750s and 7x4GB for Dell 7250s - about the same working set size can be cached thereby achieving the same cache size of the original SMP environment
• The “global cache” is extensible by adding nodes, local cache misses will be served from a remote cache via IPC or from disk
• For 32-bit architectures, memory extensions such as VLM from Oracle, are available to extend the address range

**Number of I/O Slots**
The choice of the server also depends on the number of I/O adapters the architecture can support. For maximum availability, redundant network interface cards and host bus adapters should be accommodated. More specifically
- 2 Gigabit Ethernet NICs for the private network,
- 2 Gigabit Ethernet NICs for the public network,
- For SAN storage connectivity, 2 HBAs and multi-path support.

For performance, the presence of a second adapter may be necessary for traffic across the private interconnect if the volume of data is higher than the effective bandwidth that the Gigabit Ethernet NIC allows. This may happen in Decision Support and Data Warehousing applications using Parallel Query, but will rarely be an issue in OLTP systems with random, single block data referencing patterns.

Similarly, a single HBA should (theoretically) support up to 2Gb per second I/O throughput, assuming that the SAN is based on Fibre channel; i.e., a single adapter provides sufficient capacity for the I/O workload driven by the servers.

**Network Adapters and Frame Sizes**

In view of the fact that a considerable amount of data blocks may have to be read over the private network, it is suggested to ensure that the

- Ethernet NICs and switches used to build the database cluster support 9K frames.

The advantage of using these Jumbo frames is that they can contain a full ORACLE data block and dispense with the need to fragment and reassemble it.

**NETWORKED STORAGE CAPACITY PLANNING**

The decisions on the choice of hardware as well as the storage network infrastructure must be made according to the application requirements and the service-level objectives of the organization.

In this section storage capacity planning is discussed.

**Storage Capacity**

Developing successful storage capacity planning requires thorough examination of the size and performance requirements of the application services in the environment, which is the prerequisite for the storage procurement and configuration.

Prior to the storage configuration and layout development, the following factors must be considered:

- The number of application software packages: how many software packages are serviced in the environment?
• Capacity characteristics: how much storage capacity does each software package serviced in this environment require? Will it increase in size?

• I/O characteristics: what types of read/write tendencies does each software package perform? What is the mix of the sequential or random I/O operations? What is the required response time for each application service?

• Fault tolerance: how tolerant should each software package be against disk hardware failure?

• Protection: what type of backup does each software package require?

• Limitation: what is the limitation of the operating system in terms of the supported number of devices? (E.g., the Linux operating system’s kernel version 2.4 supports up to 256 devices.)

A logical volume (as opposed to a physical disk) is the smallest assignable unit of the storage resource. A logical volume must be configured in a storage array before it can be assigned (provisioned) to a physical server as a usable storage resource (this is still the case even if a host-based local volume manager is used).

The list above captures major considerations that are part of the pre-deployment planning exercise for the storage configuration. Such an exercise could be a complex task even for a single application, but employing the storage class concept that uses the features and capabilities of the various storage systems helps reduce the complexity of developing the best storage configuration in a flexible fashion.

Prior to developing the sizing requirement for the storage system, it is important to know the backend storage configuration options and their impact on the sizing practice. For example, the RAID level chosen could potentially double the storage required as shown in Appendix 1 – RAID Level Comparison.

Application and Database Size

As the first step in storage capacity planning, the initial storage configuration must be built based on the number of application software packages and their capacity requirements. To illustrate this, consider the following example:

• Two separate instances of Oracle Database 10c Real Application Clusters for production
• Two separate instances of Oracle Application Server 10c as mid-tier for the production databases
• A shared directory for user home directories
• A shared directory as a file repository
Based on this information, the following table summarizes the minimal requirements for the storage capacity.

