

Updated for V2.1

Best Practices in IT Operations for Oracle Utilities Customer Care and Billing/Enterprise Tax Management

*An Oracle Tax and Utilities Global Business Unit White
Paper
September 07*

Best Practices in IT Operations for Oracle Utilities Customer Care and Billing/Enterprise Tax Management

Purpose of this White Paper.....	4
Introduction	4
Installation Best Practices	5
Read the Installation Guide.....	5
Ensure the prerequisites are installed.....	5
Environment Practices.....	5
Using multiple administrators	6
General Best Practices	7
Limiting Production Access	7
Removal of Staging Records	7
Using Archiving to purge data.....	9
Regular Collection and Reporting of Performance Metrics	10
Respecting Record Ownership	10
Backup of Logs	11
Post Process Logs.....	11
Check Logs For Errors	12
Optimize Operating System Settings	12
Optimize connection pools	13
Read the available manuals.....	15
Implementing Industry Processes	19
Using Automated Test Tools.....	20
Custom Environment Variables or JAR files	21
Backup and Recovery.....	22
Help and AppViewer can be used standalone.....	23
Re-Register only when necessary.....	24
Secure default userids.....	24
Consider different userids for different modes of access	25
Don't double audit.....	26
Use Identical Machines	27
Regularly restart machines	27
Avoid using direct SQL to manipulate data.....	27
Client Computer Best Practices.....	28
Make sure the machine meets at least the minimum specification.....	28
Internet Explorer Caching Settings.....	28

Clearing Internet Explorer Cache	29
Optimal Network Card Settings	29
Network Best Practices	30
Ensure legitimate Network Traffic	30
Regularly check network latency.....	30
Web Application Server Best Practices	31
Make sure that the access.log is being created.....	31
Examine Memory Footprint	32
Optimize Garbage Collection	34
Turn off Debug.....	34
Load balancers.....	34
Preload or Not?.....	36
WebSphere: Native or SPL?.....	38
Hardware or software proxy	38
What is the number of Web Application instances do I need?.....	39
Configuring the Client Thread Pool Size	39
Defining external LDAP to the Web Application Server.....	42
Synchronizing LDAP for security	43
Routine Tasks for Operations.....	46
Typical Business Day.....	47
Appropriate use of AppViewer.....	48
Clustering or Managed?	49
Allocate port numbers appropriately	51
Business Application Server Best Practices (V1.x only)	52
Make the Business Application more traceable (V1.x only).....	52
Turn on Performance Tracking (V1.x only)	53
Optimize the Tuxedo Configuration Settings (V1.x only).....	53
Use Tuxedo TMIB for detailed monitoring (V1.x only).....	54
Multiple JSL processes (V1.x only)	55
Optimize Server/Service distribution for site (V1.x only).....	55
Background Processing Best Practices.....	57
Optimize the schedule.....	57
Multi-threading guidelines	58
Use a third party batch scheduler	60
Database Best Practices	61
Resolve SQL Errors in Batch Jobs/Tuxedo/Hibernate.....	61
Regularly Calculate Database Statistics.....	61
Ensure I/O is spread evenly across available devices	62
Use the Correct NLS settings (ORACLE).....	62
Consider Partitioning	62
Rebind regularly (DB2)	63
Building the Data Model.....	63
Why is there no referential integrity built into the database?.....	63
Building the Data Model	64

Best Practices in IT Operations for Oracle Utilities Customer Care and Billing/Enterprise Tax Management

The recommendations in this document are based upon experiences from various Customer Care and Billing/Enterprise Tax Management sites and internal studies, which have benefited from implementing the practices outlined in the document.

PURPOSE OF THIS WHITE PAPER

This white paper outlines the common and “best” practices used by IT groups at Oracle Utilities Customer Care and Billing/Enterprise Tax Management sites and ORACLE internal studies, around the world, that have benefited sites in some positive way. This information is provided to guide other sites in implementing or maintaining Customer Care and Billing/Enterprise Tax Management.

While all care has been taken in providing this information, implementation of the practices outlined in this document may NOT guarantee the same level of (or any) improvement. Some of these practices may not be appropriate for your site. It is recommended that each practice be examined in light of your particular organizational policies and use of the product. If the practice is deemed beneficial to your site, then consider implementing it. If the practice is not appropriate (e.g. for cost and other reasons), then it should not be considered.

This whitepaper covers both V1.x and V2.x of the Customer Care and Billing/Enterprise Tax Management products. Where advice is applicable to a particular version of the product, a specific reference to that version is displayed.

INTRODUCTION

Implementation of Customer Care and Billing/Enterprise Tax Management at any site introduces new practices into the IT group to maintain the health of the system and provide the expected service levels demanded by the business. While configuration of the product is important to the success of the implementation (and subsequent maintenance), adopting new practices can help ensure that the system will operate within acceptable tolerances and support the business goals.

This white paper outlines some common practices that have been implemented at Customer Care and Billing/Enterprise Tax Management sites, around the globe, that have proven beneficial to that site. They are documented here so that other sites may consider adopting similar practices and potentially deriving benefit from them as well.

INSTALLATION BEST PRACTICES

During the initial phases of an implementation, a copy of the Customer Care and Billing/Enterprise Tax Management product will need to be installed. During the implementation a number of copies of additional copies will be installed, including production. This section outlines some practices that customers have used to make this process smooth.

Read the Installation Guide for the version you are installing or upgrading to ensure smoother installation of Customer Care and Billing/Enterprise Tax Management.

Read the Installation Guide

One of the most important pieces of advice in this document to implement is to read the installation guide. It provides valuable information about what needs to be installed and configured as well the order of the installation. Failure to follow the instructions can cause unnecessary delays to the installation.

If you are upgrading to a new version, read the new installation guide as well as it will contain instructions on how to upgrade to the new version as well as details of what has been changed in the new version.

Make sure that you have downloaded and installed the required third party software indicated in the Installation Guide prior to installing Customer Care and Billing/Enterprise Tax Management.

Ensure the prerequisites are installed

When installing there is a number of third party prerequisite software that must be obtained (i.e. downloaded) prior to the actual installation of Customer Care and Billing/Enterprise Tax Management software can commence. Read the Installation Guide and Quick Installation Guide to download and install the prerequisite software prior to installing Customer Care and Billing/Enterprise Tax Management.

Note: For customers who are upgrading, the installation of Customer Care and Billing/Enterprise Tax Management and its related third party software is designed so that more than one version of Customer Care and Billing/Enterprise Tax Management can co-exist.

Environment Practices

When installing Customer Care and Billing/Enterprise Tax Management at a site, each copy of Customer Care and Billing/Enterprise Tax Management is regarded as an “environment” to perform a particular task or group of tasks. Typically, without planning this can lead to a larger than anticipated number of environments. This can have a possible negative “flow on effect” by increasing overall maintenance effort and increasing resource usage (hardware and people), which may in turn cause delays in implementations. Customers to minimize the impact of environments on their implementations have used the following advice:

- At the start of the implementation decide the number of environments to use. Keep this to a minimum and consider “sharing” environments between tasks. Another technique associated with this is to specify an “end date” for each environment. This is the date the environment can be removed from the implementation. This can force “rethinks” on the

Minimizing the number of environments is important to minimize maintenance effort.

number of environments that are to be used at an implementation and may force sharing.

- For each environment, consider the impact on the hardware and maintenance effort including the following:
 - The time and resources it takes to install the environment.
 - The time and resources it takes to keep the environment up to date including application of single fixes, rollups/service packs and upgrades. Do not forget application and management of customization builds.
 - The time and resources to maintain the ConfigLab and Archiving facilities for multiple environments, if used at an implementation. This includes the setup and regular migrations that will be performed.
 - The time and resources it takes to backup and restore environments on a regular basis. In some implementations, having different backup schemes for environments based upon tasks and update frequency for that environment, i.e. more updated = more frequent backup, may provide some savings.
 - The time and resources to manage the disk space for each environment including regular cleanups.

Environments may be setup so that the database can be reduced to a single database instance with each environment having a different schema/owner. This will reduce the memory footprint of the DBMS on the machine but may reduce availability of the database instance is shut down (all environments are affected). For non-Production, most customers create a database instance, for ORACLE sites, for each environment and one database subsystem, for DB2/UDB sites, customers for each environment.

You can give control (Start/Stop/Monitor) to specific teams by creating multiple administrator accounts and installing Customer Care and Billing/Enterprise Tax Management using those accounts.

Using multiple administrators

By default, when installing Customer Care and Billing/Enterprise Tax Management a single administrator account (usually referred to as “**ci ssys**”) is used to install and “own” the product. This is the default behavior of the installation and apart from specifying a different userid than the default “**ci ssys**”, it is possible to use other userids to own all or individual environments.

For example, if the conversion team wishes to have the ability to start, stop and monitor their own environments, you can create another administrator account and install their copies of Customer Care and Billing/Enterprise Tax Management using that userid. This allows the conversion team to control their own environments. If you did not have the ability to use multiple administrators than they may have access to all environments (as you would have to give them access to the “**ci ssys**” account).

One of the advantages of this approach is that you can delegate management of a copy Customer Care and Billing/Enterprise Tax Management to other teams without compromising other environments. Another advantage is that you can quickly identify UNIX resource ownership by user rather than trying using other methods.

The only disadvantage is that to manage all copies of Customer Care and Billing/Enterprise Tax Management you will need to logon to the additional administration accounts that own the various copies.

GENERAL BEST PRACTICES

This section outlines some general practices that have been successfully implemented at various Customer Care and Billing/Enterprise Tax Management sites.

Ensure Production access is secured and valid. Giving direct database access to data can expose your site to security breaches as well as cause potential performance issues.

Limiting Production Access

One of the guiding principles at successful sites is that Production access is restricted to the processing necessary to run their business. This means that other “non-mainstream” work, such as ad-hoc queries, are either very limited or NOT performed on production at all. This may sound logical but a few sites have allowed access to production from inappropriate sources, which has had an adverse impact on performance.

For example, it is not appropriate to allow people access to the production database through ad-hoc query tools (i.e. such as DB2 Control Center¹, SQL Developer, SQL*Plus etc). The “freestyle” nature of these tools can allow a single user to wreak havoc on performance with a single inefficient SQL statement.

The database is not optimized for such unexpected traffic. Removal of this potentially inefficient access can typically, improve performance.

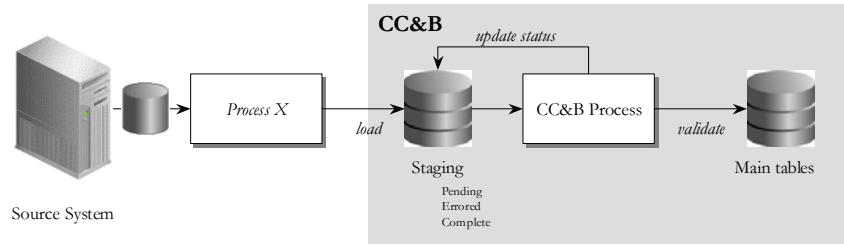
You must remember that when the status of a record in the staging tables used for interfaces becomes “Complete” then it becomes redundant data. The data will be reflected in the main Customer Care and Billing/Enterprise Tax Management tables and is not required in the staging tables anymore. Removal of completed records, on a regular basis, can have storage benefit as well as performance benefit.

Removal of Staging Records

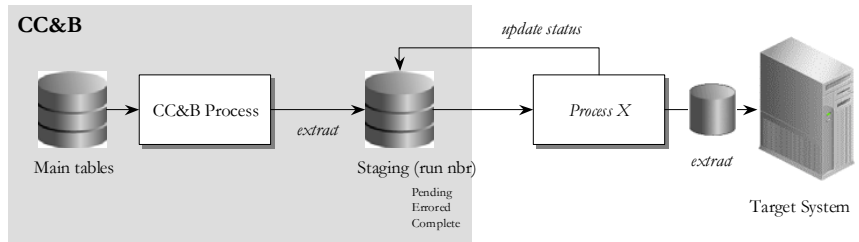
The Customer Care and Billing/Enterprise Tax Management product uses a “staging” concept for most of the major interfaces. This involves a process, known as “Process X”, to load the staging tables and then a base product background process is run to validate and “copy” the valid staging data into the relevant main tables. When records are loaded initially, the status of the records are set to “Pending” indicating they are ready to process. Once the relevant base product background process processes them, then the status is changed to either “Completed” (for valid records) or “Error” (for invalid records). Invalid records can be corrected using the relevant staging online query to manually resolve the error.

¹ Not recommended for dynamic queries only, as DB2 Control Center has an administration component that is useful for DB2 DBAs

This is summarized in the diagram below:



Input Staging



Output Staging

It is assumed that “Completed” staging records are no longer required, after a period of time, as they have been reflected in the main tables. There is no business reason to keep “Completed” staging records after they have been “Completed” for long periods of time.

Regular cleanups of the staging tables to remove completed records will have great performance benefits on interfaces. Successful sites run the provided purge jobs to improve performance and reduce disk space usage.

The table below lists the main entity for the interface, the purge job that is suggested to be executed to remove completed records, the base product background process/component that processes the processes the staging records and the possible performance benefits from deleting records:

Main Entity	Purge Job	Interface Job	Benefit
Bill Charges	BCUP-PRG	BCU1 / BCU2	Billable Charge Upload and staging queries will execute faster.
Field Activities	FAUP-PRG	FACOMPL	Field Activity Upload and staging queries will execute faster.
Meter Reads	MRUP-PRG	MUP1 / MUP2	Meter Read Upload and staging queries will execute faster.
Payments	PYUP-PRG	PUPL	Payment Upload and staging queries will execute faster.

Main Entity	Purge Job	Interface Job	Benefit
To Do	TD-PURGE	Not Applicable	To Do processing will be faster and affects most batch and online processing where To Do's are checked or auto-completed will improve.
XML Staging	XMLUP-PRG	XAI / MPL	MPL will execute faster.

