# Oracle® Agile Product Lifecycle Management for Process

Capacity Planning Guide

Release 6.2.1

**E66769-01**

February 2017

ORACLE®

Oracle Agile Product Lifecycle Management for Process Capacity Planning Guide, Release 6.2.1

E66769-01

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

# Contents

**Preface** ..... v

Audience ..... v
Variability of Installations ..... v
Documentation Accessibility ..... vi
Software Availability ..... vi
Conventions ..... vi

## 1 Components and Requirements

**Components** ..... 1-1
Application Server ..... 1-1
RemotingContainer Server ..... 1-1
Database Server ..... 1-1
Load Balancer ..... 1-1
Reverse Proxy ..... 1-2
Clients ..... 1-2
**Software and Hardware Requirements** ..... 1-3

## 2 Capacity Planning

**Basic Recommended Configuration** ..... 2-1
Testing Details ..... 2-2
**Basic Application Server Capacity** ..... 2-3
Testing Details ..... 2-3
**Sizing and Scaling Strategy** ..... 2-3
Application Server ..... 2-3
Number of Application Servers ..... 2-4
Application Server Memory ..... 2-4
Testing Details ..... 2-5
**Database Server** ..... 2-5
Network Bandwidth ..... 2-6
Testing Details ..... 2-6
File Server ..... 2-7
**Scale Notes of Runtime Data** ..... 2-7
Formulation specification vs. inputs example ..... 2-8
Case 1: Add a number of inputs to one step ..... 2-8
Case 2: Add a number of inputs to five steps ..... 2-8

Material Specification vs. ExtData ..... 2-9
Case 1: Add a number of extended attributes to one material ..... 2-9
Case 2: Add a number of custom sections to one material ..... 2-9

## 3 Performance Tips

**Database Server** ..... 3-1
**Fragmentation** ..... 3-1
SQL Server ..... 3-1
Oracle ..... 3-2
**Optimizer Statistics** ..... 3-2
SQL Server ..... 3-2
Oracle ..... 3-2
**Tablespace and Data File Configurations** ..... 3-2
SQL Server ..... 3-2
Oracle ..... 3-3
One-Disk vs. Small Size ..... 3-3
Two-Disk vs. Medium ..... 3-4
**Indexes** ..... 3-4
**Caching and Compression** ..... 3-5
Caching ..... 3-5
Compression ..... 3-6
**DB Settings Reporting Utility for Oracle** ..... 3-6
**Performance Logging** ..... 3-6
**CPU/Memory Consumption and Other Indicators** ..... 3-6
Usage ..... 3-7
**Troubleshooting Performance Issues** ..... 3-8
**(Experimental) Index Utility for Oracle** ..... 3-9
**(Experimental) Reduce Hard Parsing for Oracle** ..... 3-10

## A Capacity Testing Declarations

**Data Complexity Score** ..... A-1
**Testing Environment** ..... A-3
**Testing and Threshold Definitions** ..... A-4
Testing Details ..... A-4

## B Application Pool Distribution

**Application Pool Distribution** ..... B-1
Testing Details ..... B-2

# Preface

The *Agile Product Lifecycle Management for Process Capacity Planning Guide* provides guidelines and recommendation for planning your implementation of Agile Product Lifecycle Management (PLM) for Process.

This preface contains these topics:


- Audience
- Variability of Installations
- Documentation Accessibility
- Conventions


## Audience

This guide is intended for end users who are responsible for creating and managing information in Agile PLM for Process. Information about administering the system resides in the *Agile Product Lifecycle Management for Process Administrator User Guide*.

## Variability of Installations

Descriptions and illustrations of the Agile PLM for Process user interface included in this manual may not match your installation. The user interface of Agile PLM for Process applications and the features included can vary greatly depending on such variables as:

- Which applications your organization has purchased and installed
- Configuration settings that may turn features off or on
- Customization specific to your organization
- Security settings as they apply to the system and your user account

# Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

### Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

# Software Availability

Oracle Software Delivery Cloud (OSDC) provides the latest copy of the core software. Note the core software does not include all patches and hot fixes. Access OSDC at:

http://edelivery.oracle.com.

# Conventions

The following text conventions are used in this document:

| Convention | Meaning |
|---|---|
| **boldface** | Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary. |
| *italic* | Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values. |
| `monospace` | Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter. |

# 1

# Components and Requirements

Below is a list of the components of the Agile PLM for Process application suite and directly related third party hardware devices.

## Components

### Application Server

The Agile PLM for Process Application Server is the base for the PLM for Process platform, where all common services and business logic reside for the entire solution. All client servers and users connect to the Application Server either directly or indirectly.

### RemotingContainer Server

The Remoting Container is a companion application for Agile PLM for Process. The Remoting Container application is a collection of services which handle specialized background processing for the PLM for Process application. For more information around the Remoting Container see the *Agile Product Lifecycle Management for Process Install/Upgrade Guide*. This service can be hosted on the existing Application Server without any impact to overall performance or scalability.

### Database Server

The Agile PLM for Process Database Server persists or stores all product content and system settings. This database can run on supported Microsoft SQL Server or Oracle versions.

### Load Balancer

The hardware load balancer brokers client communications without compromising the security of your internal network. Clients communicate through the load balancer with the application server.

There are no Agile PLM for Process software components running on the hardware load balancer. They are usually deployed in the Demilitarized Zone (DMZ) where it proxies requests from outside the corporate firewall to the application server in the Safe Zone over NAT.

## Reverse Proxy

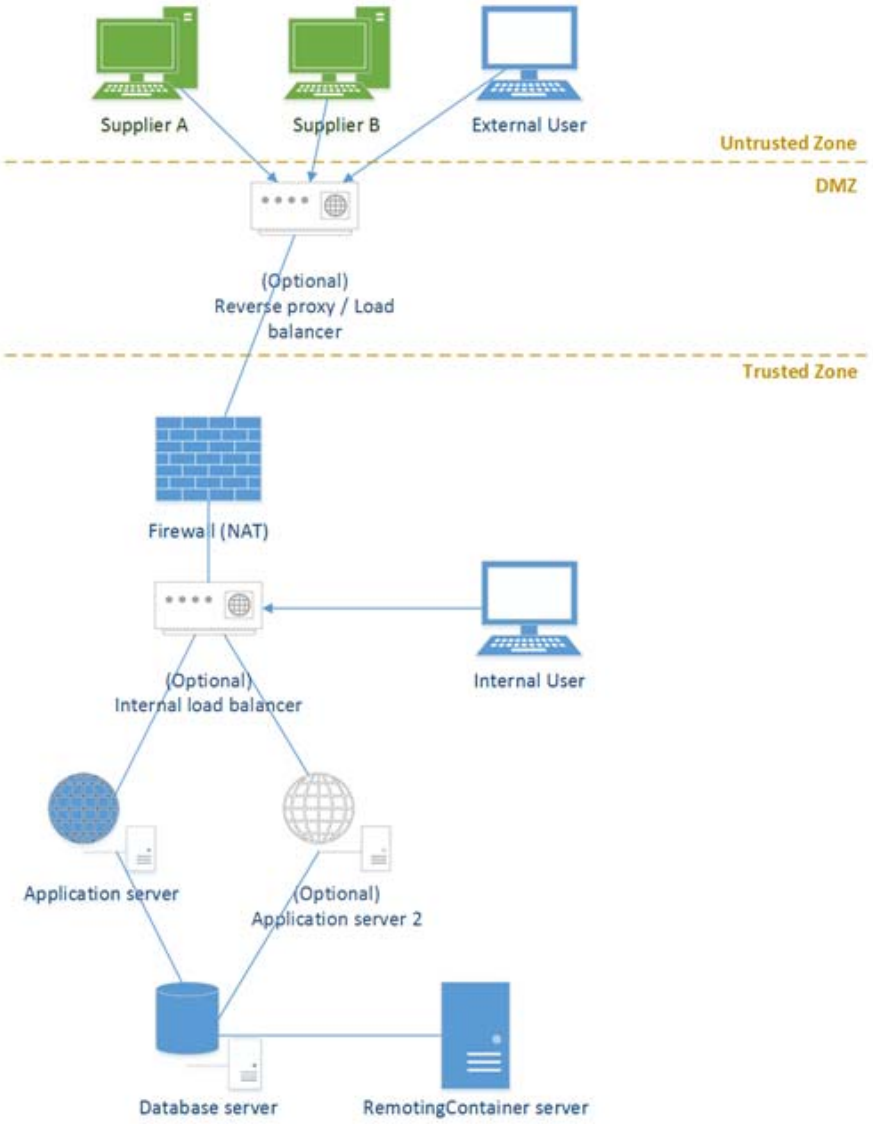
The reverse proxy acts as an intermediary, retrieving resources on behalf of the client from a number of servers. Typically, a reverse proxy is placed in the DMZ and will proxy client requests from servers in the intranet.