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Software</th>
<th>Initial Size</th>
<th>1-year Growth Rate</th>
<th>Storage Class</th>
<th>Total Size for 1-year Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Oracle Database 10g</td>
<td>Application binaries</td>
<td>Oracle Database 10g RAC</td>
<td>6GB</td>
<td>N/A</td>
<td>Application</td>
<td>6GB</td>
</tr>
<tr>
<td>Production database 1</td>
<td>Data and logs (ASM disk group members)</td>
<td>N/A</td>
<td>500GB</td>
<td>20%</td>
<td>Enterprise-level User Data</td>
<td>600GB</td>
</tr>
<tr>
<td>Production database 1</td>
<td>System Logs</td>
<td>N/A</td>
<td>10GB</td>
<td>100%</td>
<td>Application</td>
<td>20GB</td>
</tr>
<tr>
<td>Production database 2</td>
<td>Data and logs (ASM disk group members)</td>
<td>N/A</td>
<td>400GB</td>
<td>25%</td>
<td>Enterprise-level User Data</td>
<td>500GB</td>
</tr>
<tr>
<td>Production database 2</td>
<td>System Logs</td>
<td>N/A</td>
<td>10GB</td>
<td>100%</td>
<td>Application</td>
<td>20GB</td>
</tr>
<tr>
<td>Staging database 1</td>
<td>Data and logs (ASM disk group members)</td>
<td>N/A</td>
<td>50GB</td>
<td>0%</td>
<td>Departmental-level User Data</td>
<td>50GB</td>
</tr>
<tr>
<td>Staging database 2</td>
<td>Data and logs (ASM disk group members)</td>
<td>N/A</td>
<td>40GB</td>
<td>0%</td>
<td>Departmental-level User Data</td>
<td>40GB</td>
</tr>
<tr>
<td>Shared Oracle Application Server 10g</td>
<td>Application binaries</td>
<td>Oracle Application Server 10g</td>
<td>6GB</td>
<td>N/A</td>
<td>Application</td>
<td>6GB</td>
</tr>
<tr>
<td>Production mid-tier 1</td>
<td>Data</td>
<td>N/A</td>
<td>4GB</td>
<td>N/A</td>
<td>Departmental-level User Data</td>
<td>4GB</td>
</tr>
<tr>
<td>Production mid-tier 1</td>
<td>System Logs</td>
<td>N/A</td>
<td>15GB</td>
<td>100%</td>
<td>Application</td>
<td>30GB</td>
</tr>
<tr>
<td>Production mid-tier 2</td>
<td>Data</td>
<td>N/A</td>
<td>2GB</td>
<td>N/A</td>
<td>Departmental-level User Data</td>
<td>2GB</td>
</tr>
<tr>
<td>Production mid-tier 2</td>
<td>System Logs</td>
<td>N/A</td>
<td>10GB</td>
<td>100%</td>
<td>Application</td>
<td>20GB</td>
</tr>
<tr>
<td>User home</td>
<td>User home</td>
<td>N/A</td>
<td>128GB</td>
<td>N/A</td>
<td>Utility</td>
<td>128GB</td>
</tr>
<tr>
<td>Shared repository</td>
<td>File repository</td>
<td>N/A</td>
<td>256GB</td>
<td>N/A</td>
<td>Utility</td>
<td>256GB</td>
</tr>
</tbody>
</table>

Table 9: Storage Configuration Requirements Example

In addition to the initial storage capacity requirements, the extra capacity needs to be calculated according to the backup requirements.

**Backup Requirements**

In planning for the backup requirements, the three most important factors are the backup types, the frequency and the retention.

For the backup types, the two most important distinctions are the full and incremental backups. In planning for incremental backup the most important information is the size of the delta of the data over a period of time, in
conjunction with the frequency (how often its run) and the retention (how long its kept).

The detailed discussion on the various backup and replication technologies provided by Oracle software and EMC storage systems is also beyond the scope of this white paper. The information in this section covers general considerations for the storage requirements for the backup; therefore, the sizing information here applies to the replicated data as a result of backup, regardless of the backup methodology.

Based on the three factors, consider the following backup requirements for the example environment used in this section.

The online backup requirements for both production databases are the weekly full backup and the daily incremental backup, with the retention of 1 month for all the backup data (offline backup is not discussed here).