To decide when to run these Purge Jobs and what parameters to pass to them the following is recommended:

- Work out with the business at the site how long they wish to retain the number of completed records. You can stress to them that NO important data is lost in purging completed records as their data is reflected in main tables. The value is used for the **NO-OF-DAYS** batch parameter passed to the job. The value is the number of days not the number of business days (e.g. A value of 14 for **NO-OF-DAYS** means 2 weeks).
- For the To Do Purge job, there are additional parameters to decide the specific To-Do type to purge or ALL (**DEL-TD-TYPE-CD** and **DEL-ALL-TD-SW**). Work with the business to decide if this job is to be run once (for all To Do types) or multiple times for each To-Do Type. Successful customers run it to delete all To Do types to reduce the number of jobs to run.
- Decide the frequency based upon data growth of each table. Ideally these purge process should be run each business day at the end of the nightly batch schedule to keep the optimum but should be run once a week at a minimum.

Using Archiving to purge data

One of the major features of Customer Care and Billing/Enterprise Tax Management is the inbuilt Archiving engine. Typically the engine is associated with moving data from Production to an Archive environment but it also has the feature of purging data instead of moving it.

The Archiving engine is driven by meta-data that decides what to archive/purge (including relationships) and any processing that needs to be performed to maintain integrity.

This means it is possible to define your own purge db-processes for any sets of data you wish to remove within Customer Care and Billing/Enterprise Tax Management and have the archive engine remove the data for you with little or no coding required.

The Archive engine can be used to maintain the health of your system by removing transient or redundant data introduced from interfaces in staging tables.

Regular collection and analysis of performance information can help track service level agreement compliance and provide the basis for proactive problem management.

Regular Collection and Reporting of Performance Metrics

One of the major practices that successful customers perform is the regular collection of performance statistics, analysis of the statistics and reporting pertinent information to relevant parties within the organization as well as ORACLE. Collection of such information can help identify bottlenecks and badly performing transactions, as well as help understand how the product is being used at your site. They offer proof of both good and bad performance and typically allow sites to gauge the extent of any issue.

The Customer Care and Billing/Enterprise Tax Management product contains a number of collection points in the architecture that are useful for real time and offline collection of performance related data. Full details of the collection points are documented in the “**ORACLE Utilities Customer Care and Billing/Enterprise Tax Management Performance Troubleshooting Guide**”. Using the guide, decide which statistics are important to the various stakeholders at your site, decide the frequency of collection and format of any output to be provided. Use your sites Service Level Agreement (SLA), if it exists, for guidance on what to report.

In V2 a concept of record ownership was introduced to ensure critical data needed for operation of the product was not accidentally deleted.

Respecting Record Ownership

In Customer Care and Billing/Enterprise Tax Management V2.x and above, the concept of ownership of records was introduced. A data element was added to data to indicate the owner of the object and is used to protect key data supplied with the product from alteration or deletion. It is used by the online system to prevent the online users accidentally causing critical data failures. The owner is also used by the upgrade tools protect the data from deletion.

The ownership of the record determines what you can do with that record:

- **Framework** - If the record is owned by “Framework” then implementation teams cannot alter or delete the record from the database as it is deemed critical to the running of the Framework. This is usually meta-data deemed important by the Framework team. For example the user **SYSUSER** is owned by the “Framework”.
- **Customer Care & Billing** - If the record is owned by “Customer Care & Billing” then some changes are permitted but deletion is not permitted as the record as it is necessary for the operation of the Customer Care and Billing/Enterprise Tax Management product. The amount of change will vary according to the object definition.
- **Customer Modification** - If the record is owned by “Customer Modification” then the implementation has added the record. The implementation can change and delete the record (if it is allowed by the business rules).

Basically you can only delete records that are owned by “Customer Modification”.

It is possible to alter or delete the records at the database level, if permitted by database permissions, but doing this will produce unexpected results so respect the ownership of the records.

Log retention can enhance the analysis of historical performance analysis and help provide establish a repository for helping detect recurring problems.

Backup of Logs

By default Customer Care and Billing/Enterprise Tax Management removes existing log files from **\$SPLSYSTEMLOGS** (or **%SPLSYSTEMLOGS%** for Windows platforms) upon restart. This is the default behavior of the Customer Care and Billing/Enterprise Tax Management product but may not be desirable for effective analysis as the logs disappear.

To override this behavior the following needs to be done:

- A directory needs to be created to house the log files. Most sites create a common directory for all environments on a machine. The size allocation of that directory will depend on how long you wish to retain the log files. It is generally recommended that logs be retained for post analysis and then archived (according to site standards) after processing to keep this directory relevant. Typically customers create a subdirectory under **<CI SAPP>** to hold the files.
- Set the **SPLBCKLOGDIR** environment variable in the **.profile** (for all environments) or **\$SPLEBASE/scripts/cmenv.sh** (for individual environments) to the location you specified in the first step. For Windows platforms then the environment can be set in your Windows profile or using **%SPLEBASE%/scripts/cmenv.cmd**.
- Logs will be backed up at the location specified in the format **<date time>. <environment>. <filename>** where **<date time>** is the date and time of the restart, **<environment>** is the id of the environment (taken from the **SPLENVIRON** environment variable) and **<filename>** is the original filename of the log.

Once the logs have been saved you must use log retention principles to manage the logs under **SPLBCKLOGDIR** to meet your sites standards. Most sites “archive” the logs to tape or simply compress them after “post processing” the log files (See “**Post Process Logs**” for more details on post processing).

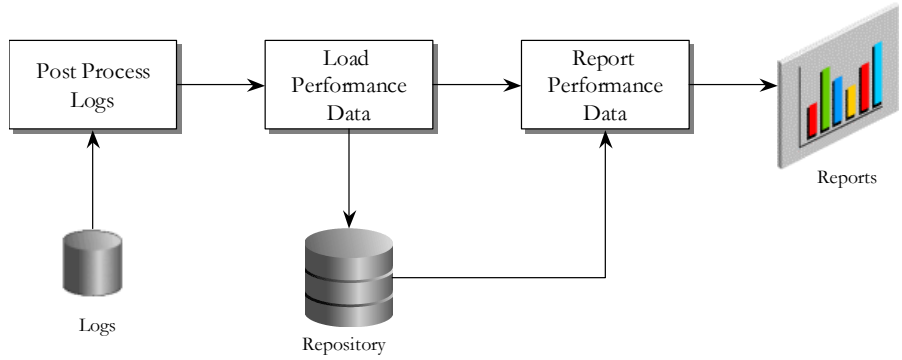
Post Process Logs

Post processing logs to summarize the information into a concise manner can assist in problem analysis and provide a basis for historical analysis.

The logs written by the various components of Customer Care and Billing/Enterprise Tax Management provide valuable performance and diagnostic information. Some sites have designed and developed methods to post process those logs to extract important information and then report on it to relevant parties.

If the logs are retained by your site (see “**Backup of Logs**” for details on this process), the consider post processing the logs on a regular basis before they are

archived or deleted permanently. One approach is to extract that information from the logs and loading the extracted data into some “analysis” repository for regular and trend reporting. The diagram below illustrates the process.

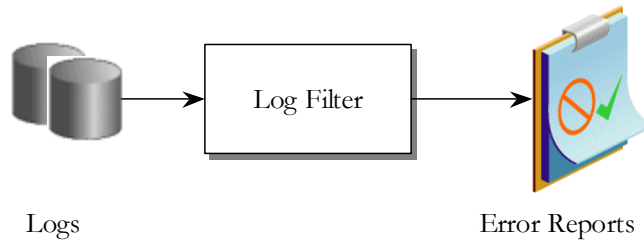


Details of the logs written by the product are documented in the “**Oracle Utilities Customer Care and Billing/Enterprise Tax Management Performance Troubleshooting Guide**”. Use these guides to determine what data to extract from the logs for post processing.

Check Logs For Errors

Checking logs on a regular basis can point out intermittent and growing problems that can be assessed and resolved before they adversely affect the business.

One of the most important tasks for a site is to regularly track errors output into logs. Whenever an error occurs in Customer Care and Billing/Enterprise Tax Management, an error record is written to the appropriate log for analysis. Some sites regularly check these logs for these errors and using the information in the log, address the error condition.



Viewing and checking for errors on a regular basis to quickly reduce the amount of error that may occur can detect trends and common problems. The “**Oracle Utilities Customer Care and Billing/Enterprise Tax Management Performance Troubleshooting Guide**” outlines the logs and error conditions contained within those logs.

Operating system settings need to be optimized for the tiers that are on the machine. Otherwise performance problems may occur.

Optimize Operating System Settings

One of the most important configuration settings for Customer Care and Billing/Enterprise Tax Management is the operating system itself. The “**Oracle Utilities Customer Care and Billing/Enterprise Tax Management Installation Guide**” highlights the fact that the operating system parameters

MUST be set to optimal values for the product to perform optimally. Some sites have experienced large improvements in performance by heeding this advice. Sites that have decided to ignore this advice have experience bad performance till the settings were corrected.

Typically, the optimization of the operating system is performed during the implementation and uses the following principles:

- The value of an individual operating system setting is the maximum value of any product on that machine. For example, typically if ORACLE or DB2 is contained on a machine, the values for those products are used. The settings used in this way are usually are sufficient for the other products on that machine.
- If the machine is dedicated for a particular product or tier, then refer to the documentation in the installation guide and the particular vendor's site for further advice on setting up the operating system in an optimal state.

Optimize connection pools

One of the settings that will affect performance is the size of the connection pools at each layer in the architecture. Insufficient pool sizes can cause unnecessary transaction queues that can cause unnecessary delays. Conversely setting the pool sizes too high can cause higher than usual resource usage on a machine also causing adverse performance. So a balance needs to be struck for optimization.

During the implementation the size of the connection pools is determined and configured (with relevant growth tolerances) depending on the usage patterns and expected peak/normal traffic levels. The goal, typically, is to have enough connections available at “normal” traffic levels to minimize queuing and also have the right tolerances to cater for any expected peak periods. Therefore, it is recommended:

- Set the number of initial connections to the “normal” number of connections expected. Remember this is not the number of users that are connecting but the expected number of concurrent connections under normal load.
- Set the tolerances for pool growth (usually a maximum pool size and a connection increment) to the “peak” load expected at any time. This tolerance will have to be tracked to determine the optimal level. Do not be tempted to set this to a very large value as memory and network bandwidth calculations are usually dependant on the values specified and wastage of resources needs to be minimized.

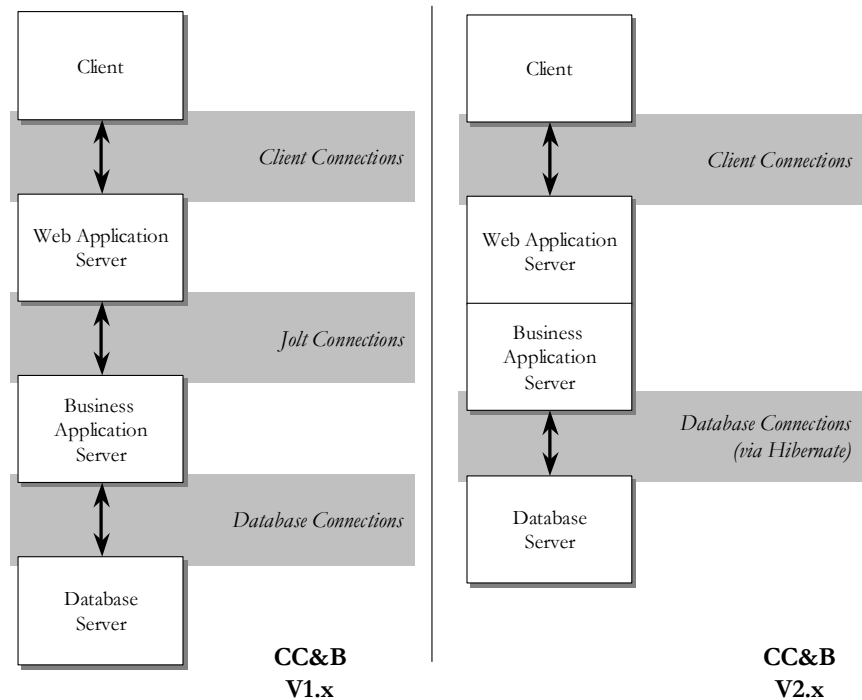
ORACLE Utilities Customer Care and Billing/Enterprise Tax Management has up to three connection pools to configure:

Making sure you have enough client and server connections will reduce application delays due to queuing. Tolerances for the pools are site specific as they are optimized for the site traffic.

- **Client Connections** – These are the number of active connections supported on the Web Application Server from the client machines. Remember that in an OLTP product (such as Customer Care and Billing/Enterprise Tax Management) the number of connections allocated is always less than the number of users on the system. It needs to be sufficient to cater for the number of actively running transactions at any given point of time.
- **Jolt Connections (V1 only)** – These are the number of connections that are open between the Web Application Server and Business Application Server. Typically this number is less than the number of Client Connections, as some of the transactions are served by the Customer Care and Billing/Enterprise Tax Management cache so does not need to access the back end. The number of connections must also be matched by the number of JSL/JSH servers allocated (allowing for multiplexing as well) and the number of connections served by Customer Care and Billing/Enterprise Tax Management itself and the XAI component combined.
- **Database Connections** – These are the number of pooled connections to the database. Customer Care and Billing/Enterprise Tax Management holds these connections open so that the overhead of opening and closing connections is minimized.
 - For Version 1.x of the Customer Care and Billing/Enterprise Tax Management product, the number of connections allocated is dictated by the number of server instances for all Customer Care and Billing/Enterprise Tax Management and Custom Tuxedo servers configured in the **ubb** file for the configuration. Each server instance is allocated a single connection to the database. Do not forget that the **MIN** and **MAX** set the number of instances of a particular server at any time and there can be multiple servers in each environment.
 - For Version 2.x of the Customer Care and Billing/Enterprise Tax Management product, the number of connections allocated is dictated in each individual web applications² **hibernate.properties** file for the c3p0 connection pool.