There are no Agile PLM for Process software components running on the reverse proxy, but using a reverse proxy for the Supplier Portal is recommended.

## Clients

Agile PLM for Process uses a web client, a thin HTML client that uses standard HTTP/HTTPS protocols.

***Figure 1–1 System overview***



# Software and Hardware Requirements

For general software and hardware requirements, refer to the *Agile Product Lifecycle Management for Process Install/Upgrade Guide* on OTN for your specific release:

http://www.oracle.com/technetwork/documentation/agile-085940.html#plmprocess

# 2

# Capacity Planning

This chapter helps you plan and gauge server capacity.

## Basic Recommended Configuration

There are a number of factors to consider when determining the implementation size for a production environment, and it is up to the customer's business and IT team to determine what makes sense for their organization. However, the simplest method for estimating size is by analyzing the number of planned registered users over the next three years.

Based on the test results and sizing and scaling calculations upon a "Medium" size, a set of recommendations based on implementation size has been given. This does not account for additional servers required for high availability.

These recommendations are based on the number of registered users, which is different from the number of concurrent users, which is the number of users actively engaged in using the system. So customers should consider the exact ***exchange rate between the number of registered users and the number of working users*** according to the actual situation. 50% seems a normally reasonable value for common companies, and we should often preserve a 20% buffer against the user concurrency threshold when evaluating a real environment. For instance, a 50-concurrent-users declaration often suggests an 80-registered-users possibility. Please review the following grid about the balance among Size, Performance, Users and Hardware.

*Table 2–1 Recommendations {Balance based Large} = {Medium} * 2*

| Size | Registered Users | Expected Response Time | Actual Response Time | Configuration |
|---|---|---|---|---|
| Small | 160 | <= 5 sec | 6.4 sec | Servers<br>■ 1 Application Production with 2 cores CPU and 8GB RAM<br>■ 1 Data Administration Staging with 2 cores CPU and 8GB RAM<br>■ 1 SupplierPortal/eQ with 2 cores CPU and 8GB RAM<br>■ 1 DB with 2 cores CPU and 8GB RAM<br>Additional Hardware<br>■ Load Balancer (optional)<br>■ Reverse Proxy (optional) |
| Medium | 300 | <= 5 sec | 3.5 sec | Servers<br>■ 2 Application Production with 4 cores CPU and 16GB RAM<br>■ 1 Data Administration Staging with 4 cores CPU and 8GB RAM<br>■ 1 SupplierPortal/eQ with 4 cores CPU and 16GB RAM<br>■ 1 DB with 4 cores CPU and 16GB RAM<br>Additional Hardware<br>■ Load Balancer (recommended)<br>■ Reverse Proxy (recommended) |
| Large | 600 | <= 5 sec | 3.5 sec | Servers<br>■ 2+ Application Production with 4 cores CPU and 16GB RAM<br>■ 1 Data Administration Staging with 4 cores CPU and 8GB RAM<br>■ 1 SupplierPortal/eQ with 4 cores CPU and 16GB RAM<br>■ 1 DB with 4 cores CPU and 16GB RAM (or other enterprise solution)<br>■ 1 Reporting DB with 4 cores CPU and 8GB RAM (or other enterprise solution)<br>Additional Hardware<br>■ Load Balancer (recommended)<br>■ Reverse Proxy (recommended) |

## Testing Details

Thresholds are outlined in the testing results below. The testing would stop once average response time exceeded 30 seconds or any resource reached its direct ceiling. For more information about environment and testing methods, please refer to:

Appendix A, "Capacity Testing Declarations"

Appendix B, "Application Pool Distribution"

*Table 2–2 Testing details*

| Size | Average Response Time | 90th% Response Time | Concurrent Users Threshold | Bottleneck Description | Conclusion |
|---|---|---|---|---|---|
| Medium | 1.6 sec | 3.5 sec | 180 | When the 180th user joined the load test loop, the response time went beyond 5 seconds on average. At the same time, application server CPU occupation raised up to 90%. | Qualified for 180 * 80% = 144 concurrency equating about 300 registered users. |
| Small | 2.5 sec | 6.4 sec | 100 | When the 100th user joined the load test loop, the response time went beyond 5 seconds on average. At the same time, application server CPU occupation raised up to 90%. | Qualified for 100 * 80% = 80 concurrency equating about 160 registered users. |

# Basic Application Server Capacity

To determine the application server capacity, the average Transactions per Second (TPS) the server can support must be determined. For each application, business scenarios were identified that users with different roles would perform daily. Based on these scenarios and the user distribution, the workload is designed per application.

In the first phase, tests were conducted on individual application pools to determine the TPS values. A single typical application server supported an average 5 seconds response time.

## Testing Details

The TPS of representative applications are as follows.

**Note:** The granularity of transaction strongly depends on the test case design. In this guide, one transaction often includes 1-2 server communications.

| Application | TPS (ThinkTime Involved) |
|---|---|
| GSM | 0.19 |
| NPD | 0.11 |
| (Combined) | 0.19 |

# Sizing and Scaling Strategy

## Application Server

When planning on sizing a production environment, the number of application servers and the amount of memory will need to be determined. The largest factors that determine both of these are the number of users simultaneously logged into the applications as well as the size of the database, especially the amount of custom data.

### Number of Application Servers

The baseline testing suggests that a 4 core application server is the optimal configuration and is able to support 180 concurrent users, who are actively hitting the system. The configuration provides a 5 seconds response time (±15%) in 90th%. Considering the differences between size “Small” and size “Medium”, CPU cores often linearly leverage the other measures.

If it is estimated that there will be fewer than 160 users, then 1 server would be enough, or an additional one could be added if failover is needed. Theoretically, 1 server should be added for every additional 150 users. Again, to support multiple application servers, clustering should be implemented, which adds an additional 15% load on each server.