The simple calculation based on the above requirements is (incremental is simplified as 5% of the total size):

- Database 1: \((600\text{GB} \times 4 \text{ [weeks/month]} + (600\text{GB} \times 0.05 \text{ [incremental]} \times 7 \text{ [days]} \times 4 \text{ [weeks]} = 2,400\text{GB} + 840\text{GB} = 3,240\text{GB})

- Database 2: \((500\text{GB} \times 4 \text{ [weeks/month]} + (500\text{GB} \times 0.05 \text{ [incremental]} \times 7 \text{ [days]} \times 4 \text{ [weeks]} = 2,000\text{GB} + 700\text{GB} = 2,700\text{GB})

Therefore, for this sample environment, approximately 6TB of additional storage for the Backup storage class is required.

### Storage System Requirements Summary

To decide the storage configuration and layout for the environment, it is useful to characterize the storage platforms because different types of data require different types of storage according to size, performance and fault tolerance requirements.

<table>
<thead>
<tr>
<th>Storage Class</th>
<th>Usage</th>
<th>RAID Level</th>
<th>Performance Characteristics</th>
<th>Size Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>For application binaries and their logs</td>
<td>RAID 1+0*1</td>
<td>Medium</td>
<td>Low</td>
</tr>
<tr>
<td>Enterprise-level User Data</td>
<td>For database data files and online logs</td>
<td>RAID 1+0</td>
<td>High</td>
<td>Medium-High</td>
</tr>
<tr>
<td>Departmental-level User Data</td>
<td>For database data files and logs</td>
<td>RAID 5 (SAN) or RAID 1+0*1 (NAS)</td>
<td>Medium</td>
<td>Medium-High</td>
</tr>
<tr>
<td>Storage Class</td>
<td>Usage</td>
<td>RAID Level</td>
<td>Performance Characteristics</td>
<td>Size Characteristics</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------</td>
<td>---------------------</td>
<td>-----------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Backup</td>
<td>For online backups of user data</td>
<td>RAID 5, or RAID 3 with ATA</td>
<td>Cost Effective</td>
<td>High</td>
</tr>
<tr>
<td>Utility</td>
<td>For shared directories to store tools, sources, etc.</td>
<td>RAID 1+0* or RAID 5+0*</td>
<td>Medium</td>
<td>High</td>
</tr>
</tbody>
</table>

Note 1: The RAID level may be the combination of the hardware- and software-based.

### Table 10. Storage Classes

In summary, the storage systems required for building this environment are:

- **High Performance and Capacity**: SAN storage system (e.g., EMC Symmetrix DMX) with at least 3TB of raw capacity.
- **Medium Performance and Capacity**: SAN storage system (e.g., EMC CLARiiON CX) with at least 200GB of raw capacity based on FC, and 10TB (assuming 4+1 RAID3) of raw capacity based on ATA.
- **Cost-effective, High Capacity**: NAS system (e.g., EMC CNS or NS series) with at least 1TB of raw capacity (assuming RAID1 configured in the backend)

Using these storage systems, the following storage classes can be configured:

- Multiple 32GB volumes of the **Application storage class** using for example, the EMC NS Series/Integrated system
- Multiple 64GB volumes of the **Enterprise-level User Data storage class** using, for example, the EMC Symmetrix DMX series
- Multiple 64GB volumes of the **Departmental-level User Data storage class** using, for example, the EMC CLARiiON CX system
- Multiple 600GB volumes of the **Backup storage class** using, for example, the EMC CLARiiON CX series with ATA
- Several 128GB volumes of the **Utility storage class** using, for example, EMC NS Series/Gateway system

The storage volumes should be configured modularly: i.e., when configuring multiple volumes as a specific storage class for a specific data type, the size of the volumes should be standardized, especially for the User Data storage classes. That way, it is easy to configure the volumes because the configuration can be standardized, and it is easy to manage the volumes because the standardized volumes are easily swappable, reusable, and therefore flexible.