In summary:

² Only XAIApp and the Customer Care and Billing/Enterprise Tax Management application use database connections. Help and AppViewer are “static” applications.



Refer to the “**ORACLE Utilities Performance Troubleshooting Guides**” for advice on the configuration and monitoring of the connection pools.

Read the available manuals

Customer Care and Billing/Enterprise Tax Management includes a set of documentation that should be downloaded with the software and read as part of the implementation and support of Customer Care and Billing/Enterprise Tax Management. The following technical documentation is available on the distribution web:

- **Customer Care and Billing/Enterprise Tax Management Installation Guide for ORACLE | DB2 | SQL Server** – Installation documentation for the base product including supported platforms and required patches.
- **Developer documentation** – This is detailed documentation on the customization aspects of the implementation including standards for implementations. This includes:

V1.x Documentation	V2.x Documentation
Application Logs - List of logs produced by the development and deployment process.	Application Logs - List of logs produced by the development and deployment process.
Common Routines - List of common routines used by	

Due to the ISV licensing of Web Application Servers, there may not be as much details as other platforms. Refer to the vendor’s site for more detailed information.

V1.x Documentation	V2.x Documentation
programmers	
Cobol Programming Standards - List of naming conventions and programming advice used for COBOL modules including algorithms, Tuxedo services, business objects etc.	Cobol Programming Standards - List of naming conventions and programming advice used for COBOL modules including algorithms, business objects etc
	<p>Java Programming Standards - List of naming conventions and programming advice used for java modules including algorithms, business objects etc</p> <p>Java Annotations – Brief overview of the Customer Care and Billing/Enterprise Tax Management annotation classes available to the java developer.</p> <p>Public API – Overview of the API available to the java programmer.</p>
SQL Programming Standards - Documentation of the SQL standards used in Customer Care and Billing/Enterprise Tax Management.	SQL Programming Standards - Documentation of the SQL standards used in Customer Care and Billing/Enterprise Tax Management.
	HSQL Programming standards - Documentation of the Hibernate SQL standards used in Customer Care and Billing/Enterprise Tax Management.
User Interface Design Standards - Documentation about the User Interface standards used by Customer Care and Billing/Enterprise Tax Management	User Interface Design Standards - Documentation about the User Interface standards used by Customer Care and Billing/Enterprise Tax Management
Database Design Standards - Documentation of the database	Database Design Standards - Documentation of the database

V1.x Documentation	V2.x Documentation
standards employed in Customer Care and Billing/Enterprise Tax Management including naming conventions for tables and columns and layout advice.	standards employed in Customer Care and Billing/Enterprise Tax Management including naming conventions for tables and columns and layout advice.
System Table Guide - Documentation of the meta data tables used in the development process.	System Table Guide - Documentation of the meta data tables used in the development process.
Utilities - Documentation of the other development utilities used by the SDK.	Utilities - Documentation of the other development utilities used by the SDK.
Technical Background - An introduction to the development process and internals of Customer Care and Billing/Enterprise Tax Management.	Overview - - An introduction to the development process and internals of Customer Care and Billing/Enterprise Tax Management.
Packaging Utilities - Documentation of the tools provided to package custom builds.	Packaging Utilities - Documentation of the tools provided to package custom builds
	Key Generation – Overview of the routines and tables used for generation of random keys in the Customer Care and Billing/Enterprise Tax Management product.
	Application Workbench Overview – An overview of the Application Workbench component of the SDK.
	User Guide – A developer’s cookbook and users guide to the SDK.

- **Customer Care and Billing/Enterprise Tax Management Utilities Documentation** – This is detailed guides to the various tools supplied with Customer Care and Billing/Enterprise Tax Management including:
 - **Background Processing** – Details of all the background processes available with the product.

- **Reports** – Details of the reporting interface available with the product including installation of the algorithm and configuration of the reporting interface.
- **Batch Scheduler** – Details of the internal batch scheduler delivered with the product.
- **CTI/IVR Integration** – An overview of the installation, capabilities and configuration of the CTI/IVR integration components delivered with the product.
- **Framework/System Wide Standards** – An overview of the various UI standards employed by the product.
- **Application Security** – An overview of the authorization security model used in the product including guidelines for configuration.
- **User Interface Tools** – An overview of the meta data tools available for the user interface including menus, navigation keys etc.
- **Zone Configuration** – An overview of how to configure the zones and portals supplied with the product.
- **Database Tools** – An overview of the Meta data tools available for maintenance object, table and field definition including auditing.
- **Algorithms** – An overview of all the algorithms supplied with the product.
- **Scripting** – Details of the Business Process Scripting engine supplied with the product including configuration.
- **Application Viewer** – Overview of the maintenance and operation of the Data Dictionary and code view supplied with the product.
- **XAI** – Detailed overview and configuration of the XML Application Integration component of the product.
- **LDAP Import** – Detailed overview of the LDAP import function supported by the product to synchronize LDAP information with the authorization information stored in the product.
- **Configuration Lab** – Detailed configuration and operations of the copy facility supplied with the product.
- **Archive Engine** - Detailed configuration and operations of the archive and purging facility supplied with the product.
- **Conversion Tool** – Detailed configuration and operations of the conversion component supplied with the product.
- **Web Self Service** - Detailed configuration and operations of the sample Web Self Service facility supplied with the product.

Supplemental documentation is available from your local Utilities Global Business Unit representative.

- **Integration with other ORACLE Utilities products** – An overview and configuration guide to integration to MWM, OMS and EAM supplied with the product.
- As part of an implementation supplemental documentation is available from the SPL implementation team, as part of the SPL Implementation methodology, upon request including:
 - **Software Configuration Management Series** – Series of documents that outline a generic process (that can be used as part of the site process) for managing code and data changes. This series includes documents that cover concepts, change management, defect management, release management, version management, distribution of code and data, management of environments and auditing configuration. Additionally the series includes documentation on how to implement single fixes, service packs and full application server upgrades.
 - **Performance Troubleshooting Series** – A detailed guide of the tracking points available in the architecture for performance and a troubleshooting guide based upon common problems. This does NOT include SNMP tracking.
 - **ConfigLab Design Guidelines** – A document on how to design, setup and monitor a ConfigLab solution for an implementation. This is a companion document to the Software Configuration Management Series.

Be familiar with this documentation and be aware that documentation is also upgraded regularly with each release of Customer Care and Billing/Enterprise Tax Management with new and improved information and advice.

Implementing Industry Processes

While not critical some sites have found benefit from adopting a standard set of industry practices.

Implementing a product such as Customer Care and Billing/Enterprise Tax Management can mean that an IT has to adopt new processes in the company to cater for the new product in their portfolio of applications. This is not unique to Customer Care and Billing/Enterprise Tax Management. Any new product that is implemented into an IT portfolio not only requires business process changes but also IT process changes.

In the IT industry at the moment most software application vendors are realizing that implementing a product is not just simply configuration, there are some change management that needs to be performed with the IT group. Luckily the industry has started to adopt some sort of standard framework that helps define an IT "business" and the processes necessary to run that business. This framework is called the IT Infrastructure Library.

IT Infrastructure Library (ITIL) is a set of consistent and comprehensive documentation of best practice for IT Service Management. Used by many hundreds of organizations around the world, a whole ITIL philosophy has grown up around the guidance contained within the ITIL books and the supporting professional qualification scheme. ITIL consists of a series of books giving guidance on the provision of quality IT services, and on the accommodation and environmental facilities needed to support IT. ITIL has been developed in recognition of organizations' growing dependency on IT and embodies best practices for IT Service Management.

The ethos behind the development of ITIL is the recognition that organizations are becoming increasingly dependent on IT in order to satisfy their corporate aims and meet their business needs. This leads to an increased requirement for high quality IT services. ITIL provides the foundation for quality IT Service Management. The widespread adoption of the ITIL guidance has encouraged organizations worldwide, both commercial and non-proprietary, to develop supporting products as part of a shared "ITIL Philosophy".

For more information about ITIL refer to <http://www.itil.co.uk>

The implementation methodology of Customer Care and Billing/Enterprise Tax Management includes references and interfaces to the ITIL recommended procedures and aspects of ITIL have been used at Customer Care and Billing/Enterprise Tax Management sites.

Using Automated Test Tools

Some of the Customer Care and Billing/Enterprise Tax Management sites around the world use third party testing tools for performance and regression testing. While Customer Care and Billing/Enterprise Tax Management is open in terms of the standard it uses not all test tools are applicable to simulate exact expected traffic. In choosing an automated testing tool that you can use with Customer Care and Billing/Enterprise Tax Management the following must be supported:

- **Support for HTTP** – The automated test tool must be able to trap HTTP traffic, as this is the traffic used by the Customer Care and Billing/Enterprise Tax Management product. If the tool supports HTTPS, and you intend to use the HTTPS protocol, be careful as support for HTTPS varies greatly with most testing tools.
- **JSP Support** – Customer Care and Billing/Enterprise Tax Management uses JSP coding to perform most functions. A tool that can leverage this technology will enable screens to be recognized.
- **Support simulation of IE caching** – The Customer Care and Billing/Enterprise Tax Management client utilizes the Internet Explorer cache to locally hold an image of the screen for performance reasons. The

If you are doing regular regression testing then you can use Automated-testing tools to save time. Not all sites use them but they can come in handy.

automated test tools needs to be able to simulate this behavior otherwise results will not reflect reality.

- **Support Pop up screens** – The Customer Care and Billing/Enterprise Tax Management utilizes pop up windows for some lists and some searches as well as confirmation and error messages. The automated test tool needs to be able to support the use of these to adequately simulate Customer Care and Billing/Enterprise Tax Management transactions.
- **Session Cookies** – The Customer Care and Billing/Enterprise Tax Management product utilizes in-memory session cookies to pass security credentials as part of every screen call. This is an essential part of the security authentication used on the product. The automated test tool will need to simulate this behavior otherwise the security screen will be displayed prior to EVERY screen call, which does not simulate reality.
- **Valid calls** – Ensure that the test tools simulate “valid” Customer Care and Billing/Enterprise Tax Management calls. A “valid” call is a call that the Customer Care and Billing/Enterprise Tax Management browser user interface issues to the web server or a call that the XAI component will accept. An invalid call that is sent by a test tool to Customer Care and Billing/Enterprise Tax Management may result in unpredictable results. Check EVERY call is “valid” (try them with Customer Care and Billing/Enterprise Tax Management browser user interface and simulate the call) and fix any invalid calls.

The following products have been used with Customer Care and Billing/Enterprise Tax Management at customer sites:

- Borland Silk Performer
(<http://www.borland.com/us/products/silk/silkperformer/index.html>)
- Mercury Load Runner
(<http://www.mercury.com/us/products/performance-center/loadrunner/>)
- IBM Rational Performance Tester (<http://www-304.ibm.com/jct03002c/software/awdtools/tester/performance/index.html>)

Custom Environment Variables or JAR files

Implementations of the Customer Care and Billing/Enterprise Tax Management product sometimes use third party java classes or third party tools to perform specialist functions. Sometimes these tools require additional configurations settings that can be integrated in the infrastructure provided by the Customer Care and Billing/Enterprise Tax Management product. For example, if you use a third party jar file to be called by the Customer Care and Billing/Enterprise Tax

It is possible to add custom jar files or add/change environment variables, for third party tool integration, as part of the Customer Care and Billing/Enterprise Tax Management startup process at a global or local level.

Management product then you will need to add it to the **CLASSPATH** to ensure it is picked up by the runtime.

Luckily, there is a feature that allows custom environment variables settings and other commands to be run after the **spl environ. sh** script (or **spl environ. cmd** on Windows) has been executed.

To do this create a **cmenv. sh** script (or **cmenv. cmd** on Windows) in the **\$SPLEBASE/scripts** directory (**%SPLEBASE%\scripts** on Windows) with the commands you want to execute. For example, if an implementation used AXIS2 jar files to call web services. Well you place the AXIS2 jar files in a central location (e.g. **/axis/lib** in this example) and create a the **cmenv. sh** script with the lines:

```
export CLASSPATH=/axis/lib/axis.jar; $CLASSPATH
```

or

```
set CLASSPATH=c:\axis\lib\axis.jar: %CLASSPATH%
```

When **spl environ. sh** script (or **spl environ. cmd** on Windows) runs it will look in the scripts directory for the existence of the **cmenv. sh** script (or **cmenv. cmd** on Windows) and executes it.

Additional to this, it is possible to do this WITHOUT adding the **cmenv. sh** script (or **cmenv. cmd** on Windows). Set the **CMENV** environment variable to the location of a script, with the above commands contained, BEFORE running any command and **spl environ. sh** script (or **spl environ. cmd** on Windows).

The **CMENV** facility is for global changes as it applies across all environments and the **cmenv. sh/cmenv. cmd** solution is per environment. You can use both as **CMENV** is run first then **cmenv. sh/cmenv. cmd**.

Note: It is possible, using this technique, to “manipulate” any environment variable used by the Customer Care and Billing/Enterprise Tax Management product but this is not recommended.

One of the critical components of any site is the backup and recovery regime used to protect the business. Typically most sites, reuse their existing regimes and tools.

Backup and Recovery

One of the most critical components of the implementation and ongoing support of Customer Care And Billing/Enterprise Tax Management at a site is the ability to backup the data and software to ensure business continuity. Equally important is the ability to easily restore that data if the need arises.

Typically a site will have a preferred regime and set of tools that is used to achieve a backup and recovery of all systems that the site. When implementing Customer Care And Billing/Enterprise Tax Management this regime and set of tools is typically reused to cater for the products and business needs.