**Note:** This is a typical estimate and will vary depending on how aggressively users are using the system. Refer to Table 2–1, "Recommendations {Balance based Large} = {Medium} * 2" for the exchange rate between the number of registered users and the number of working users.

### Application Server Memory

The memory need for each application server depends on the maximum number of logged in users, the size of the user's session, the number of applications running on each server and the base amount of memory needed for each application (for example, initial memory footprint).

**Note:** It is recommended to measure the on-site amount of memory needed since its value strongly depends on deployments. Particularly, user session size will vary from customer to customer. The reference values of the guide-scoped environment are provided below.

| Application | User Session Size |
|---|---|
| GSM | 62MB |
| NPD | 50MB |

For a specific deployment, if you take 60MB as the average session size, and 1GB as the initial memory footprint, then the table below shows the basic memory requirements. So 8GB memory is reasonable for this “small” example. IIS always executes a dynamic session/memory management strategy, so it won’t go into the situation of memory shortage even though the number of active users is a little greater than 80. Instead, the application pool is going to be recycled more frequently.

*Figure 2–1 Basic memory requirements*

| | |
|---|---|
| User Session Size | 60MB * 80 Users = 4.8GB |
| Application Initial Load | 1GB |
| OS Base | 2GB |
| **Memory Requirement** | **7.8GB** |
| Recommended Configuration | ☐4GB ☑8GB ☐12GB ☐16GB |

### Testing Details

To get an accurate user session size for a specific environment, the test is going to run a load test using load testing software with at least 100 active users hitting a single application pool. The server was assigned sufficient memory in order to avoid auto-recycling impacting the result.

Using GSM as the example:

1. Prime the application with a single user and measure the initial memory baseline.
2. Execute the test with 100 users simulating a reasonable test case.
3. At the end of the test, measure the final memory consumed, subtract the baseline, and divide by the total number of users. This provides an approximate user session size for GSM.

```
User Session Size = (Final Consumed - Initial Baseline) / Number of Users
= (7373 MB - 1137 MB) / 100 users
= 62.4 MB
```

## Database Server

For production environments, it is strongly recommended to run the database server on dedicated hardware. Database hardware sizing depends on both concurrent usage and the amount of data or size of the database. The easiest measure of database size is initial dump file size and estimated monthly incremental increases.

Exporting the schema at periodic intervals and analyzing its size helps a customer determine if a larger database sizing model is needed to better manage database growth, and to minimize ongoing database maintenance and tuning.

For existing customers, getting the initial dump file size as a baseline is easy. For new customers, the dump file size can approximately equal the certified database size.

Also, the size of the database must be estimated by monitoring database growth to predict future disk size needs.

For instance, assuming initial dump file is 1GB, and the size growth is about 100MB per month, it's recommended to preserve at least 6 months buffer (600MB disk) and continue analyzing the new size every month.

Alternatively, it would better to monitor the exact data file size with the same policy. (Refer to the table space physical file "*.dbf" in Oracle or the data/log physical file "*.mdf/*.ldf" in SQL Server.)

Finally, assuming the hardware recommendations outlined in the *Agile Product Lifecycle Management for Process Install_Upgrade Guide* are followed, the database cannot normally be a performance bottleneck when considering what hardware to buy. The DBA set the parameter "CURSOR_SHARING = FORCE" in this sample database for

capacity planning testing, because the advance experiments hint that would bring a better overall performance. Furthermore, it's recommend following the tips described in Chapter 3, "Performance Tips".

## Network Bandwidth

The data transfer of the client averages about 200KB per second according to the representative testing upon 100 concurrent users. The first time the client is accessed, all of the static content like images and style sheets are downloaded. After the initial file load until the web browser cache is emptied or there is an update to the application on the server, the average data transfer goes a little higher again. This is a regular volatility.

Additionally, an application server will frequently communicate with a database server once transactions occur. Also, some reasonable network traffic always occurs on the server in an enterprise environment. The network monitor in the testing theoretically reflects an 8.8Mbps (1.1MB/sec) bandwidth on average jumping in the range of 8Kbps (1KB/sec) to 24Mbps (3MB/sec).

---

**Note:** Over a WAN, the network latency and bandwidth are the major factors affecting the performance. On a 2Mbps T1 line, the effective bandwidth is approximately ~225KB/sec compared to ~10MB/sec on a 100Mbps Ethernet LAN. So, the higher bandwidth would be required on a WAN to support the number of concurrent users expected on the WAN. Conversely, the response time increases if the bandwidth is not sufficient to support all of the concurrent requests over the WAN.

A 100Mbps network will slow down with about 100 concurrent hits, not counting any other network traffic. Therefore, it is highly recommended that all servers are connected to a Gigabit LAN.

---

### Testing Details

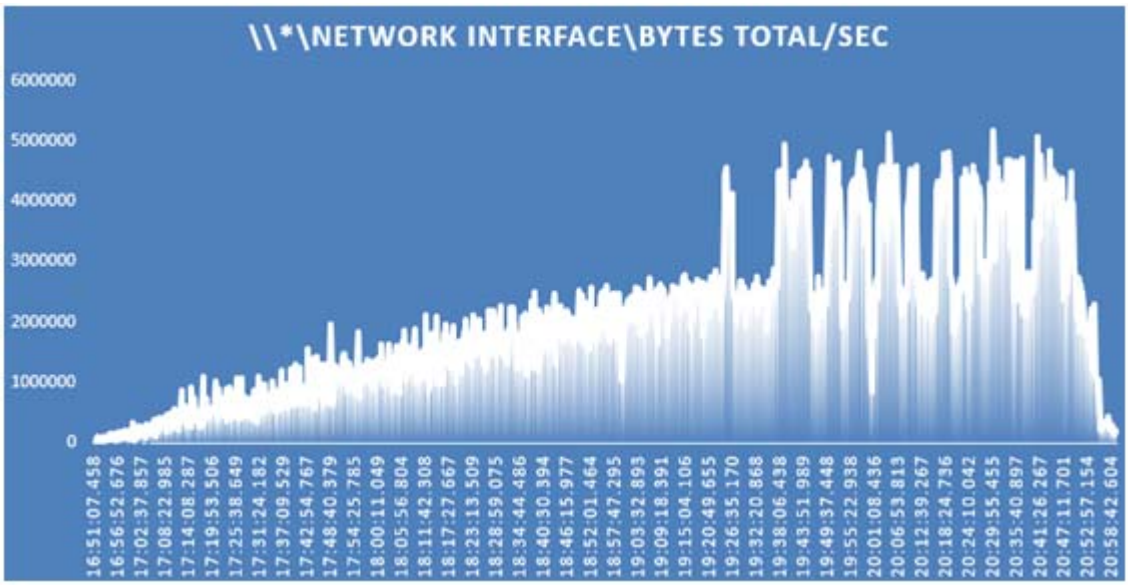
See the graphic below for a sample of network traffic on "Medium" size network.

---

**Note:** A "Small" size produces a similar graphic, but with the peak linearly reduces along with the number of concurrent users.