### Storage Configuration Decisions for the MegaGrid

The sizing requirements for the Phase I of the MegaGrid are summarized in the following table:
<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Software</th>
<th>Initial Size</th>
<th>1-year Growth Rate</th>
<th>Storage Class</th>
<th>Total Size for 1-year Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Oracle Database 10g</td>
<td>Application binaries</td>
<td>Oracle Database 10g RAC</td>
<td>6GB</td>
<td>N/A</td>
<td>Application</td>
<td>6GB</td>
</tr>
<tr>
<td>Inventory database mg32</td>
<td>Data and logs (ASM disk members)</td>
<td>N/A</td>
<td>360GB</td>
<td>400%</td>
<td>User Data</td>
<td>1.5TB</td>
</tr>
<tr>
<td>Inventory database mg64</td>
<td>Data and logs (ASM disk members)</td>
<td>N/A</td>
<td>360GB</td>
<td>400%</td>
<td>User Data</td>
<td>1.5TB</td>
</tr>
<tr>
<td>User home</td>
<td>User home</td>
<td>N/A</td>
<td>128GB</td>
<td>N/A</td>
<td>Utility</td>
<td>128GB</td>
</tr>
<tr>
<td>Shared repository</td>
<td>File repository</td>
<td>N/A</td>
<td>256GB</td>
<td>N/A</td>
<td>Utility</td>
<td>256GB</td>
</tr>
</tbody>
</table>

Table 11. Sizing Requirement for MegaGrid Phase I

On the basis of the I/O performance and sizing requirements described above for the Inventory production system, the following storage configuration was determined:

- Approximately 10000 IOPS and build volumes striped over 8 members (150 IOPS per spindle * 8 = 1200 IOPS)
- 24 volumes @ 64GB each (8+1)
- EMC Symmetrix DMX series.

Note that this configuration is expected to support about 19200 IOPS and provide 1.5TB of database storage per database.

In order to meet the backup requirements for a Telco Inventory database of the size described in Table 11 above, the calculation is

- \[(1.5\text{TB x 4 [weeks/month]} + (1.5\text{TB x 0.05 [incremental] x 7 [days] x 4 [weeks]}) = 6\text{TB + 2.1 TB = 8.1TB per database}\]

Additional Sizing Considerations for Adding Cluster Nodes

In addition to the user data, allowances must be made for each additional node added to a database cluster to accommodate

- Additional thread of redo, e.g. \#of redo logs x redo log size * \#of nodes in the cluster
- An undo tablespace for each additional instance
- Approximately 500MB of disk space for the SYSAUX tablespace to retain the AWR repository for all cluster instances without causing premature purging of snapshots
• File system space on low-cost storage for exported AWR repository data.

By virtue of ASM, disks can be added very quickly and data rebalanced when additional storage is required for the expansion of the cluster.

**Storage Infrastructure Issues**

An Enterprise Grid Computing environment by design adopts a large number of cost-effective server nodes to support multiple application services with a large amount of data. The storage-accessing traffic increases due to the sheer number of nodes but also the importance of the data increases substantially, in both the SAN and NAS infrastructures. The higher the number of services in the environment, the higher the traffic becomes and the higher the value of the data become. Hence, the demand for higher serviceability of the storage platform increases, especially for performance, fault tolerance, and protection.

**Bandwidth**

The storage bandwidth issue has two aspects; the "size of the pipe" of each point-to-point connection that has an impact on individual and aggregated traffic, and the design by which the nodes and the storage systems are connected.

The size of the pipe is rather simple; at the time of this writing, the maximum theoretical bandwidth of the fibre channel technology was 2Gb (gigabit) per second for each point-to-point communication, and the bandwidth for the Ethernet-based IP network was 1Gb per second. As for the aggregated traffic, the application requirements for the bandwidth should be analyzed carefully so that the most appropriate storage systems that can meet the requirements can be selected for deployment.

**Connectivity**

The key infrastructure consideration for storage connectivity is the accessibility of all the server nodes to all the storage resources in the environment. Any data, not just the user data in a database but also the executables are stored on storage before they are read into the memory on a particular server node. In an Enterprise Grid Computing environment, services are virtualized in a way that a combination of any server nodes can be dynamically assigned the task of
running any service. Server nodes, therefore, must be able to access any data as shared resources in the environment whether they are the user data for a database or some application executable binaries.