When considering a backup regime for Customer Care and Billing/Enterprise Tax Management the following should be considered:

- There is nothing within Customer Care and Billing/Enterprise Tax Management technically that warrants a particular approach to Backup and Recovery. Most customers continue to use their existing approaches.
- There is nothing within Customer Care and Billing/Enterprise Tax Management technically that warrants a particular backup and recovery tool. Most customers use the native tools provide with their platforms, for cost savings, but some customer have purchased additional infrastructure to take advantage of faster backups/recoveries or additional features provided by such tools.

If your site does not have a backup regime already the following can be considered “default industry practice”:

- Use “Hot” Incremental backups on Production during the business week to reduce outage times.
- Do a FULL backup (“Hot” or “Cold”) once a week at least to reduce recovery times.
- Verify backups after they are taken to reduce risk of delayed recoveries.
- On non-Production, consider either the same regime as Production or consider regular FULL backups at peak periods in an implementation.

It is possible to run the Help and AppViewer web Applications in offline mode to help assist designers and developers with data dictionary, configuration and help information.

Help and AppViewer can be used standalone

The Help and AppViewer components may be used in standalone mode (a.k.a. offline mode). This can be handy for developers, designers and architects who do wish to access up to date information without the need to connect to a “live” copy of the product at their site.

Under the **spl app/applications** directory on V2.x or **ci sdomai n/applications** on V1.x, there are two directories named **hel p** and **appVi ewer**. These contain the online help and AppViewer application and data. Copy these directories to your desired target machine (such as a shared drive, web server or your laptop).

*Note: On platforms supporting WebSphere and Oracle Application Server the directories are contained within WAR files named **hel p. war** and **appVi ewer. war**, these will need to be decompressed on the target platform using an appropriate utility such as jar from the Java SDK or WinZip (or similar).*

To operate the applications in standalone mode you will need to open the following files in your web browser:

- **appVi ewer. html** – Application Viewer startup file. It is also possible to reconfigure the behavior of the standalone copy by altering the **confi g. xml** file located in the **confi g** subdirectory of the AppViewer.

- **SPLHel p. html** – Help file located in language subdirectory (**ENG** = English et al).

Note: The AppViewer and Help applications are only supported in the browsers supported by the product.

The ConfigLab or Archiving re-registration process only needs to be performed if there are table/view additions or removals as part of any product upgrade, single fix or custom code release.

Re-Register only when necessary

As part of the ConfigLab definition process it is necessary to **register** the environments to be used by ConfigLab. The registration process creates remote synonyms (*the database technology used to achieve this will vary from database type to database type*) and an environment reference in the database.

One of the processes that must be performed is that the environments must be “re-registered” after upgrades (product as well as customization upgrades) as the upgrade may remove or add a new table or view and this needs to be reflected in the synonyms.

The registration process **does not need** to be executed if the product upgrade or customization upgrade does not add or remove any tables or views. This will save time in the upgrade process. If there are no database changes to reflect then the re-register process will remove all synonyms and rebuild exactly the ones removed. This can be a waste of time. Basically remember only to re-register if there are any database table or view additions or deletions.

The installation procedure creates a number of default userids with default passwords. It is pertinent to reduce security risk by changing the default passwords and reflecting that change in the configuration.

Secure default userids

There are a number of default users (and associated default passwords) that are supplied with the installation of Customer Care And Billing/Enterprise Tax Management. It is recommended that the default users and their passwords be altered according to the site security standards. The table below lists the default users supplied with the products:

Userid	Scope	Comments
CCBUSER	Default user for Business Objects integration (some versions)	See CI SREAD
CI SADM	Default Database Administrator account.	Owner of the database. Database Administrators are the only valid users of this account. This account is created during the database creation process.
CI SREAD	Default Reporting account	This account is used by Archiving, ConfigLab and Reporting. Only available on ORACLE database installations.

Userid	Scope	Comments
		This account is created during the database creation process.
ci ssys	UNIX administration account	Treat this user as you would treat root or an Administrator account.
CI SUSER	Default Application account	This account is used for all application database access. This account is created during the database creation process.
RPTUSER	Default user for Business Objects integration	See CI SREAD
SPL	Default Application user (v1.x)	See SYSUSER .
STGADM	Default Conversion Staging Administration account	See CI SADM
STGREAD	Default Conversion Staging Read only account	See CI SREAD
STGUSER	Default Conversion Staging account	See CI SUSER
SYSUSER	Default initial framework user provided with product.	This user needs to be available to add other users. This needs to be defined to the Web Application Server on install. The password will reside in the repository defined in the Web Application Server (<i>usually LDAP</i>).
WEB	Web Self Service Default User	See SYSUSER
XAI	Default XAI userid (some versions)	See SYSUSER

Note: There are other userids supplied by products used by Customer Care And Billing/Enterprise Tax Management, refer to the documentation for the products on these users.

Consider specifying different authentication/authorization userids for different modes of access for security and traceability.

Consider different userids for different modes of access

Note: It is not possible to configure Customer Care And Billing/Enterprise Tax Management to use different database accounts for access. All modes of access will share the relevant pool of

database connections as a single database user (usually **CIUSER** or **STGUSER**, for conversion).

By default the application is configured to either use SYSUSER, SPL or XAI to access the product for online, XAI and in background processing. This means any audit or staging records are associated with a common userid. Some implementations have created additional userids to use as a filter for reporting, traceability and auditing purposes. The following guidelines may be used in this area:

- Create a different userid for XAI transactions. This allows tracking of XAI within the architecture. It is also possible to assign each transaction in XAI a different userid, as it is passed as part of the transaction but usually most customers consider this overkill.
- Create a different userid for each background interface. This allows security and traceability to be tracked at a lower level.
- Create a generic userid for mainstream background processes. This allows tracking of online versus batch initiation of processes (especially To Do, Case and Customer Contact processing).

Note: Remember that any Customer Care And Billing/Enterprise Tax Management user must be defined to the product as well as the authentication repository.

Don't enable auditing on financial objects such as bills, payments, adjustments etc as they are automatically audited using financial transaction objects.

Don't double audit

The Customer Care And Billing/Enterprise Tax Management product has an auditing facility that is “soft” configured. The facility can be enabled by configuring the auditing parameters (location of the audit data, audit rules etc) against the meta data definitions of the tables. This ensures that any online or XAI updates are audited according to those rules. Auditing is used to track online changes to critical entities.

The financial component of Customer Care And Billing/Enterprise Tax Management already has a separate auditing facility, as all customers generally require it. Any changes to financial information such as payments, adjustments, bills etc are registered in the Financial Transaction tables. Therefore enabling auditing on those entities is not required and constitutes “double” auditing (i.e. auditing information is stored in two places).

While the impact of the “double” auditing may be storage related, enabling auditing on bills, for example, can have a performance hit on online bills. Customers with large numbers of bill segments per bill (i.e. several hundred) have experienced negative performance impact during online billing when “double” auditing is enabled on financial entities. This does not affect batch performance as auditing is not used in batch.

Avoid using different size machines for the infrastructure as the cost in many cases will be higher than using identical machines.

Use Identical Machines

The flexibility of the technology used by the Customer Care And Billing/Enterprise Tax Management product allows the ability to mix-and-match different hardware for a configuration. While this may be attractive and allow for some innovative solutions, it makes overall manageability and operations harder. Hence, it should be avoided.

Having identical hardware allows for ease of stocking spare parts, better reproducibility of problems (both software and hardware), and reduces the per platform testing cost. This cost, in many cases, will surpass the savings from reusing existing disparate hardware.

Restarting machines regularly can clear out memory and free space.

Regularly restart machines

It is generally a good practice to restart servers periodically. This recovers “slow” memory leaks, temporary disk space build-up, or other hidden problems that may manifest themselves only when the server is up for such a long duration. This is a simple way to avoid unexpected or unexplained failures.

Most hardware vendors have recommendations on optimal time intervals to restart machines. Some vendors even “force” the issue for maintenance reasons. Check with your vendor for specifics for your platform.

Do not use direct SQL against the database as it may corrupt the configuration.

Avoid using direct SQL to manipulate data

Unless the outcome can be verified as correct, you should not use ANY direct SQL statement against Customer Care And Billing/Enterprise Tax Management database as you may corrupt the data and prevent the product from operating correctly.

All the data maintenance and data access in the product is located in the Business Objects. The Business Objects validate ALL changes against your sites business rules and the rules built into the product. If you are using the objects to manipulate the data then integrity is guaranteed as:

- All the validations including business rules, calculations and referential integrity are contained within the Business Objects.
- The Business Object performs a commit when all validations are successful. If any validation is failed the whole object is rolled back to a consistent state. In background processing, a commit is performed after a number of Business Objects have been processed (known as the COMMIT INTERVAL). At that point the last commit point is registered on the Batch Control for restarting purposes. If the background process fails between commit points, the database is rolled back to the last commit point.

- All access modes (online, XAI, background processing) from code supplied with the product use the Business Objects for processing. This means that integrity is guaranteed across all modes.
- Any customizations (algorithms etc) using the ORACLE Utilities SDK will should be using the Business Objects.

Using incorrect SQL may violate any of the validations and even make the system unusable.

If you have to manipulate data within the product, use one or more, of the following provided methods:

- The browser user interface.
- XML Application Integration.
- Conversion Toolkit.
- Software Development Kit.

CLIENT COMPUTER BEST PRACTICES

Even though Customer Care and Billing/Enterprise Tax Management is browser based there are some practices on the client machine that affects performance. This section outlines the practices about the client machine that have proved beneficial.

Make sure the machine meets at least the minimum specification

As part of the installation documentation for each installation of Customer Care and Billing/Enterprise Tax Management, the minimum and recommended hardware for the client is specified. Typically SPL takes the following into consideration when specifying this information:

- The minimum and recommended hardware as specified by Microsoft for the operating system used for the client.
- A “typical” set of other applications running on the machine, typically Office style applications.

While all care is taken in specifying the hardware will cost in mind, experience has shown that customers need to review the specification in light of their internal standards.

Internet Explorer Caching Settings

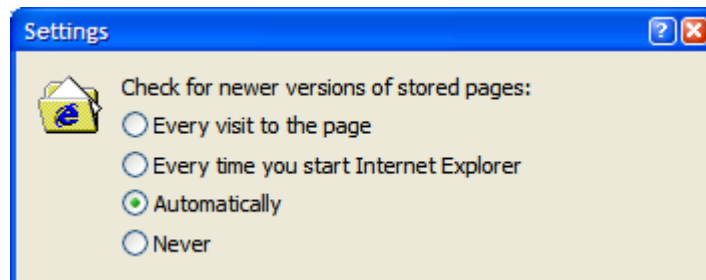
The Internet Explorer settings used in the Citrix image must match the recommended settings as outlined in the Customer Care and Billing/Enterprise Tax Management Installation Guide, which includes:

- Internet Explorer cache settings should be set to “Automatically” NOT “Every visit to EVERY page”. Certain elements on the browser user interface pages

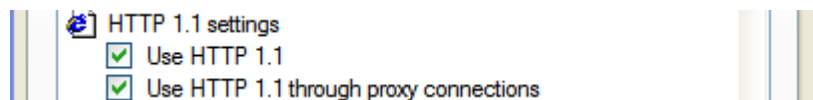
Check the Customer Care and Billing/Enterprise Tax Management installation documentation to see what the minimum specification of the client machine is expected to be.

Optimization of the Internet Explorer cache settings can reduce bandwidth significantly and improve performance.

are cached on the client for performance reasons. Incorrect setting of the cache settings in Internet Explorer will increase bandwidth usage significantly and degrade performance, as screen elements will be retrieved on each rather than from the cache. The correct setting is shown below:



- Java script must be enabled. The Customer Care and Billing/Enterprise Tax Management framework uses javascript to implement the browser user interface.
- In-memory Session cookies must enabled.
- HTTP 1.1 supports must be enabled. If you use a proxy to get to the server, then also check “*Use HTTP 1.1 through proxy connections?*”.



Clearing out the cache is not a common occurrence and is only done if necessary.

Clearing Internet Explorer Cache

Between upgrades, it is advisable to manually clear the Internet Explorer cache to remove any elements that may be still in the cache that are not applicable to the new version. This is a rare situation but sometimes clearing the cache can ensure corrections in caching or inappropriate elements “left over” from upgrades from being incorrectly displayed.

Typically network card manufacturers optimize the settings of the NIC for the various standards. Usually these defaults are sufficient but you may need to check if the NIC settings have been tampered with and are suboptimal. Reset them to factory defaults to see if that improves performance.

Optimal Network Card Settings

Typically the manufacturers of NIC devices provide a number of configuration settings to allow further optimization of network transmit and receive settings. Typically the defaults provided with the card are sufficient for the needs of the network traffic transmitted and received by the machine.

It may be further optimal to investigate whether changing the settings can improve performance at your site (particularly the number of network buffers used). Altering the settings may improve performance but also may adversely affect performance (due to higher CPU usage). Typically the majority of customers use the default settings provided by the manufacturer.

NETWORK BEST PRACTICES

Customer Care and Billing/Enterprise Tax Management traffic exists on the network with other traffic. Performance and consistency are greatly affected by the traffic on your network.

Ensure legitimate Network Traffic

Ensure bandwidth is only being used for legitimate traffic. Bandwidth can be “stolen” by Trojans and viruses.

One of the major factors on performance is the amount of “legitimate” traffic on the network. The traffic to and from Customer Care and Billing/Enterprise Tax Management shares the bandwidth with all other traffic on the network. If there is any network congestion than all transactions from all network-based applications will be adversely affected.

Some customer sites have found that traffic that is not “legitimate” can adversely affect network performance. Traffic that is considered not “legitimate” includes:

- Traffic generated from viruses and Trojans – There are a plentiful number of viruses and Trojans in the general Internet network that can cause bandwidth issues. Most sites have regular virus protection to minimize the impact to your network but not all. While it is not a requirement within Customer Care and Billing/Enterprise Tax Management to have such protection, the industry in general recognizes the need for such protection.
- Unauthorized “large” transfers – Large transfers of data can adversely affect performance as it can soak up bandwidth if the transfer is not configured correctly. There have been instances of large FTP transfers slowing down traffic on lower bandwidth networks.