---

*Figure 2–2 Sample, medium size network traffic*




## File Server

Attachments are stored on the file system with a pointer in the database. This can be stored locally on the same application server, a dedicated server, or on a SAN. The team recommends setting an initial disk space allocation for attachments to 25GB or greater, and further recommends monitoring disk space and adding more when customers have reached a predefined threshold (25% free space remaining).

The typical separated repository handling up to 100 typical logged-in users requires a server with 2 cores CPU, 4GB RAM plus 100Mbps+ bandwidth.

# Scale Notes of Runtime Data

The testing performed a set of investigations to figure out the objective data size ceiling with the optimal user experience so that the customer could understand the reality about the essential bottleneck for such a typical web server hosted application system.

Any type of heavy structure often introduces a performance degradation; what was reflected in frontend is varying degrees of slowness no matter whether there is a hard limit value announced according to functional design. Therefore, any kinds of further enhancement for performance should be done with the purpose of optimizing instead of complete elimination.

In order to reveal the implicit bottleneck and keep everything else working at the designed performance, the team selected representative use cases and used their reports as specimens for analysis. A testing script was designed to continuously add inputs or EA/CS to a specification whose computable data-complexity was 5 until the test went to down. Once the result curve showed an abnormal slope in response time increment, the threshold could be determined.

## Formulation specification vs. inputs example

### Case 1: Add a number of inputs to one step

| Sequence # | Input # | Save Time in Seconds | Total Step Time in Seconds |
|---|---|---|---|
| 1 | 0—5 | 12.50 | 55.62 |
| 2 | 5—10 | 12.82 | 29.30 |
| 3 | 10—15 | 13.67 | 30.64 |
| 4 | 15—20 | 16.50 | 33.59 |
| 5 | 20—25 | 21.07 | 39.74 |
| 6 | 25—30 | 20.20 | 40.19 |
| 7 | 30—35 | 22.24 | 42.14 |
| 8 | 35—40 | 26.23 | 47.60 |
| 9 | 40—45 | 23.54 | 46.82 |
| 10 | 45—50 | 24.72 | 48.40 |
| 11 | 50—55 | 30.56 | 55.36 |
| 12 | > 55 | (Not applicable) | (Not applicable) |

### Case 2: Add a number of inputs to five steps

| Sequence # | Input # | Save Time in Seconds | Total Step Time in Seconds |
|---|---|---|---|
| 1 | 0—5 | 10.31 | 64.87 |
| 2 | 5—10 | 14.33 | 57.30 |
| 3 | 10—15 | 22.87 | 66.33 |
| 4 | 15—20 | 18.17 | 68.72 |
| 5 | 20—25 | 19.45 | 64.85 |
| 6 | 25—30 | 23.07 | 68.71 |
| 7 | 30—35 | 23.99 | 72.97 |
| 8 | > 35 | (Timeout) | (Timeout) |

As a formulation specification should be considered the most complex object in PLM for Process, the testing set 30 seconds as the response time (including both server-responding and client-rendering) limit under the user experience expectation. As the above testing results show, 35 inputs versus 1 formulation was a reasonable threshold recommendation for regular usage. Any scaling of data going beyond that may introduce many kinds of unexpected exceptions.

## Material Specification vs. ExtData

### Case 1: Add a number of extended attributes to one material

| Sequence # | Input # | Save Time in Seconds | Total Step Time in Seconds |
|---|---|---|---|
| 1 | 150—155 | 1.20 | 8.40 |
| 2 | 155—160 | 1.25 | 8.58 |
| 3 | 160—165 | 1.19 | 8.11 |
| 4 | 165—170 | 1.06 | 7.85 |
| 5 | 170—175 | 1.06 | 7.76 |
| 6 | 175—180 | 1.74 | 8.65 |
| 7 | 180—185 | 1.64 | 8.58 |
| 8 | 185—190 | 1.12 | 8.19 |
| 9 | 190—195 | 1.08 | 8.07 |
| 10 | 195—200 | 1.10 | 7.88 |
| 11 | > 200 | (Not applicable) | (Not applicable) |

### Case 2: Add a number of custom sections to one material

| Sequence # | Input # | Save Time in Seconds | Total Step Time in Seconds |
|---|---|---|---|
| 1 | 0—5 | 3.31 | 27.74 |
| 2 | 5—10 | 2.94 | 23.84 |
| 3 | 10—15 | 3.38 | 26.08 |
| 4 | 15—20 | 2.67 | 27.64 |
| 5 | 20—25 | 3.45 | 33.48 |
| 6 | 25—30 | 7.17 | 23.05 |
| 7 | > 30 | (Not applicable) | (Not applicable) |

With the testing results, it looks like the tests are not able to reach any explicit limitation under specific performance expectations. However, assuming there are about 200 extendedattributes (or 30 custom sections) dynamically displaying in a single page, the browser stability as well as the element-rendering are going to be the new bottleneck against usability. Capacity planning testing cannot record the time consumed on remote client side; instead a part of testing behaviors used a manual monitor. So the potential threshold recommendation was set as 200 extended attributes or 30 custom sections in this case, and the response time (including both server-responding and client-rendering) on the client-side was around 30 seconds on average.

# 3

# Performance Tips

To ensure proper performance of PLM for Process, knowledge of the end to end product architecture is important because performance issues can be introduced at any level. This section outlines a list of best practice steps to help keep the PLM for Process deployment performing well.

## Database Server

Maintaining a tuned database is a key component in ensuring the long term health of the overall PLM for Process deployment. Though the applications take advantage of caching when possible, PLM for Process still relies heavily on database interaction. For this reason, please follow the below recommendations in addition to the “Best Practice” guidelines as recommended by the corresponding database vendor.

**Note:** Many kinds of database tuning introduce some level of risk. Do a full backup prior to the implementation, and involve your DBA as necessary.

## Fragmentation

### SQL Server

When data is inserted into, deleted from, or updated in a SQL server table, the indexes defined on that table are automatically updated to reflect those changes. As the indexes are modified, the information stored in them becomes fragmented, resulting in the information being scattered across the data files. When this occurs, the logical ordering of the data no longer matches the physical ordering, which can lead to a deterioration of query performance.

To fix this problem, indexes must be periodically reorganized or rebuilt (defragmented) so the physical order of the leaf-level pages matches the logical order of the leaf nodes. This means that DBA should analyze the indexes periodically to determine whether they've become fragmented and the extent of that fragmentation. From there, the DBA can either reorganize or rebuild the affected indexes, depending on the results of the analysis.

As a helpful utility, the product has provided a stored procedure in the database called "XSP_AGILE_PLM4P_DEFRAG_INDEXES". This stored procedure will analyze the fragmentation of the indexes, generate SQL statements to rebuild or reorganize the over-fragmented ones, and then execute those statements. There are two values that can be adjusted based on the preference of a knowledgeable DBA. These are the percent average fragmentation and the actual number of fragments per index. These

values help determine which indexes should be defragmented and whether they should be rebuilt or reorganized.

This procedure or another defragmentation routine should be executed on a regular basis to ensure the fragmentation does not affect performance.

### Oracle

The Oracle database handles fragmentation internally so this topic is not applicable.

## Optimizer Statistics

Database optimizers rely on statistics to determine the best plan for each query. It is important that these statistics are up to date.

### SQL Server

Statistics are gathered automatically in SQL server so this topic is not applicable.