The same rule applies to both SAN and NAS infrastructures. To design the SAN infrastructure, it is important that all the volumes configured in all the storage systems can be accessed by connecting all the server nodes and the storage systems in the same SAN fabric. That is not to say that all the volumes must be visibly presented to all the servers; having a physical data path(s) from a server to all the storage systems is different from have the volumes already provisioned. The point is that volumes cannot be provisioned to a server unless there is a physical data path from the server to the storage systems providing the volumes.

As for the NAS environment, the key is to not only make sure that all the servers can communicate with all the NAS systems via IP network, but also all the servers can do so without having to go through a router (hop). That is, all the NAS systems should be connected to all the server nodes on layer 2, preferably via a segregated network dedicated for the NAS traffic.

**Network Design and Port Density**

The connectivity design can be complex especially in a large multi-node environment as in the Project MegaGrid environment. As noted above, the grid computing environment is built with multiple server nodes and storage systems, and the higher the number of discrete nodes, the more complex the connectivity design becomes. It is especially important to plan that the networking equipment be equipped with the appropriate port density.

For example, if the number of the server nodes is higher than the port density of the SAN switches deployed in the environment, the multiple SAN switches may need to be connected with inter-switch links (ISLs), as shown below.
The ISL will be the bottleneck if multiple servers from the left SAN (01) send I/O requests to the Storage 02, traversing the ISL.

Figure 3: Expanding port density to accommodate many servers

To alleviate the possible problem, the SAN switch with the appropriate port density to accommodate all the server nodes should be chosen (as shown below), or the number of ISLs should be increased to accommodate the traffic traversing the ISLs.
A SAN switch with enough switch ports to accommodate all the client and storage connections eliminate the possible performance issue caused by ISL.

**Figure 4:** Provisioning a big switch can accommodate many servers as well

The same rule applies to the NAS connectivity. Shown below is the situation similar to that of the SAN described above.

![SAN switch diagram](image)

**The aggregated throughput from the server group far exceeds the bandwidth of the inter-switch link, a possible bottleneck.**

**Figure 5:** NAS design requires careful end-to-end throughput planning

The network design must take into consideration the NAS traffic bandwidth requirements as well as the inter-node communication requirements, so that the ISL does not become the bottleneck.

**The NAS network design consideration**

**Port Density for the MegaGrid**

The initial assumption for the SAN configuration of the MegaGrid was that the server grid would grow to 64 servers in the first 2 phases of the project and use the additional ports to connect the EMC Symmetrix DMX 1000 and EMC CLARiion CX. The configuration therefore was initially...
• 2 switches with 112 ports = 224 ports in total

The total number of 64 servers could be connected comfortably, with sufficient ports for storage and ISLs to support the growth of the server and storage grid.

Tiered Storage and the Storage Class

Tiered networked storage is designed to address the very fact that not all information is of equal value and the value of information changes over time. For example, data of one application may be of greater value than those of the other application. Similarly, the data within the same application may be of different value, such as recent account information in an OLTP application as opposed to very old transactional data in the same application.

Virtualize and Automate Storage Management

The new ORACLE Database 10c Automatic Storage Management (ASM) functionality provides file system and volume manager capabilities built into the Oracle database kernel. ASM virtualizes the database storage into disk groups and spreads data evenly across all available storage resources. It has the ease of access of a file system and the performance of raw I/O.

Using ASM simplifies the task of adding disks to increase the database size or eliminate hot spots because it automatically rebalances the data while an application is running. When moving data to a high performance storage system to guarantee good IO service times or throughput, ASM can be used to migrate data from the a disk group virtualizing the currently used storage to the new, high-performance system online.

When creating a disk group one needs to decide what level of redundancy ASM should provide, whether

• External (no mirroring),
• Normal (mirroring) and
• High (triple mirroring).

It is generally recommend to

• Use External redundancy when creating disk groups with high-end storage arrays
• Add disks or LUNs with the same sizes and performance characteristics to a disk group
• Physically separate the database area from the recovery area by creating a different disk group for each
In the MegaGrid, two disk groups were created

- For the 24 64GB LUNs built from the Symmetrix DMX storage to store data and undo tablespaces and redo logs
- For the LUNs provided from the ATA disks to store archive logs and backups.