Ensuring that only legitimate traffic is on a network can provide greater bandwidth for all applications (including Customer Care and Billing/Enterprise Tax Management) and improve consistency.

Regularly check network latency

Check that ping times between sites are consistent and minimal, as this will affect performance.

In a network, latency, a synonym for delay, is an expression of how much time it takes for a packet of data to get from one designated point to another. In some usages, latency is measured by sending a packet that is returned to the sender and the round-trip time is considered the latency. The greatest impact on performance is inconsistency latency.

The latency assumption seems to be that data should be transmitted instantly between one point and another (that is, with no delay at all). The contributors to network latency include:

- **Propagation** - This is simply the time it takes for a packet to travel between one place and another at the speed of light.
- **Transmission** - The medium itself (whether optical fiber, wireless, or some other) introduces some delay. The size of the packet introduces

delay in a round trip since a larger packet will take longer to receive and return than a short one.

- **Router and other processing** - Each gateway node takes time to examine and possibly change the header in a packet (for example, changing the hop count in the time-to-live field). This is a common cause of network latency.
- **Other computer and storage delays** - Within networks at each end of the journey, a packet may be subject to storage and hard disk access delays at intermediate devices such as switches and bridges.

Minimizing latency or latest ensuring consistent latency is the goal of most of the Customer Care and Billing/Enterprise Tax Management sites. A discussion of latency and how to measure it is contained in the “**Oracle Utilities Customer Care and Billing/Enterprise Tax Management Performance Troubleshooting Guide**”.

WEB APPLICATION SERVER BEST PRACTICES

The Web Application Server is used by Customer Care and Billing/Enterprise Tax Management to serve the pages to the client and contains a control data cache. There are a number of practices that sites find useful for maintaining the health of the Web Application Server.

Make sure that the access.log is being created

One of the key log files for traffic analysis is the **access. l og**. This is a log generated by every hit on the system from the end users. Every element of the screen is logged, asynchronously, including the time and userid. This log must be configured/enabled to be generated by the configuration. Refer to the Web Application servers documentation on how to enable this log to be generated.

The log is generated in W3C common log format and can be analyzed by third party log analyzers for further analysis. A full description of the log, its usefulness and the log analyzers that can read the log are documented in the “**Oracle Utilities Customer Care and Billing/Enterprise Tax Management Performance Troubleshooting Guide**”.

Some customers use the log for various purposes:

- It is possible to track errors and trends from the log using the log analyzers.
- It is possible to parse the log at a low level and determine the number of concurrent users and the users that have used the system (and interestingly conversely who has NOT used the system).
- It is possible to track flows of individual sessions, known as click streaming, to track the screens and data used for the screens.

The access.log contains useful information that can be used for tracking bandwidth and usage patterns to make changes to configuration.

- It is possible to determine the criteria used by users for searches. This is useful for detecting wildcard searching.

This log is useful but it is large so needs to be managed as suggested in “**Backup of Logs**”.

Examine Memory Footprint

Ensure you have enough memory for the number of users you wish to support. JVM's from the various vendors have limitations on the number of concurrent users.

One of the common experiences for ALL the J2EE Web Application Servers that Customer Care and Billing/Enterprise Tax Management runs upon is that there seems to be a Java Virtual Machine (JVM) limit on exactly how many concurrent online users a server will support. Typically the experience has been that between 150-200 concurrent users are served by each instance of a JVM.

There are a number of techniques that are available to maximize this:

- Increasing the java memory parameters used for the JVM – This can be a configuration setting change (WebSphere, Oracle AS) or a script change (WebLogic, Tomcat). Typically customers change the default settings to either 512MB or 1GB per JVM.
- Creating additional servers within the instance to cater for the load.

Customers implement the latter suggestion in the following ways:

- **WebLogic** – A server entry for each new server is setup in the same WebLogic instance. The port number can be the same (if the server is housed on a separate machine, known as clustering) or a different port number (i.e. managed servers). A proxy is required to have a common connection point and to implement load balancing. The memory footprint will be the same size for each server.
- **WebSphere** – A new server is created within the WebSphere instance. The port number can be the same (if the server is housed on a separate machine, known as clustering) or a different port number (i.e. managed servers). A proxy is required to have a common connection point and to implement load balancing. The memory footprint can be different for each server as that is held against the server entry within WebSphere.
- **Oracle Application Server** - A new server is created within the Application instance. The port number can be the same (if the server is housed on a separate machine, known as clustering) or a different port number (i.e. managed servers). A proxy is required to have a common connection point and to implement load balancing. The memory footprint can be different for each server as that is held against the server entry within Oracle Application Server.
- **Tomcat** – A separate complete server has to be cloned within Tomcat to configure multiple servers. Tomcat does not support clustering but can

handle managed servers. A proxy is required to have a common connection point and to implement load balancing.

Java memory management is notorious for being inconsistent. Minimizing the affects of garbage collection can improve performance

Optimize Garbage Collection

Tests with customers and benchmarks revealed that while taking the default settings for the Java garbage collection is usually sufficient, in some circumstances tuning the garbage collection process could yield performance benefits.

Automatic garbage collection is a feature of Java whereby, at a certain memory tolerance, the JVM will examine “objects” in memory and clear any unused objects from memory to prevent out of memory conditions happening. The unused “objects” are considered garbage and hence are collected and removed from memory.

The issue with garbage collection is at the time of the actual garbage collection the JVM “freezes” to perform the task. This can cause performance issues if garbage collection is performed more frequently than expected. Optimization of the garbage collection is a balance between the frequency of the garbage collection and amount of traffic on the server at the time of the collection. The amount of memory to collect will depend on the traffic at the time

Ideally, Sun recommends that garbage collection should occur every few minutes at most, more frequently the more impact on performance. Customers aim for garbage collection every 5 minutes at minimum during normal times. A study on the garbage collection tolerance and their affect is documented at <http://java.sun.com/docs/hotspot/gc/>.

Turn off Debug

Debug information is useful for developers but not for production.

One of the development features of Customer Care and Billing/Enterprise Tax Management is the ability to output useful debugging information as part of the running of the application. While this information is useful for development environments it is not useful for Production or other performance “sensitive” environments.

Most customers change the debug setting to false to disable global debug information. Refer to the “**Oracle Utilities Customer Care and Billing/Enterprise Tax Management Operations and Configuration Guides**” for details of this process.

It is possible to debug individual transactions using the interactive debug facility.

Note: This requires the Application Descriptors for all applications to be updated.

Use of a load balancer for a multi server environment is crucial to ensure consistent response times across all users and all servers. Customers have used either hardware or software load balancers using the appropriate load balancing algorithm for their usage patterns.

Load balancers

Oracle Utilities Customer Care and Billing/Enterprise Tax Management customers who have more than one Application Server (physical or logical) must use a load balancer to route the traffic evenly across the available servers. This load balancer can be either software based (such as a web server with the appropriate “plugin”

from the Application Server vendor) or a hardware based load balancer (such as BigIp or other Layer 7 switches)³. Experience has shown that customers with a large number of users (typically greater than 1500) tend to use hardware load balancers and smaller customers use software based load balancers.

Using load balancers with Customer Care and Billing/Enterprise Tax Management may not guarantee that load is evenly distributed, as the transactions do not have a consistent resource load factor. The resource load factor for any Customer Care and Billing/Enterprise Tax Management depends on the transaction type and the data used in that transaction. For example, Search transactions are different from maintenance transactions and resource usage of any search is dependent on the criteria used. Two executions of the same search will have different response and resource usage profiles. Factored on top of that is the fact that the load on a server is a summation of the all the transactions sent to it and that transactions vary from second to second, minute to minute, hour to hour etc. The best you can do is

When installing a load balancer there are a number of algorithms for load balancing offered:

Algorithms	Processing	Comments
Round Robin	Traffic is routed to each server on a rotating basis.	This is the most common used by Customer Care and Billing/Enterprise Tax Management customers.
Random	Traffic is routed randomly to the servers.	Not commonly used but may be used if traffic is random enough.
Weighted Round-Robin Allocation	Variation on Round Robin but allows support for clusters where all servers are not the same size.	Not generally used by Customer Care and Billing/Enterprise Tax Management customers.
IP Address	Traffic is routed using client IP address as the identifier where servers are assigned IP address ranges.	Has been used by customers but found that has limitations if used with virtual servers such as Terminal services or Citrix.
Load	Load factors of transactions are measured and used to determine which server is best	Not used with Customer Care and Billing/Enterprise Tax Management, as most

³ A layer 7 device is network device that integrates routing and switching by forwarding traffic using application layer (layer 7 in the OSI model) information. For example, such a switch can analyze the data at the application level and make forwarding decisions. Switches for other layers may be used, but Layer 7 switches are the most common used by Customer Care and Billing/Enterprise Tax Management customers.

Algorithms	Processing	Comments
	suited.	load factors are inconsistent across transaction invocations.

Typically most customers use “Round Robin” as it is simple and given load is unpredictable can yield the best results. Most customers understand that on some periods of time the load will not be balanced but on average the load is relatively balanced. Remember that each transaction time is a function of how much data is changed

If using load balancing the following additional advice is applicable:

- Ensure that the load balancer does not interfere with Internet Explorer caching. This may result in a low cache hit rate and increase bandwidth used.
- Ensure that the load balancer supports HTTP 1.1 headers to support compression.
- Ensure that the balancer supports Passive and Active Cookie persistence for session cookies. The Web Application Server uses session cookies, for passing security credentials between the client and the server. The load balancer must not compromise this facility.
- Ensure that the load balancer supports SSL persistence, if SSL is used, to ensure that encryption and decryption are not compromised.

Preload or Not?

One of the startup features of Customer Care and Billing/Enterprise Tax Management V1.5, and above, is the preloading of pages to save time. This preloading process dynamically rebuilds the screen definitions from the XML meta data on startup. While this setting (by default) enables the startup to pre-build them (instead of on first invocation) the startup of the Web Application Server is delayed while the preload process is executing. The startup of the server is delayed until the last of the screens is preloaded.

While the preloading of individual screens is very quick (measured in milliseconds) building all screens (1000+) can cause significant delays to initial availability AFTER a restart. It is possible to influence the amount of preloading using two parameters in the Web Application Descriptor called:

- **preloadAllPages** – This parameter affects how much preloading is taking place if it is preloaded. A value of true preloads every screen for Customer Care and Billing/Enterprise Tax Management. A value of false preloads screens off the Main menu only (the screens the end users will be using).

Preloading is handy for caching data but can reduce availability, as the startup time is longer. Balance availability with performance.

Web Application Servers come with admin and monitoring tools that can be used with Customer Care and Billing/Enterprise Tax Management.

WebSphere: Native or SPL?

One of the features of the WebSphere version of the Web Application Server is the ability to either use the WebSphere infrastructure to start and stop the Web Application or use the provided scripts with Customer Care and Billing/Enterprise Tax Management to do that. Typically customers prefer to use the Websphere tools as they are more familiar with the tools than the Customer Care and Billing/Enterprise Tax Management scripts.

In this case for customers the **spl . sh** (or **spl . cmd** script should only be used to start and stop non-Web Application Server components.

You will need to proxy connections if you use clustering or a number of managed servers. The choice of software or hardware proxy is site specific. Large customers prefer hardware proxies and smaller ones software proxies.

Hardware or software proxy

If the implementation uses multiple servers then a proxy is needed to group the servers into a "cluster" or "managed" configuration for load balancing purposes. There are two alternatives for such a proxy:

- **Software** – Each of the Web Application Server's supported by Customer Care and Billing/Enterprise Tax Management provide a “plugin” to use a HTTP server such as Apache, ORACLE HTTP server, IBM HTTP Server, Netscape or IIS as a proxy. Typically the plugin is installed within the HTTP server and configured to define the server address and scheme of load balancing.
- **Hardware** – Increasingly the network router manufacturers are making hardware products that act as network proxies or load balancers (known as Layer 7 load balancers). Hardware such as BigIp, WebSwitch, NetScaler etc are increasingly performing load balancing within intelligent hardware. In this case, you simply configure the servers and ports to a virtual address in the hardware and the load balancing scheme to use.

Customers with multiple servers are either using a hardware or software proxy. The larger scale customers favoring hardware based solutions. The only thing to remember with a proxy is to make sure the following are taken into account:

- The proxy server must support the IE caching scheme and not disable it or adversely affect its operation. This will increase network through put.
- The proxy server must support session cookies. It must be configured to support the passing and processing of session cookies as they are used for security tokens in Customer Care and Billing/Enterprise Tax Management. Failure of this point will result in the security dialog being displayed before EVERY screen.

Typically java has a limitation of around 150-200 threads (depending on the version of java used), so to support more than this number of active threads you may need to create additional "servers/instances" of the Web Application Server.

Configuring the number of connections that the client machines has to the Web Application Server is important for performance of access to the server.

What is the number of Web Application instances do I need?

One of the most common questions for an implementation is "how many Web Application Servers do I need to support the number of users that we have planned to be attached to the product for Production?" The answer to this depends on the JVM you are using and its limitations.

Tests and experience has shown that the Java Virtual Machine has an internal limitation on the number of threads that can be safely supported for transactions. This is not a sever limitation but represents the number of active transactions (i.e. Users) that are supported on a Web Application Server at any time.