### Oracle

The process to maintain updated statistics in Oracle will vary depending on the version. Our product team recommends that customers follow the advice of its on-site DBA, or consult the Oracle database documentation when determining the plan.

## Tablespace and Data File Configurations

While the proper sizing of extents minimizes dynamic extensions in the same segments, disk I/O contention within the same logical tablespace or physical data file can also be harmful.

Customers can improve disk I/O performance by spreading the I/O burden across multiple disk devices. And it is always advisable to use more disks.

Also, a reasonable size growth plan should be thought of as a mandatory guarantee to overall functionalities and performance.

### SQL Server

Direct data file assignment is the preferred tablespace concept. Optionally you can set up a data file as 10% auto-growth with a specific maximum size. Further, consider assigning multiple data files which are located at different storages so that backend balance can be made. See the assignment example below:

***Figure 3–1 SQL server example***

| File Type | Filegroup | Initial Size (MB) | Autogrowth / Maxsize | | Path |
|---|---|---|---|---|---|
| ROWS Data | PRIMARY | 256 | By 10 percent, Unlimited | ... | C:\DB\SQLServer\ |
| LOG | Not Applicable | 1 | By 10 percent, Unlimited | ... | C:\DB\SQLServer\ |
| ROWS Data | PRIMARY | 256 | By 10 percent, Unlimited | ... | D:\DB\SQLServer\ |

## Oracle

With the similar thought as SQL Server I/O distribution, it would be better to manage the DATA tablespace and TEMP tablespace separately. For instance:

1. Create a dedicated DATA tablespace for PLM for Process with a specific growing plan.

| TABLESPACE_NAME | SIZE(MB) | USED(MB) | FREE(MB) | FREE% | SEGMENT_SPACE_MANAGEMENT |
|---|---|---|---|---|---|
| PLM4P_DATA | 3584 | 3400.19 | 183.81 | 5.13 | AUTO |

| FILE_NAME | TABLESPACE_NAME | SIZE(MB) | AUTOEXTENSIBLE | MAX_SIZE(MB) | increment_by(MB) |
|---|---|---|---|---|---|
| /oracle/oradata/PLM4P/PLM4P_DATA.DBF | PLM4P_DATA | 3584 | YES | 8192 | 128 |

2. Create a dedicated TEMP tablespace for PLM for Process, especially in a shared database environment. Normally, sorting, grouping, indexing, joins and so on may strongly depend on the TEMP tablespace capacity.

| TABLESPACE_NAME | SIZE(MB) | AUTOEXTENSIBLE | MAX_SIZE(MB) | INCREMENT_BY(MB) | USED(MB) |
|---|---|---|---|---|---|
| PLM4P_TEMP | 256 | YES | 4096 | 128 | 255 |

3. Additionally, a proper PGA size along with a set of analyzed "XXX_AREA_SIZE" would reduce the utilization of the TEMP tablespace. That also helps in performance since memory is much faster than disk.

As far as the I/O distribution resolution upon the Oracle database, a common recommendation for "Small" and "Medium" follows:

**One-Disk vs. Small Size** A one-disk configuration can result in disk I/O contention when the storage device is a single physical disk. As both database size and usage increase, performance can decline. A one-disk configuration is best for a demonstration, preproduction, and testing environment or where the database files are stored on a RAID array or other storage subsystem with built-in striping and mirroring. The configuration can be implemented as shown below.

| | |
|---|---|
| Disk A | ORACLE_HOME |
| | SYSTEM |
| | UNDO |
| | TEMP |
| | USERS |
| | PLM4P_DATA |
| | PLM4P_INDEX |
| | PLM4P_TEMP |

**Two-Disk vs. Medium** A two-disk configuration is best for a medium database. To eliminate potential I/O contention, DATA and INDEX data files are on separate disks.

| | |
|---|---|
| Disk A | ORACLE_HOME |
| | SYSTEM |
| | UNDO |
| | PLM4P_DATA |
| Disk B | TEMP |
| | USERS |
| | PLM4P_INDEX |
| | PLM4P_TEMP |

# Indexes

PLM for Process comes out of the box with indexes to account for common usage of the application. Since the need for indexes greatly depends on certain characteristics that are specific to each deployment, such as row counts, ongoing analysis will need to be performed to detect the need for additional ones.

It is strongly recommended that the executing SQL is monitored and the proper analysis is performed for index maintenance.

Any customized indexes will not be removed during upgrades and it is recommended to use a naming convention different than the one used for the core indexes, especially when customer is creating customized tables.

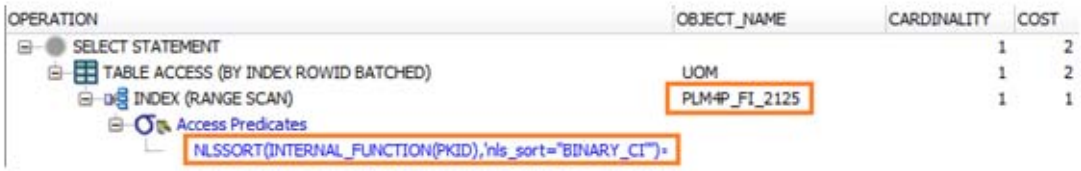
> **Note:** PLM for Process took the case-insensitive string-matching policy as the default behavior. SQL server supports that by default. However, Oracle needs an extra pre-setting prior to executing business SQL statements.

```
ALTER SESSION SET NLS_SORT=BINARY_CI;
ALTER SESSION SET NLS_COMP='LINGUISTIC';
ALTER SESSION set NLS_TIMESTAMP_FORMAT = 'MM/DD/RRRR HH24:MI:SS';
ALTER SESSION set NLS_DATE_FORMAT = 'MM/DD/RRRR HH24:MI:SS';
```

In v6.2.0.0 or above, these settings were delivered with an out-of-box database trigger named “NLS_SETTINGS”, which would be auto-performed when a database user is logging on the database.

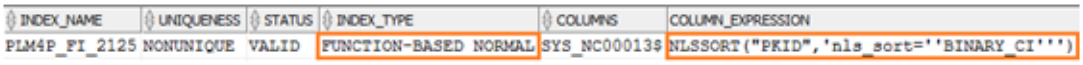
In other words, the settings become a mandatory precondition when executing any business SQL statements as well as troubleshooting. This drives Oracle to use functional-based PKID (or other Foreign Keys) indexes instead of normal indexes. Below is an expected plan when executing a simple index-based query:

```
SELECT * FROM UOM WHERE PKID='21251C1A1522-134D-45a4-BC6B-6E3DEA51f740'
```




And a sample to create such an index:

```
/*BeginContiguousBlock*/
DECLARE CNT number;
BEGIN
  SELECT COUNT(1) INTO CNT FROM user_indexes WHERE table_name = UPPER('UOM')
and index_name = UPPER('PLM4P_FI_2125');
  IF (CNT = 0) THEN
    EXECUTE IMMEDIATE 'CREATE UNIQUE INDEX PLM4P_FI_2125 ON UOM(NLSSORT(PKID,
''NLS_SORT=BINARY_CI''))';
  END IF;
END;
/*EndContiguousBlock*/
```




# Caching and Compression

Using caching and compression is the most effective way to reduce page response time. There is a very large performance impact on sites with low bandwidth/high latency. It is strongly recommended that all sites use the appropriate levels of caching and compression.