Based on the available performance and capacity planning, a simple provisioning model that could be used for all database storage was chosen. The redo log activity was not significant enough for the logs to be placed into a separate disk group.

**EFFECTIVE CLUSTER CAPACITY AND PERFORMANCE**

Based on our knowledge of clustered IA server performance in an Oracle environment, we estimated that the calculations used to predict the capacity required to downsize the Inventory application to a cluster might overestimate the CPU power necessary to fulfill the service level objectives.

In reality, the throughput of 544,000 business transactions per hour at less than 1 second average response time using 95% of their CPU capacity could be achieved with:

- (10) 2-way, Intel Xeon processor-based Dell PowerEdge 1750s, or:
- (4) 4-way, Intel Itanium 2 processor-based Dell PowerEdge 7250s

In comparison, the predictions for 100% utilization (assuming 5-10% cost for cluster coherency) were:

- 13-14 Dell PowerEdge 1750 servers
- 6-7 Dell 7250 PowerEdge servers

These results suggest that

- The cost of maintaining the global cache is lower than 5-10% of CPU per transaction
- The Intel Xeon and Itanium 2 processor-based Dell servers use fewer cycles to process the transaction load than the reference processors.

The following table compares the CPU cost per transaction based on the real data from the deployment of the Inventory system in the MegaGrid:
On the Itanium 2 processor-based Dell PowerEdge 7250 servers, the CPU per transaction was reduced by almost 50% when compared to the reference CPU. The Dell 7250 therefore would not only ensure a high transaction rate for the CPU intensive application, but also reduce the CPU time and therefore have a significant impact on the service and response times.

In view of the total cost of the servers, the overestimation and resulting excess capacity can be exploited

- To ensure a growth path
- As spare capacity for replacement, repair and provisioning
- To expand the cluster and run each individual server safely at a lower utilization to balance a “bursty” workload and with extra capacity for failover
- For additional applications or modules of the same application
- To parallelize operations which are scheduled to run at different times of the day
- As management and maintenance nodes to run management software, remote management consoles.

The following figures show the performance and scalability when additional nodes were provisioned:

<table>
<thead>
<tr>
<th>Reference CPU</th>
<th>Xeon</th>
<th>Itanium2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dell PE 1750</td>
<td>0.5109</td>
<td>0.4151</td>
</tr>
</tbody>
</table>

Table 12: CPU Cycles per Business Transaction Comparison

![Cluster of Dell 1750 Throughput/h](Image)
In summary, on both clusters, additional servers resulted in increased throughput at acceptable response times. The scalability of the clusters was 80%; i.e., each additional node added 80% of effective capacity and business volume.

Using 10 clustered PowerEdge 7250 servers with Itanium 2 Processors, the transaction volume that could be processed was almost 3 times higher than the throughput achieved on a large SMP. The other 4 available servers were used for maintenance, administrations and spares.

Given that spare capacity is available in the server and storage pool and that resource allocation is very flexible owing to the Grid infrastructure, the EMC storage setup, ORACLE Database 10g RAC and ASM, capacity management is simplified and capacity on demand provisioning is a distinct opportunity. The automation of performance monitoring built into the Server Manageability features in ORACLE Database 10g, in conjunction with the concept of SERVICE, MODULE and ACTION it supports, can provide the hints to a capacity management and provisioning model which works by “expanding” and “shrinking” of services.

CAPACITY MANAGEMENT BY EXPANDING AND SHRINKING OF SERVICES

When the tuning recommendations produced by the advisories indicate that additional CPU or storage bandwidth is required to increase the capacity of a service, there are various options in the Grid:

- If the elapsed time threshold is violated and the service is CPU bound,
- The service should be moved to nodes with spare capacity,
- Other services should be relocated.
- Additional nodes may have to be added from a spare pool or by shrinking less important services.