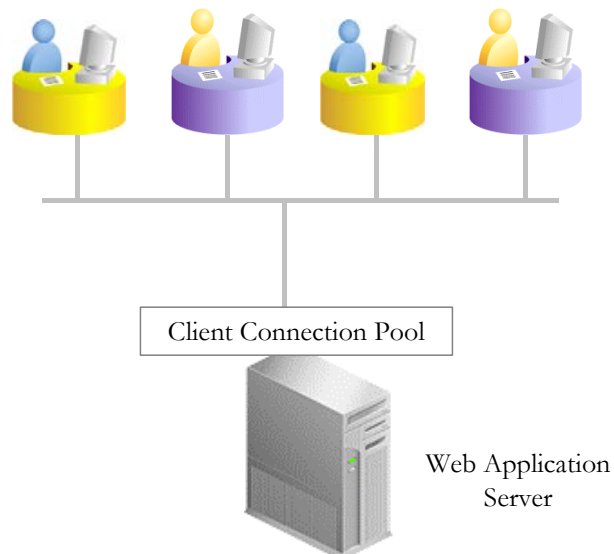
Tests have shown that this number varies between 150 – 200 users on a single Web Application Server JVM instance. The number varies according to the JVM version used and the vendor that supplies the JVM. This number represents maximum number of simultaneous active users hitting the Web Application Server at peak time.

The easiest method for finally determining the number of instances this will become is to divide the number of users expected on the system, at worst case, by 150 and then round up to the next integer. For example, to support 250 users then you can specify 2 instances, to support 400 then you specify 3 instances etc. This method assumes worst case. Regular monitoring of the actual number of connections will reveal whether this needs to be altered.

Configuring the Client Thread Pool Size

One of the first settings that will need to be configured for CC&B is the Client Pool Size on the Web Application Server.

The thread pool manages the number of active connections to the Web Server (see diagram below). A pool is used as it saves resources by allowing reuse of connection threads instead of constantly creating and destroying threads.



Each Web Server calls it a different name:

- **Oracle Application Server/WebSphere Server/Tomcat** - Thread Pool
- **WebLogic Server** - Default Execute Queue/Threads

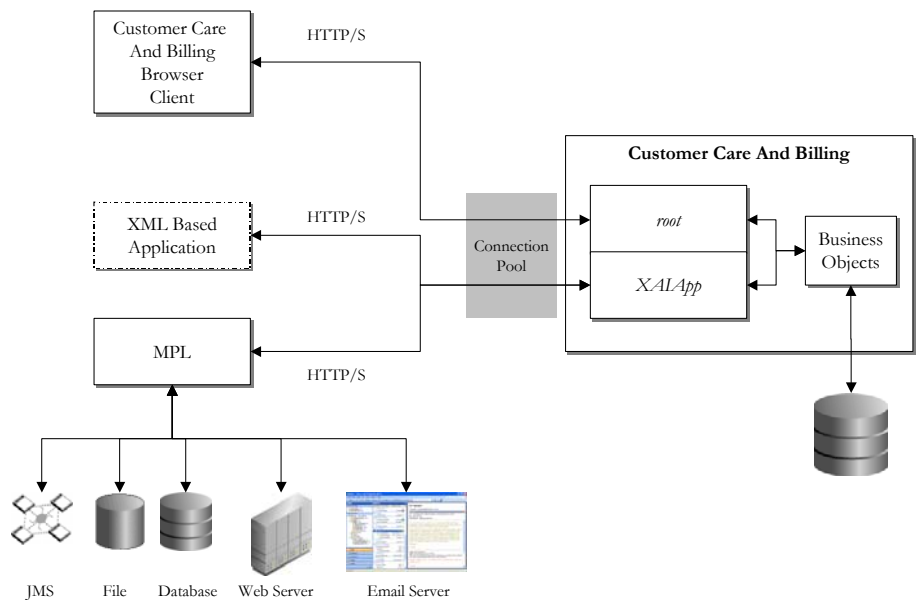
For purposes of this article we will call it “**thread pool**”.

The number of connections allocated in the pool is not the same as the number of users "logged on". As Customer Care and Billing/Enterprise Tax Management is a stateless application the thread pool represents the number of users actually "hitting" the web server, not idle users. Idle users in a stateless application consume little or no resources (actually the only resource an inactive user holds is an open socket to the web server).

Therefore the size of the thread pool at any time is the number of ACTIVE "users" using the product. You might notice that the word "users" is in quotes. For Customer Care and Billing/Enterprise Tax Management the number of "users" for the Web Server is dictated by this formula:

$$\text{Number of Active "Users"} = \text{Number of Active Users in the CC\&B Application} + \text{Number of Active Threads in XAI} + \text{Number of threads in MPL}$$

XAI and MPL threads should be treated as users as well. This is because they typically share the same thread pool.



Thread pools are not static in size, they can grow and shrink in size depending on the traffic volumes experienced. For Customer Care and Billing/Enterprise Tax Management, thread pools have three attributes that need to be considered for sizing:

- **Minimum Size** - This is the size of the thread pool at Web Application Server startup time and the absolute minimum if the pool is shrunk due to inactivity. For Customer Care and Billing/Enterprise Tax Management, this typically represents the "typical load" on the Web Application Server. In other words, the "typical number of active users", on the system at any time. Most customers either use the typical load for the day period or the typical load for after business hours. The latter is used where sites want to minimize the resource usage as the pool is directly related to the amount of memory used by the Web Application Server. The higher the minimum, the higher the memory usage for the server (even at rest).
- **Maximum Size** - This is the maximum size the thread pool can grow to within the Web Application Server responding to the peak load of the traffic. For CC&B this typically represents the "peak" load expected on the largest amount of traffic expected at any point in time. You know "those days". If the maximum is set too low for the load then end users will experience delays even getting a connection to the Web Application Server. Again the value here is also tied to the memory usage. The higher the value, the higher the memory footprint at peak.
- **Inactivity Tolerance** - This value (usually in seconds) is the amount of time that a thread is not allocated to a user before it is destroyed. This value is to reduce the pool size when it has grown about the minimum to detect when there is a drop in traffic. Each Web Application Server will have a default and even a different name for it. Typically customers leave the default but it is worth noting to see if it needs changing in the future.

How do you work out the pool sizes? Customer Care and Billing/Enterprise Tax Management does not have a specific recommendation as it varies according to the volume of transactions but the following has been observed at customer sites:

- For the minimum pool size, set the tolerance to the minimum number of active users for your site. This may be able to deduced from testing but be aware that each transaction has different durations depending on the transaction type (*Maintenance, List and Search*) and the actual data used in the transaction. Experience has shown us that if you divide the number of defined users by three (3) then it may be a good "rule of thumb". Several Customer Care and Billing/Enterprise Tax Management customers have noticed that only about a third of *their* users are active at any time. It should be pointed out that this rule of thumb may not apply to your site but at least it may be used a guide.
- As for the maximum, the only advice that is applicable is that the value should NOT equal the number of users you have defined to the system. The value will vary according to the expected peak traffic experienced at the site. Customers have used between 33-70% of the number of defined

users as the setting for the maximum pool size. To determine the optimum value for your site, it may be necessary to use trial and error.

Note: Setting the minimum and maximum to higher than normal values may waste memory resources on the Web Application Server and cause performance degradation.

Once you have set the settings in your configuration you will need to monitor it to see whether you need to adjust the minimums and maximums.

Customers have determined their own “rules of thumb” and get to the “sweet spot” after a few weeks or months of testing or production.

Typically security information is centrally stored in a repository (such as LAN logons). LDAP is used to get information from that repository and use to authenticate the Customer Care And Billing/Enterprise Tax Management users.

Defining external LDAP to the Web Application Server

Lightweight Directory Authentication Protocol (LDAP) is promoted as a means to leverage an organizational directory as a principal registry for Customer Care And Billing/Enterprise Tax Management user authentication. Therefore as part of the security setup of Customer Care And Billing/Enterprise Tax Management you may need to integrate to an onsite LDAP security repository. This is supported directly by the Web Application server software and Customer Care And Billing/Enterprise Tax Management does not require additional configuration.

Each of the Web Application Server vendors has specific instructions for integrating LDAP but the same process is followed:

- **Determine LDAP Query** - The LDAP query to find the users is required to be determined. Even though LDAP is a standard protocol determined by the IETF the repository structure itself will vary from vendor to vendor and even the same vendors repository structure will vary from customer to customer as it can be altered to suit the business model. This is the hardest part of the process, as the query needs to be correct else it will not return the right records. It is akin to submitting the wrong SQL statement. There are tools, like ADFIND⁴ (for Microsoft ADS for example). to help you with this process.
- **Define LDAP settings to Web Application Server** - Input the query and credentials to access the LDAP repository. This will vary between Web Application servers but basically you need to define the following:
 - The location (host) of the LDAP server(s)
 - The port numbers for the LDAP server(s) (usually 389)
 - The credentials used to read the LDAP server(s) (userid/password)
 - The LDAP query to get the users (and sometimes groups for some Web Application Servers).

⁴ Adfind is a freeware tool for Active Directory Services (ADS). Refer to the ADFIND website (<http://www.joeware.net/freetools/tools/adfind/index.htm>) for more details.

- (Optional) Cache settings to save data retrieved from the LDAP server for performance reasons.

Note: Ensure that the LDAP you have specified contains a definition of the administration account you use to start/stop/administrate Customer Care And Billing/Enterprise Tax Management, else if you have made a mistake it may not be possible to restart the Web Application Server. To reduce the risk of this happening, some sites define two repositories, one to the LDAP server and one to the default security repository provided by the Web Application Server vendor as a precaution. The latter is used to house the administration accounts you do not want to store in the company LDAP.

- **Restart to reflect changes** - Restart the Web Application Server.

For more information see the following sites for your Web Application Server:

- WebLogic - <http://edocs.bea.com/wls/docs91/secmanage/atn.html#1198953>
- WebSphere - <http://websphere.sys-con.com/read/43210.htm> or
- http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.base.doc/info/aes/ae/tsec_ldap.html
- OAS - http://download.oracle.com/docs/cd/B32110_01/web.1013/b28957/ldap3rdparty.htm#CHDIDHDE
- Tomcat - <http://tomcat.apache.org/tomcat-5.0-doc/realm-howto.html>

One of the options available is to synchronize the authentication information stored in an LDAP repository with the authorization information in the Customer Care And Billing/Enterprise Tax Management security model.

Synchronizing LDAP for security

After integrating the LDAP to the authentication components of the Web Application Server some customers then want the ability to “import” the LDAP information into the security model.

The LDAP import process uses a special XAI service (LDAP Import) that reads the information from the LDAP store and creates the appropriate security entries by calling standard XAI services to maintain users and groups. The entire import process may be more appropriately called synchronize because groups, users, and the connections between them are synchronized between the LDAP store and the security model used by Customer Care And Billing/Enterprise Tax Management.

Customer use the LDAP import when they want to synchronize large numbers of user changes from the LDAP to Customer Care And Billing/Enterprise Tax Management or want a “one step process”. Without the LDAP import, every time a user is added to the LDAP server to give that user access to the Customer Care And Billing/Enterprise Tax Management, that user must be manually enter into the authorization model to define what access they have within the product. If you have a lot of users this can be perceived as an unacceptable overhead. With the

import function, you add the user once in the LDAP and import them into the authorization model on a regular basis.

The process for configuring the LDAP import can seem complicated but is logical. The following process is used to configure the LDAP Import function:

- **LDAP Query** - The LDAP query to return the users, groups and group membership is determined using the LDAP vendor or third parties provide. As pointed out previously, each vendor and each site has a different structure to implemented. The LDAP vendor sometimes provides utilities to assist in this. There are third party tools such as AdFind and LDAP Explorer that can be of assistance.
- **Mapping** - A mapping must be provided between the LDAP repository attributes and the User and User Group objects in Customer Care And Billing/Enterprise Tax Management. The Customer Care And Billing/Enterprise Tax Management security authorization model stores additional attributes particular to the product that are usually not stored in the LDAP repository. The following process can be used to assist in the mapping process:
 - Generate an XAI schema for the User and User Group objects using the XAI Schema Editor. This will give you the attribute/tag names to use in the mapping.
 - Use a tool to query the LDAP repository to determine what values and fields have been stored in the repository. This will vary from implementation to implementation.
 - Look at both inputs and find the fields from the LDAP that you can map to the Customer Care And Billing/Enterprise Tax Management schema. The mapping uses the “**cdxName**” attribute for Customer Care And Billing/Enterprise Tax Management fields and “**ldapAttr**” for LDAP field name. For example:

```
<LDAPCDXAttrMapping      cdxName="First Name"
  ldapAttr="cn">
```

The above entry maps Customer Care And Billing/Enterprise Tax Management field “**First Name**” to the “**cn**” field in the LDAP. So when the import is performed it will use the “**cn**” value in “**First Name**”.

- Any fields in Customer Care And Billing/Enterprise Tax Management required but not in the LDAP will use the “**default**” tag instead of the “**ldapAttr**” tag.
- Repeat the above tasks for “**User Group**” and “**membership**” of the groups.

- **Store the mapping file** - Create a mapping file in a location readable by the Customer Care And Billing/Enterprise Tax Management administration account. The format of the file is documented in the “**Importing Users And Groups**” (a.k.a. **94LDAPImport.doc**) documentation.
- **Include the mapping file** - Include the mapping file in the **XMLParameterInfo.xml** configuration file as documented in the “**Importing Users And Groups**” documentation.
- **Define the LDAP server location** - Configure the LDAP Server location and port number in the “**XAI JNDI Server**” dialog to define the location.
- **Run** - Initiate the import as documented in the “**Importing Users And Groups**” (a.k.a. **94LDAPImport.doc**) documentation.

The LDAP import service calls the LDAP Import Adapter, which performs the following:

- It reads the configuration information provided as XAI parameters to the request. Parameters include the Java Name Directory Interface (JNDI) server, user and password for the LDAP server, and the transaction type (i.e., import).
- It connects to the LDAP store using a JNDI specification.
- For each element (user or group) in the request, the LDAP is searched by applying the search filter (specified for the element) and the searchParm (specified in the request).
- The adapter goes through each entry found in the search and verifies whether or not there is already an entry in the system and whether a user belongs to a user group. From this information, it automatically determines the action to be taken:
 - Add
 - Update
 - Link user to group
 - Unlink user from group (by setting the expiration date)
- If the entry is a group, the adapter also imports all the users in LDAP that are linked to the group. If the entry is a user, the adapter imports the groups to which the user belongs in LDAP.
- For each imported entity, the adapter creates an appropriate XML request and adds it to the XAI upload staging table. For example if the action is to add a user, it creates an XML request corresponding to the **CDxXAI UserMaintenance** service; and if the action is to add a

group, it creates an XML request corresponding to the **CDxXAI UserGroupMaintenance** service.