Although PLM for Process provided links to set up caching and compression within Microsoft IIS, both can be implemented or negated at an upstream device in the architecture. The product team recommends customers discuss options for caching and compression within its IT organization.

Using a utility like Internet Explorer/Chrome developer tools (F12) as a client to verify caching and compression are set up and working correctly is also recommended.

## Caching

Caching content will drastically reduce the amount of round trips between the client and the server. By default, IIS caches the following extensions.

- .css
- .js
- .jpg
- .gif
- .png
- .ico

Refer to the following article for more details:

https://technet.microsoft.com/en-us/library/cc732475(v=ws.10).aspx

### Compression

Compression reduces the size of the data passed between the server and the client browser. Because compression algorithms are so efficient, the team recommends all data be compressed.

For instructions on how to enable compression in IIS, please refer to:

http://technet.microsoft.com/en-us/library/cc754668(v=ws.10).aspx

## DB Settings Reporting Utility for Oracle

The utility "_DBInfo_ORCL.aspx" is a single page program for .NET runtime released along with the product package since v6.2.0. It collects Oracle database information which can be used for further research and remote troubleshooting.

Please review the document about the usage at:
https://community.oracle.com/docs/DOC-994455

**Note:** Make sure the connecting account has been granted with "grant select any dictionary to <DB Username>" prior.

## Performance Logging

Performance Logging was included in the official release. The feature is only enabled according to communication with Oracle support when encountering any interaction-related performance problems. The utility generates a detailed time-counting report about representative activities performed on the system. It is greatly helpful to sustaining progress, by assisting in figuring out the module name and shortening the issue-reproducing lifecycle. Please utilize that under guidance from Oracle support, or directly refer to the "Using Performance Logging" chapter of the *Agile Product Lifecycle Management for Process Administrator User Guide*.

**Note:** When this feature is enabled, there is little impact to the user's experience, however we recommend only enabling it while troubleshooting.

## CPU/Memory Consumption and Other Indicators

Under sizing the server capacity for the PLM for Process deployment will have a direct impact on performance. However, because of the complexity of a customer's environment, it's mostly impossible to imagine a simple positive correlation between hardware and performance. Even if a customer has followed the recommendations in this guide initially, there may be a need to launch a termly monitor session until every indicator as well as the size-extending plan become stable.

## Usage

As user adoption of the system grows, the amount of load put on the system could increase past the initial expectations. If CPU utilization, memory consumption, disk busyness or network traffic increased beyond the optimal range, the user experience will degrade due to slow response time.

These can be monitored by various tools, one being the "Windows Performance Monitor". Instructions can be found at:

http://technet.microsoft.com/en-us/library/cc749115.aspx

The team recommends monitoring the following:

| Target | | Counter | Limit Comments |
|---|---|---|---|
| Application Server | CPU | Processor / % Processor Time / _Total | <= 70% |
| | Memory | Process / Private Bytes / _Total<br>Process / Private Bytes / w3wp (Application Pool)<br>Process / Private Bytes / w3wp#1 (Application Pool)<br>Process / Private Bytes / w3wp#2 (Application Pool)<br>Process / Private Bytes / w3wp#... (Application Pool) | <= 80% for _Total<br><= (Initial Load * 2) for each Application Pool |
| | Disk | Physical Disk / Current Disk Queue Length / _Total | <= 5 (The lower, the better) |
| | Network | Network Adapter / Bytes Total per sec / (Connected Network Adapter) | <= Bandwidth * 50% |
| Database Server | CPU | Processor / % Processor Time / _Total | <= 70% |
| | Memory | Process / Private Bytes / _Total<br>Process / Private Bytes / oracle (Oracle Instance) or sqlservr (SQL Server Instance) | <= 80% for _Total |
| | Disk | Physical Disk / Current Disk Queue Length / _Total | <= 5 (The lower, the better) |
| | Network | Network Adapter / Bytes Total per sec / (Connected Network Adapter) | <= Bandwidth * 50% |

# Troubleshooting Performance Issues

There are many possible causes of performance issues. It is best to eliminate as many variables as possible, in a methodical way, before calling Oracle Support. Below is a list of items to go through.

All tests should be performed after a user executes the exact use case at least two times. The first time through will be slower and is not considered an issue by Oracle because of the concern to caching work.

| | Action | Description |
|---|---|---|
| 1 | Isolate performance use cases | Determine the exact pages, actions and data that are involved with the performance issue. |
| 2 | Ensure caching and compression is enabled | Make sure the caching and compression recommendations in this document are followed. |
| 3 | Eliminate custom code | Eliminate all custom code by testing with a vanilla installation. If this improves performance it is likely the issue is in the custom code. |
| 4 | Add database indexes if needed | Monitor and analyze all SQL statements that are executed for requests that are considered slow. Have a DBA determine if new indexes are needed. |
| 5 | Monitor the hardware for capacity limits | Determine if the issue is due to improper sizing of the hardware by performing the test with a single user on the system. If the performance issue cannot be reproduced with a single user then it is likely a hardware capacity issue.<br>If the issue only occurs under normal production load, then monitor the application data server for overloading resources.<br>Check that the IIS application pools are configured according to the recommendations in this document. |
| 6 | Eliminate network hardware | Determine if the issue is due to hardware on the network such as load balancers, reverse proxies or firewalls by testing in an environment that does not have any of these between the application server and the client. |
| 7 | Check network bandwidth and latency | Determine if the issue is due to a poorly performing network. |
| 8 | Monitor log files | Monitor all log files for error conditions that might cause additional overhead. This includes PLM for Process' event logs as well as all third party log files such as SSO, LDAP, web logs, and database. |
| 9 | Oracle Support | If you have determined that the issue is within the Oracle code, then open a service request with Oracle Support and provide them as much detail as possible, including the outcome of all the above tests. Oracle Support will need to have the exact steps taken to reproduce the performance issue along with an obfuscated database. |

# (Experimental) Index Utility for Oracle

For index maintenance work, PLM for Process released an out of the box package named “INDEX_UTIL” for Oracle database with version 6.2.1.0.

**Note:** This is a fast indexing utility that can be used when customers encounter specific performance issues during sustaining progress.

| INDEX_UTIL | | | |
|---|---|---|---|
| **Procedure** | **Parameters** | **Perform Changes** | **Description** |
| createIndex | p_tableName varchar2, p_indexName varchar2, p_columnNames varchar2, p_allowRebuild number default 0, p_isUnique number default 0 | Yes | The procedure performs a real change on the specified database. It scans the existing index by both index name and column definition. “p_allowRebuild” helps determine if the old (existing) index (as well as corresponding constraints) would be removed before creating. The procedure could intelligently determine according to basic PLM for Process rule if a NLSSORT function-based index is needed. |
| scanDOBasedIndexes | (None) | No | The procedure scans all meta-data definitions which explicitly need an index according to PLM for Process object-relationship models, such as: primary keys, foreign keys, join table keys.<br>It populates a temp table with caught items, and then shows the step guide in DBMS_OUTPUT. |
| generateCreateIndexScript | (None) | No | The next step to “scanDOBasedIndexes”, which helps generate DB scripts to create those indexes. However it will not actually execute the statements. The step guide will be shown in DBMS_OUTPUT. |
| normalizeIndexForJoinTable | p_tableName varchar2, p_indexName varchar2, p_columnNames varchar2 | Yes | The procedure performs a real change on the specified database. It normalizes indexes in join-relationships. As foreign keys in join tables, they must be not-nullable, sequential and have an explicit default value so that indexes over them can push performance forward. This may trigger an index rebuild. |
| scanJoinTableIndexes | (None) | No | The procedure scans all join-table indexes which did not match the performance recommendation.<br>It populates a temp table with caught items, and then shows the step guide in DBMS_OUTPUT. |
| generateNormalizeIndexScript | (None) | No | The next step to “scanJoinTableIndexes”, which helps to generate DB scripts to normalize those indexes but will not actually execute the statements. The step guide will be shown in DBMS_OUTPUT. |