Figure 8: CPU bound services

- If the elapsed time threshold is violated and the service becomes I/O bound, i.e. User I/O in the service statistics consumes significant time,
  - Disks may have to be added to the disk groups primarily used by the service and the data rebalanced. The additional disks could be taken from other disk groups and databases managed by Oracle ASM.
  - The capacity of the storage system is exceeded and the disk group used by the application must be moved to a high-performance storage system; e.g., an estimation of the IO requirements indicates that a DMX 1000 would be suitable to sustain the capacity for the service.
The corresponding capacity increase by removing and adding disks or migrating to a different storage system is managed efficiently by Oracle ASM.

Figure 9: I/O bound services

CONCLUSION

ORACLE Database 10g provides a flexible framework to manage capacity for databases by monitoring and detecting bottlenecks caused by the saturation of a resource by adjusting application usage patterns, rearranging or extending server and storage configurations. An integral part of the flexibility is provided by the Service Concept, ORACLE Real Application Clusters and ORACLE ASM.

The combination of standard, high-volume servers and storage systems allowing for incremental increases of workload and changes of usage patterns as well as adjustment to higher performance requirements, along with the networking technology to interconnect these resources, can allow capacity to be available on demand and can be efficiently used a relatively low-cost.

Capacity planning and management is critical to ensuring the Enterprise Grid Computing infrastructure will scale. Through Project MegaGrid, ORACLE, EMC, Dell and Intel have pre-tested and validated the optimal combinations of
technologies to design an Enterprise Grid environment where capacity planning and management can be efficiently achieved.
APPENDIX A – RAID LEVEL COMPARISON

The table below summarizes the types of the RAID configurations and their impact on the size available from the EMC storage platforms.

<table>
<thead>
<tr>
<th>RAID Level</th>
<th>Type</th>
<th># of Disks</th>
<th>Size</th>
<th>Fault Tolerance</th>
<th>Performance Characteristics</th>
<th>Storage Systems Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Stripe</td>
<td>2 or more</td>
<td>Total of all the disks</td>
<td>None (failure of a single disk causes the failure of the entire set)</td>
<td>Good read and write (the more disks, the better)</td>
<td>Symmetrix DMX NAS</td>
</tr>
<tr>
<td>1</td>
<td>Mirror</td>
<td>2</td>
<td>Half of the total</td>
<td>Tolerates a failure of one of the two disks</td>
<td>Basically the same as a single disk, but the write operation requires twice the effort</td>
<td>Symmetrix DMX CLARiiON CX NAS</td>
</tr>
<tr>
<td>3</td>
<td>Stripe with dedicated parity</td>
<td>3 or more</td>
<td>Total minus the size of a single physical disk</td>
<td>Tolerates a failure of one disk in the set</td>
<td>Read operation can be as good as the RAID 0 (stripe), but the write operation requires additional tasks of parity calculation, which usually involved a read of parity information prior to the write.</td>
<td>CLARiiON CX</td>
</tr>
<tr>
<td>5</td>
<td>Stripe with distribute d parity</td>
<td>3 or more</td>
<td>Total minus the size of a single physical disk</td>
<td>Tolerates a failure of one disk in the set</td>
<td>Read operation can be as good as the RAID 0 (stripe), but the write operation requires additional tasks of parity calculation, which usually involved a read of parity information prior to the write.</td>
<td>Symmetrix DMX CLARiiON CX</td>
</tr>
<tr>
<td>1+0</td>
<td>Mirror and stripe</td>
<td>4 or more, even number</td>
<td>Half of the total</td>
<td>Tolerates multiple failures of up to half of the total number of the disks in the set, unless the failure occurs to the same mirror set</td>
<td>Good read and write (the more disks, the better), but the write operation requires twice the effort</td>
<td>Symmetrix DMX NAS</td>
</tr>
<tr>
<td>5+0</td>
<td>Stripe and distribute d parity, and then stripe</td>
<td>6 or more, even number</td>
<td>Total – (the size of a single physical disk x the number of RAID5 subsets)</td>
<td>Tolerates multiple failures of up to the number of RAID5 subsets in the entire set, unless the second failure occurs to the same RAID5 subset</td>
<td>Read operation can be as good as the RAID 0 (stripe), but the write operation requires additional tasks of parity calculation, which usually involved a read of parity information prior to the write.</td>
<td>NAS</td>
</tr>
</tbody>
</table>