- The XML upload staging receiver processes the upload records in sequential order (based on the upload staging ID). The MPL is used to complete the processing.

Note: If a user is imported because it belongs to an imported group, the adapter does not import all the other groups to which the user belongs. If a group is imported because the imported user belongs to it, the adapter does not import all the other users that belong to the group.

Note: Users and groups whose names exceed the length limit in the system are not synchronized.

After implementation there are sets of tasks that need to be performed on a regular basis.

Routine Tasks for Operations

After the implementation of Customer Care And Billing and Enterprise Tax Management has been completed there is a common set of tasks that IT groups perform to maintain the system. The table below lists these tasks:

Task	Comments
Perform Backups	Perform the backup of the database and file system using the site procedures and the tools designated for your site.
Post Process Logs	Check the log files for any error conditions that may need to be addressed. Refer to “ Post Process Logs ” and “ Check Logs For Errors ” for more details.
Process Performance Data	Collate and process days performance data to assess against any Server Level targets. Identify any badly performing transactions.
Perform Batch Schedule	Execute the batch schedule agreed for your site. This will include overnight, daily, hourly and ad-hoc background processes.
Rebuild Statistics	DB2 and ORACLE require the database statistics for the Customer Care And Billing and Enterprise Tax Management schemas to be rebuilt on a regular basis so that the access to the SQL is optimized. At DB2 sites, a rebind is also required to reflect the changes in the execution plan/packages.
File Cleanup	On a regular basis, the output files from the background processes and logs will need to be archived and removed to minimize disk space usage.
Archive Data not	The Customer Care And Billing and Enterprise Tax

Task	Comments
required	Management products features an inbuilt archiving facility that can transfer transaction data not considered “required” for online processing to another environment, to a file or simply deleted. Refer to “Using Archiving to purge data” for more details.
Run Cleanup Batch Jobs	There are a number of background processes that remove staging records that have been already successfully processed. Refer to “Removal of Staging Records” for more details.

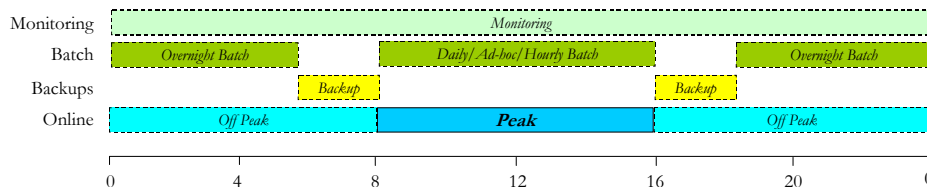
Note: The tasks listed above do not constitute a comprehensive list of what needs to be performed. During the implementation you will decide what additionally needs to be done for your site.

During the implementation it is important to decide what constitutes a typical “business day” for planning purposes.

Typical Business Day

One of the patterns experienced at Customer Care And Billing and Enterprise Tax Management customers is the notion of a “common” definition of a business day. Typically during the implementation the “business day” is defined for planning purposes. It defines when the call center is at peak or non-peak, background processing can be performed and when backups are performed during the business day.

The diagram below illustrates a simplified model of a typical customer “business day”:



Note: The above diagram is for illustrative purposes only and could vary for your site.

Typically a “business day” contains the following elements:

- There is a “peak” online period where the majority of call center business is performed. Typically this is performed in “business hours” varying according to local custom.
- There is a call center “off peak” period where the volume of call center traffic is greatly reduced compared to the peak period. Typically in call centers, which operate 24x7, this represents overnight and weekends. At this time the call center is reduced in size (usually a “skeleton” shift). Some sites do not operate in non-peak periods and rely on automated technology (e.g. IVR) to process transactions such as payments etc.

- Backups are either performed at the start of the peak period or the end of the peak period. The decision is based upon risk around failure of the background processing and its risk to the impact of online processing. Customer Care And Billing and Enterprise Tax Management background processes can be run anytime but avoiding them during peak time will maximize the available computing resources to the successful processing of call center transactions. The backup at the end of the peak period is the most common patterns amongst Customer Care And Billing/Enterprise Tax Management customers.
- Background processes are run at both “peak” and “off peak” times. The majority of the background processing is performed at off-peak times to maximize the computing resources to the successful completion of the background processing. The background processing that is run at “off peak” times is usually to check ongoing call center transactions for adherence to business rules and process interface transactions ready for overnight processing.
- Monitoring is performed throughout both “peak” and “off peak” times. The monitoring regime used may use manual as well as automated tools and utilities to monitor compliance against agreed service levels. Any non-compliance is tracked and resolved.

The definition of the “business day” for you site is crucial to schedule background processing and set monitoring regimes appropriate for the traffic levels expected.

The AppViewer is a combined data dictionary and meta-data information viewer. It is only needed in development environments.

Appropriate use of AppViewer

The AppViewer is a component of Customer Care and Billing/Enterprise Tax Management that displays meta-data in a more useable format. In past versions of the product, it was preloaded with every Customer Care and Billing/Enterprise Tax Management environment. Typically the information is used for design and development work only.

To make the best use of the AppViewer the following advice is offered:

- For V1.x of Customer Care and Billing/Enterprise Tax Management, the AppViewer is preloaded with content and cannot be extended without use of the ORACLE Utilities Software Development Kit. It may be possible to “zero” out the war file and disable the AppViewer menu item. Check with your Web Application Server vendor details of how to substitute a “blank” AppViewer WAR/EAR file.
- For V2.x (and above) of Customer Care and Billing/Enterprise Tax Management, the AppViewer is provided blank intentionally. It must be primed using a predefined set of Batch jobs. This will take data from the meta-data (including ANY customizations) and generate it. You will need

to run the jobs regularly if you update the meta-data regularly and want the information reflected in the Application Viewer.

Batch Control	Usage
F1-AVALG	Generate AppViewer XML file(s) for Algorithm data (includes javadocs). This is code generation as well.
F1-AVBT	Generate AppViewer XML file(s) for Batch Control. This is useful for “run book” information.
F1-AVMO	Generate AppViewer XML file(s) for Maintenance Object data
F1-AVTBL	Generate AppViewer XML file(s) for Table/Field data
F1-AVTD	Generate AppViewer XML file(s) for To Do Type

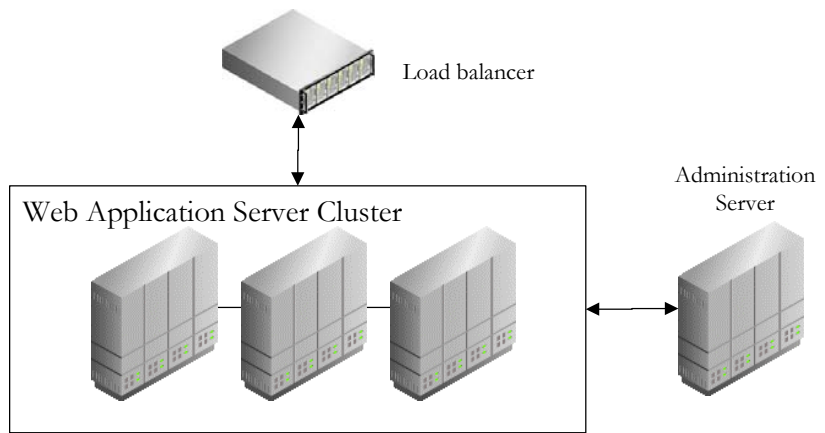
- For V2.x, the introduction of the batch jobs, means you can decide which information is important for your site to display in the AppViewer. For example, if you wish not to have To Do Types documented then you can omit that information by not running that job. If you wish to populate ALL the information then you can use the **genappvi eweri tems** command (or **genappvi eweri tems. sh** for UNIX).
- For V2.x, consider only populating the information in any design and development environments to save disk space. The AppViewer can extend to a number of gigabytes if fully loaded.

One of the important decisions that has to be made, if you are using multiple Web Application Servers, is if you are going to run clustered or managed servers.

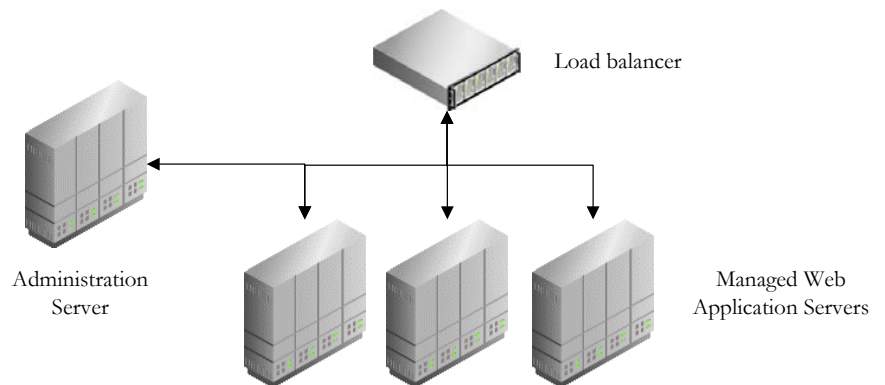
Clustering or Managed?

One of the decisions that has to be made, when dealing with multiple web application servers, is to whether the servers will be clustered or managed. The attributes of each style are outlined below:

- **Clustered** – A cluster is a group of servers running a Web application server simultaneously, appearing to the users as if it were a single server (usually managed by a separate administration server). The advantages of using a cluster is that you can manage the servers as a group and also the servers communicate to each other to monitor availability. Clusters can load balance within themselves as they are in constant communication with each other. The disadvantages are that there is an overhead in communication (usually each server uses multicast to communicate to the other servers in a cluster) and each server must use a different IP address and port number. This means clusters can only operate on one machine per server. The diagram below summarizes a cluster:



- Managed⁵** – A managed set of Web Application servers that are independent of each other. They can be housed on a single machine or multiple machines and can be housed on machines of differing size. The advantage of managed servers is that each server can be targeted for specific user groups and can be managed independently. There is no additional communication between the servers. A separate administration server can manage the servers but that role can be taken by one of the managed servers if desired. The disadvantages are that the load balancing software/hardware housed between the users and the managed servers performs the load balancing and that deployment must be performed individually. The diagram below summarizes managed servers:



There are no clear winners between clustering and managed Web Application Servers as the main factors in the decision are:

- Amount of hardware** – Clustering requires a hardware server per server⁶. Sites where a small number of servers are deployed cannot use clustering.

⁵ Managed servers are sometimes called “Clustering without the overhead” but this is not strictly true.

⁶ It is possible to bypass this limitation using “multi-homing” or using “cluster test” modes provided by Web Application Server vendors but this is not recommended for production use by ORACLE or the respective Web Application Server vendor.

- **Maintenance Effort** – Clustering can reduce maintenance overhead if there are a large number of servers involved. Managed servers require individual maintenance.
- **Tolerance for multi-casting** – Some sites ban multi-casting as it constitutes can be perceived as an unacceptable overhead on the network. Deploying a “private” network between the servers can minimize this, though this is more expensive.
- **Flexibility** – Many sites use managed due to its flexibility in routing particular traffic to particular servers. For example, setting up specific servers for non-call center traffic (e.g. XAI, interfaces, “depots”).

Whether your site uses clustering or managed servers does not factor into high availability solutions as customers have deployed high availability solutions using either technique.

When allocating port numbers ensure that the ports are not used and can be allocated to the relevant components appropriately.

Allocate port numbers appropriately

When installing a copy of Customer Care And Billing/Enterprise Tax Management you need to allocate a number of port numbers for each environment. It is recommended to allocate a previously unused range of ports per environment so avoid port conflicts.

The following table outlines all the port numbers required by Customer Care And Billing/Enterprise Tax Management at installation time:

Port	V1.x	V2.x	P/I	Comments
DBPORT		✓	P	Database Connection Port
JSLPORT	✓		I	Jolt Server Listener Starting port
MPLADMINPORT		✓	I	MPL Administration Port
MPLPORT	✓		I	MPL Administration Port
OASREQPORT		✓	P	OAS Request Port
RMI PORT		✓	I	JVM Child process starting Port
SSLPORT	✓		P	Web Server Port using SSL
WLPORT	✓	✓	P	Web Server Port
WLSSLPORT		✓	P	Web Server Port using SSL
WSLPORT	✓		I	Web Service Listener Port (unused)

Legend: **P** – Port allocated prior to installation of Customer Care And Billing/Enterprise Tax Management, **I** – Port allocated during installation of Customer Care And Billing/Enterprise Tax Management.

Prior to installation of Customer Care And Billing/Enterprise Tax Management, the database and Web Application Server need to be installed and the ports allocated to these components recorded and provided for the installation of the product (they are indicated with a “P” in the table). Each vendor will have the port definitions stored in different places. Refer to the vendor documentation for more information.

When allocating ports (indicated with an “I” in the table) during the installation the following advice may be useful:

- Pick the same port numbering scheme per environment to save time allocating ports. Some sites find using the same last digits for the type of port is helpful. For example, having 4 allocated for **RMI PORT** or **JSLPORT** (6504, 7914, 9724, 22034... etc).
- **RMI PORT** and **JSLPORT** are known as “starting ports” therefore are allocated a number of ports starting from that port number. The **JVMCOUNT** and number of **JSH**’s respectively determine the ports allocated. Ensure that there are free ports in the range starting from that port number.
- Document the ports used in your documentation or **services** file for future reference.
- Do not allocated used ports as there will be port conflicts and potentially the applications will refuse to work.