Here is a sample to create a formal custom index:

```
-- For PKID
INDEX_UTIL.createIndex('npdProjects','PLM4P_FI_3202','PKID',0,1);
-- For Join Table
INDEX_UTIL.createIndex('UserGroupJoin','PLM4P_JN_Useroin_fkGups_
fkUers','fkGroups,fkUsers',0,0);
-- For Foreign Key
INDEX_UTIL.createIndex('npdActivitySections','PLM4P_FK_3331_
fkActivity','fkActivity',0,0);
```

# (Experimental) Reduce Hard Parsing for Oracle

The bind-variables are important in SQL performance because they allow Oracle to reduce the amount of 'hard parsing'. With a typical object-oriented implement over entity mappings to a database table, the SQL statements are often literally organized by code against the advantage of bind-variables, especially when resolving the literal SQL statements for traversing heavy collections.

Oracle parameter “CURSOR_SHARING” determines what kind of SQL statements can share the same cursors, which helps to improve performance of this kind of situation. However it shouldn’t be considered as a wonder pill unless the performance-tracker logs reflect plenty of SQL statements with a similar format. In some provable cases recorded by the sustaining team, setting “CURSOR_SHARING = FORCE” sometimes introduces a dramatic performance improvement globally. Also, it is able to reduce the CPU consumption of the database server when high concurrency happens.

The available options are being listed in the table:

| CURSOR_SHARING | |
|---|---|
| EXACT | [Default] Only allows statements with identical text to share the same cursor. |
| FORCE | Forces statements that may differ in some literals, but are otherwise identical, to share a cursor, unless the literals affect the meaning of the statement. |
| SIMILAR | Causes statements that may differ in some literals, but are otherwise identical, to share a cursor, unless the literals affect either the meaning of the statement or the degree to which the plan is optimized. |

# A

# Capacity Testing Declarations

## Data Complexity Score

Data complexity is an important factor affecting overall performance. Capacity testing used the following pre-defined data complexity. You can use the EXCEL template located here: https://community.oracle.com/docs/DOC-998885 to determine your own data complexity score.

**Note:** If the gap between your score and capacity planning goes beyond ±50%, the corresponding performance expectation should be re-scoped.

***Figure 3–2 Example A: Formulation specification data complexity in capacity testing database***

| Specification | Subordinate Data | Qty. Range | Qty. AVG | Weight Factor | Score |
|---|---|---|---|---|---|
| Formulation | Summary Tab | (1) | 1 | 0.02 | 0.02 |
| | Summary Tab/Facility Information | (3, 5) | 4 | 0.03 | 0.12 |
| | Ext Data Tab | (1, 3) | 2 | 0.20 | 0.40 |
| | Related Specs Tab | (3, 5) | 4 | 0.10 | 0.40 |
| | Supporting Documents Tab | (3, 5) | 4 | 0.10 | 0.40 |
| | Process Tab/Steps | (5, 10) | 8 | 0.40 | 3.20 |
| | Process Tab/Inputs per Step | (2, 4) | 3 | 0.10 | 0.30 |
| | Activities | (3, 5) | 4 | 0.05 | 0.20 |
| | | | | Final Score: | 5.04 |

***Figure 3–3 Example B: Material specification data complexity in capacity testing database***

| Specification | Subordinate Data | Qty. Range | Qty. AVG | Weight Factor | Score |
|---|---|---|---|---|---|
| Material | Summary Tab | (1) | 1 | 0.01 | 0.01 |
| | Formulation Tab | (5, 10) | 8 | 0.30 | 2.40 |
| | Nutrition Tab | (5, 10) | 8 | 0.05 | 0.40 |
| | Compliance Tab | (3, 5) | 4 | 0.05 | 0.20 |
| | Ext Data Tab | (3, 5) | 4 | 0.20 | 0.80 |
| | Related Specs Tab | (3, 5) | 4 | 0.10 | 0.40 |
| | Supporting Documents Tab | (3, 5) | 4 | 0.10 | 0.40 |
| | Reference Tab | (3, 5) | 4 | 0.10 | 0.40 |
| | Activities | (3, 5) | 4 | 0.05 | 0.20 |
| | LIOs | (3, 5) | 4 | 0.04 | 0.16 |
| | | | | Final Score: | 5.37 |

# Testing Environment

***Figure A–1 Testing environment***

| Size | Server Type | Details | |
|---|---|---|---|
| Small | Application | Virtual Machine | Yes |
| | | OS | Windows Server 2012 Standard |
| | | Web Server | IIS 8 |
| | | CPU | 2.93GHz, 2 Cores |
| | | Memory | 8 GB |
| | | Network Interface | 1 Gbps |
| | Database | Virtual Machine | Yes |
| | | OS | Windows Server 2008 R2 Enterprise |
| | | DB Server | Oracle 12c Enterprise Edition 12.1.0.2.0 - 64bit, No CDB |
| | | CPU | 2.93GHz, 2 Cores |
| | | Memory | 8 GB |
| | | Network Interface | 1 Gbps |
| Medium | Application | Virtual Machine | Yes |
| | | OS | Windows Server 2012 R2 Standard |
| | | Web Server | IIS 8.5 |
| | | CPU | 2.93GHz, 4 Cores |
| | | Memory | 16 GB |
| | | Network Interface | 1 Gbps |
| | Database | Virtual Machine | Yes |
| | | OS | Windows Server 2008 R2 Standard |
| | | DB Server | Oracle 12c Enterprise Edition 12.1.0.2.0 - 64bit, No CDB |
| | | CPU | 2.93GHz, 4 Cores |
| | | Memory | 16 GB |
| | | Network Interface | 1 Gbps |
| Large | Application | Refer to Medium | (Application Balance) |
| | Database | Refer to Medium | (Database Isolation Balance) |

**Note:** The majority of testing used “Medium” data complexity, unless otherwise stated.

# Testing and Threshold Definitions

The load testing is performed using scripts that simulate users. The simulation mainly consists of READ and WRITE actions. Users' ThinkTime (the time-delay between two click actions) was pre-defined based on the grid below.

| Action | Percentage | ThinkTime Range in Seconds |
|---|---|---|
| READ | 80% | 20—30 |
| WRITE | 20% | 40—60 |

The test starts with one virtual user, then every minute another virtual user would join until the cumulative time whose average response time exceeded 5 seconds took up 10% of the whole testing period. The number of virtual users is what was used as the maximum number of concurrent users under the configuration.

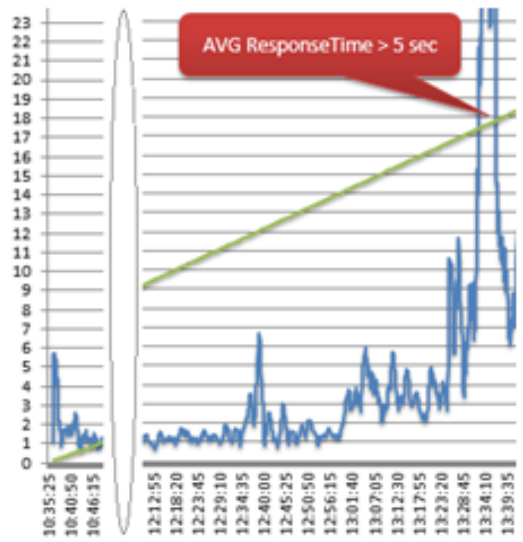
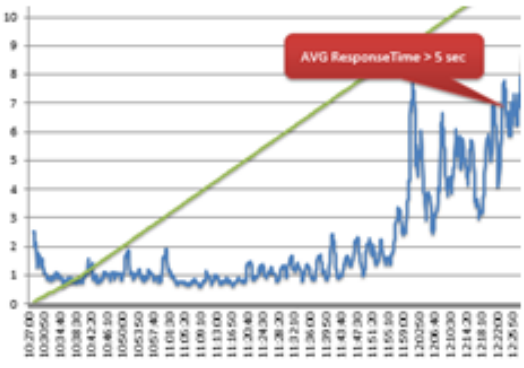
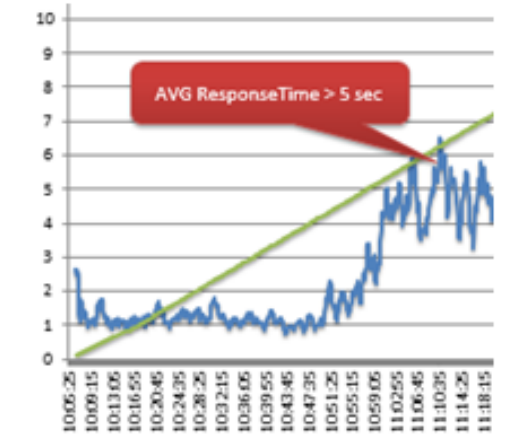
> **Note:** The "Response time > 5 seconds" case normally implies a user experience degradation, but not the service termination. That would be acceptable in a load testing once response time occasionally came up to 60 seconds upon some hard case.

## Testing Details

ThinkTime, ResponseTime and Concurrency: any of those would essentially leverage others in the testing. The ThinkTime parameter often shows the expectation of concurrency pressure. At the testing conclusion, the concurrency capacity shows a linear relationship to ThinkTime. And Concurrent Users Threshold always comes up along with a high CPU consumption of application server. The tests did help to determine the set of thresholds by formally adjusting the range of ThinkTime with the same READ/WRITE percentage.

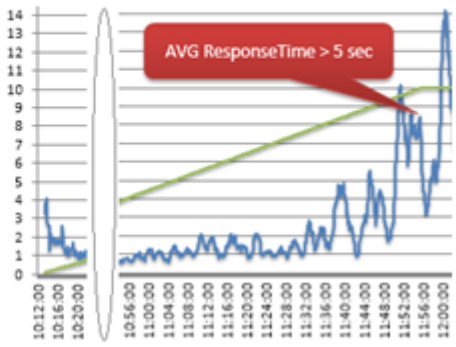
> **Note:** Session #1 upon "Medium" environment is determined as the recommended baseline of this guide by thinking about the gradient over efficiency, stability, availability and usability.

***Figure A–2 Testing details, medium environment***

| Session # | Think Time Range | Duration in Minutes | Concurrent Users Threshold | Graph |
|---|---|---|---|---|
| 1 | 20 - 30 READ, 40 - 60 WRITE | 183 | 180 |   |
| 2 | 10 - 15 READ, 20 - 30 WRITE | 120 | 117 |   |
| 3 | 5 - 8 READ, 10 - 15 WRITE | 70 | 69 |   |

Then test on “Small” environment to figure out the corresponding thresholds.

***Figure A–3 Testing details, small environment***

| Session # | Think Time Range | Duration in Minutes | Concurrent Users Threshold | Graph |
|---|---|---|---|---|
| 1 | 20 - 30 READ, 40 - 60 WRITE | 108 | 100 |  |

# B

# Application Pool Distribution

This appendix contains guidelines for application pool distribution.

## Application Pool Distribution

Agile PLM for Process is designed to be hosted by Microsoft Internet Information Services (IIS). An IIS application pool is a grouping of URLs that is routed to one or more worker processes. Because application pools define a set of web applications that share one or more worker processes, they provide a convenient way to administer a set of web sites and applications and their corresponding worker processes. Application pools significantly increase both the reliability and manageability of a web infrastructure. Again, a reasonable application pool distribution makes much sense to leverage the performance outlines.

The following is applicable to all implementation sizes and serves as an initial recommendation for how to distribute customer applications in a production environment. With monitoring in place, it's recommended to adjust these as needed.

Initially, an application pool would auto-start once a corresponding website is being visited. As a regular environment, 5000MB available memory is the minimum to adapt a typical initialization.

***Table 3–1 Application pool distribution***

| Application Pool | Applications | Comments |
|---|---|---|
| PLM4P_GSM | GSM | GSM is a heavily used application and has the potential to consume the most memory. Therefore, it should always be isolated in its own application pool. |
| PLM4P_GSMView | GSMView | GSMView is a read-only copy of GSM with the purpose of offloading certain memory intensive operations. It may not be required to be in an isolated application pool, but should be separated from GSM. |
| PLM4P_MAIN | Portal, UGM, REG, CSSPortal, DRL, DRLService | Shared light applications could be handled by one application pool. |
| PLM4P_PDM | SCRM, WFA, Integration, Reporting, ReadyReports | Consider distributing as the product bundle. |
| PLM4P_NPD | NPD | NPD is a heavily used application and has the potential to consume more memory than other applications. Therefore, it should be isolated in its own application pool. |

*Table 3–1 Application pool distribution*

| Application Pool | Applications | Comments |
|---|---|---|
| PLM4P_FC | CACS, OPT, PQS | Consider distributing as the product bundle. |
| PLM4P_PQM | PQM, SupplierPQM | Consider distributing as the product bundle. |
| PLM4P_PSC | SupplierPortal, eQ | Consider distributing as the product bundle. |

## Testing Details

The following is the initial memory footprint observed for each application pool in the recommended application distribution. This represents loading the application with a single user.

**Note:** The values were theoretically made up of application initial objects and data caches as well as the only user session data. Avoid absolutely equating the value to a single user session. Instead, please review following table about user session calculation.

*Table 3–2 User session calculation*

| Sequence | Application Pool | Memory (MB) |
|---|---|---|
| 1 | PLM4P_MAIN | 757 |
| 2 | PLM4P_GSM | 1137 |
| 3 | PLM4P_PDM | 1097 |
| 4 | PLM4P_PQM | 897 |
| 5 | PLM4P_NPD | 991 |
| 6 | PLM4P_PSC | 855 |
| / | PLM4P_FC | (Not applicable) |
| / | PLM4P_GSMView | (Not applicable) |