BUSINESS APPLICATION SERVER BEST PRACTICES (V1.X ONLY)

The Business Application Server is used by Customer Care and Billing/Enterprise Tax Management to process the business logic. It features Tuxedo (Customer Care and Billing/Enterprise Tax Management V1.x only) and the business objects used by the product. There are a number of practices that sites find useful for maintaining the health of the Business Application Server.

Make the Business Application more traceable (V1.x only)

One of the simple things that customers do in BEA Tuxedo, is to make the server processes more recognizable in UNIX (so that the UNIX **ps** command can be used to track environments). By default, all executables have the same name and therefore make tracing via **ps** very difficult. Some variants of UNIX allow more detail in the **ps** command to identify processes from an environment, but not all.

The technique involves the following :

- Change the **DOMAINID** from **splwgci s** in the **ubb** file to the environment name or something more meaningful).

Making the BEA Tuxedo components of the application more traceable can be handy.

- Create symbolic links to the BEA Tuxedo executables with names that include the environment identifier. Update the **ubb** file, replacing the executable names with the symbolic link names. BEA Tuxedo will use the symbolic link names instead of the generic names.

To find the processes for an environment, simply filter on the environment name in UNIX commands. This tip can also apply to Server names, queue names etc to improve general traceability.

Turn on Performance Tracking (V1.x only)

BEA Tuxedo includes a performance tracking facility that does not use a lot of resources and can provide valuable performance and input information for tuning.

One of the features in Tuxedo that customers find useful is the performance tracking facility known as **txrpt**. After every Tuxedo transaction is complete a performance log record is written to the **stderr** file specified for the server, if the facility is enabled. There is no performance impact on the transaction as the value is written asynchronously to the file after the transaction is complete. Therefore the data in the file represents the full time the transaction was active within the business object and database server. It is extremely useful for identifying transactions that are badly performing.

By default, the **ubb** file provided with the Customer Care and Billing/Enterprise Tax Management installation does not enable this facility. To use the facility, add the **-r** switch to the **CLOPT** for each server in the **ubb**. After the facility is enabled, and the business application server restarted, the **stderr** file can be used as input to the **txrpt** command to produce a performance report that can be used to track badly performing transactions.

Refer to the “**Oracle Utilities Customer Care and Billing/Enterprise Tax Management Performance Troubleshooting Guide**” for a detailed description of the facility and how to use the data for tracking performance.

Optimize the Tuxedo Configuration Settings (V1.x only)

Customer Care and Billing/Enterprise Tax Management uses only a few facilities in BEA Tuxedo. Switching off the facilities that are NOT used can save CPU.

The BEA Tuxedo configuration file, located at **\$SPLEBASE/etc/ubb**, can be further optimized using the following advice from BEA:

- Turn off Tuxedo features that Customer Care and Billing/Enterprise Tax Management does not use – Later versions of Tuxedo introduce new features that are turned on by default that are not used by Customer Care and Billing/Enterprise Tax Management. The Customer Care and Billing/Enterprise Tax Management product does not use the multithreading, security and XA features of BEA Tuxedo. For full details, refer to “**Enhancing Overall System Performance**” under the “**Tuning a BEA Tuxedo ATMI Application**” at <http://edocs.bea.com/tuxedo/tux81/ada/adtun.htm#1030643>.

Note: Tests by some customers have not revealed significant savings but it may be useful to minimize CPU for high volume sites.

- The sizing parameters in the **ubb** should follow BEA recommendations.
- Overall the sizing parameters should reflect each other such as:
MAXACCESSERS > **MAXWSCLIENTS** > Sum of the MAX of all servers + Number of JSH running * Multiplexing factor
- The number of server instances per MSSQ set should be between 10-12. Any higher would cause content on IPC resources that causes delays. If more instances are needed, then multiple MSSQ sets should be used.
- The value of **Y** for **LDBAL** will only load balanced connections to the queues not the within queues themselves. This algorithm, although effective, is expensive and should be used only when necessary, that is, only when a service is offered by servers that use more than one queue. Services offered by only one server, or by multiple servers, all of which belong to the same MSSQ (Multiple Server, Single Queue) set, do not need load balancing.
- Consider setting timeouts on individual popular services, such as searches, to minimize their impact on the overall configuration.
- Monitor the BEA Tuxedo queues to check for queuing. This is done while the system is running and under stress. BEA recommend that generally queues that have 3 – 5 transactions waiting are generally acceptable. Excessive waiting transactions in the queues may indicate a need to increase the number of BEA Tuxedo servers of a particular type that are available.

Use Tuxedo TMIB for detailed monitoring (V1.x only)

Tuxedo 8.x and above offers a SNMP like MIB that contains detailed diagnostic and management information about your Tuxedo setup. For more details of the facility refer to <http://edocs.bea.com/tuxedo/tux81/snmpmref/1tmib.htm> and refer to <http://edocs.bea.com/tuxedo/tux81/rf5/rf5.htm> for more details of the individual MIB. Use the following MIB specifications for monitoring Customer Care and Billing/Enterprise Tax Management:

BEA Tuxedo has a SNMP like MIB that can provide lots of performance information. Customers who want detailed tracking can use that information proactively to tune the application.

MIB	Usage
EVENT_MIB	SNMP Style events subscriptions
TM_MIB	Monitoring of Servers, Queues and Services
WS_MIB	Monitoring of Jolt Client/Connection Pool connections

Some customers use the **ud32** utility to query the TMIB and some customers have written a TMIB daemon that collates the TMIB information on a regular basis.

Multiple JSL processes (V1.x only)

One technique that has been used at a few sites is to setup multiple JSL entries, broadcasting on different ports. This offers additional capacity and also covers availability if one JSL shuts down. This is possible by altering the **\$SPLEBASE/etc/ubb** file and adding additional JSL entries. For example:

```
JSL SRVGRP=JSLGRP SRVID=30 CLOPT=" -- -n //host1:6022
-d /dev/tcp -m4 -c1024" RESTART=Y GRACE=0 MAXGEN=255
```

```
JSL SRVGRP=JSLGRP SRVID=40 CLOPT=" -- -n //host1:6032
-d /dev/tcp -m4 -c1024" RESTART=Y GRACE=0 MAXGEN=255
```

To use this in the client configuration you must add the new address to the **j o l t. p r o p e r t i e s** files. For example:

```
appaddrlist=//host1:6022; //host1:6032
```

*Note: The above advice also applies to the failoverlist in the **j o l t. p r o p e r t i e s** files.*

Optimize Server/Service distribution for site (V1.x only)

One of the most successful activities that customers have performed is the optimizing of the service to server setup within BEA Tuxedo to maximize throughput and minimize queuing. The idea of it is to setup a number of dedicated servers that advertise a subset of services that are optimized for speed. The goal of the configuration is provide enough resource to satisfy the request and to remove unnecessary queuing. This method has been named the “Tic Tac Toe” method.

Tic Tac Toe may sound like a game but Customer Care and Billing/Enterprise Tax Management largest customers use the technique to optimize their BEA Tuxedo server to service distribution to minimize queuing.

	Pages	Lists	Searches
Fast (e.g. < 10ms)			
Medium (e.g. > 10 ms)			
Slow (e.g. > 1 sec)			
Common			
Custom			

The key principles of this process are:

- Extract the **txrpt** statistics of a performance test or production (if available). Exercising the services to determine those that have services times that are wildly different from the average for that class of service. This is key information as it outlines the usage profile of the site. This information includes the name of the service called, the number of times called and the average response time.

- Group like services in server lists which will define the servers for the site using the following guidelines:

- You will create services lists based upon service type and speed of service. Services of different types **MUST** be separated. This is to reduce unnecessary queuing as searches and lists have a different resource profile than page objects. Customer Care and Billing/Enterprise Tax Management ships with three types of services, pages (maintenance), list and searches. The type of service is denoted by the last character of the service name (P = Pages, L = List and S = Searches). Similarly Services with similar service times need to be grouped into servers so that queues can be optimized. Decide on what is considered “Fast”, “Slow” and “Medium”. This will form up to approximately nine (9) service lists (i.e. Fast Pages, Fast Lists, Fast Searches, Medium Pages, Medium Lists, Medium Searches, Slow Pages, Slow Lists and Slow Searches).

Note: You might at worst have the nine (9) service lists but in reality it will be less than that.

- Services broadcast by the server must be contained in server lists for that executable.
- Custom services and services NOT used by the implementation need to be defined in servers that either consolidate the services or separate them by service type. Defining services NOT used may seem like a daft idea but experience has shown that eventually they may be used. Not defining such services can result in errors if there is an attempt to use a missing service.
- Administration Services are typically low volume and should have minimal settings unless otherwise recommended.
- High profile services such as Control Central and Portals need to be separated out into individual servers to reduce contention for resources.
- The **MIN** and **MAX** values of the server should be set according to popularity of the server and the average response time used in that server. For example, faster services may need fewer occurrences than slower services.
- The **ubb** file will need to be updated with the latest server lists, **MIN** and **MAX** values and other server tolerances according to the principles outlined above.

Details of how to perform this are outlined in the “**Oracle Utilities Customer Care and Billing/Enterprise Tax Management Performance Troubleshooting Guide**” under “**Tuxedo Service/Server Configuration**”.

BACKGROUND PROCESSING BEST PRACTICES

Customer Care and Billing/Enterprise Tax Management contains a background-processing component to process multiple records over a selected period (daily, nightly, weekly etc). Optimization of this component is critical to the overall success of the implementation.

Optimize the schedule

Optimizing the schedule is important to an implementation to ensure a batch schedule uses the available resources within the time constraint allocated by the site practices.

During the implementation of the product, the implementation will work with the business to determine which processes are applicable to your business to determine the sites schedule. Remember that NO site has implemented the schedule as it is supplied. It is always a relevant subset of the provided schedule.

Business processes and business activity drive the schedule that the implementation will go live with. Some of the jobs will be removed and some will be added. Individual background processes can be removed if business process does not require the process or, for any reason, the process is not applicable to the business. Custom Processes (typically for interfaces) will be added by your implementation team.

Therefore the following goals are applicable to schedule optimization within Customer Care and Billing/Enterprise Tax Management:

- Maximize throughput by limiting process concurrency. Do not run too many processes simultaneously. The CPU can flood if you run too much at the same time.
- Aligning schedule with Bill Cycle and Meter Read Schedules. The Bill cycle and Meter Read cycles, if used, can influence the schedule. Refer to the Business Process documentation for billing and meter reading for more details of this.

The following process can be used to optimize the base schedule supplied with the product:

- Remove jobs that are not to be implemented. Check with the business if the process is applicable or needed for a business process. If in doubt, leave it in.
- Add custom jobs to schedule. For custom interfaces or custom business processes, outside the scope of the base product, you will be adding a few custom batch processes. Your implementation team will have details of these processes.
- Adjust dependencies for the added and removed jobs. When you take away and add jobs, the dependencies will change and the overall flow of data will change.
- Run schedule in test as initially documented – You need to run the new schedule at a basic level to get an idea of how it hangs altogether.

- Gather elapsed times and throughput rates - You need to get some stats to determine which jobs will need optimization and how much optimization is really needed.
- Determine “heavy” jobs – Jobs that take a long time (we will leave that tolerance up to you) need to be determined, as they will greatly affect the overall schedule. These become candidates for multiple threading.

Now that we have the basic information we can start optimization.

- “Heavy” jobs can be run multi-threaded – Consider multiple threading the “heavy” jobs but not too much threading as it can drive up contention. See **“Multi-threading guidelines”** for more information.
- Move scheduling of jobs to minimize number of jobs running in parallel – Reduce contention around heavy jobs by scheduling other jobs earlier or later. Also try not to run a lot of “less heavy” jobs at the same time. It has the same impact that “heavy” jobs have.

When altering the base schedule remember the following:

- If in doubt, do NOT leave it out. Keep processes in the schedule that you are not sure about.
- Only run multi-threaded if necessary – remember too much threading can increase contention and therefore reduce throughput.
- To run background processes during busy business days then reduce the Commit Interval to increase transaction concurrency.
- Where possible, use defaults.
- Set the your site defaults on your Batch controls.

Multi-threading guidelines

Running a background process in multiple threads is almost always faster than running it in a single thread. The trick is determining the number of threads that is optimal for each process.

A good rule of thumb is to have one thread for every 100 MHz of application server CPU available. For example if you have four 450 MHz processors available on your application server, you can start with 18 threads to begin your testing: $(450 * 4) / 100 = 18$.

This is a rule of thumb because each process is different and is dependent on the data in your database. Also, your hardware configuration (i.e., number of processors, speed of your disk drives, speed of the network between the database server and the application server) has an impact on the optimal number of threads. Please follow these guidelines to determine the optimal number of threads for each background process:

Running background processes multithreaded increases throughput but can increase contention.

- Execute the background process using the number of threads dictated by the rule of thumb (described above). During this execution, monitor the utilization percentage of your application server, database server and network traffic.
- If you find that your database server has hit 100% utilization, but your application server hasn't one of the following is probably occurring:
 - There may be a problematic SQL statement executing during the process. You must capture a database trace to identify the problem SQL.
 - It is also possible that your commit frequency may be too large. Commit frequency is a parameter supplied to every background process. If it is too large, the database's hold queues can start swapping. Refer to Parameters Supplied to Background Processes for more information about this parameter.
- It is normal if you find that your application server has hit 100% utilization but your database server has not. This is normal because, in general, all processes are CPU bound and not IO bound. At this point, you should decrease the number of threads until just under 100% of the application server utilization is achieved. And this will be the optimal number of threads required for this background process.
- If you find that your application server has NOT hit 100% utilization, you should increase the number of threads until you achieve just under 100% utilization on the application server. And remember, the application server should achieve 100% utilization before the database server reaches 100% utilization. If this proves not to be true, something is probably wrong with an SQL statement and you must capture an SQL trace to determine the culprit.

Another way to achieve similar results is to start out with a small number of threads and increase the number of threads until you have maximized throughput. The definition of "throughput" may differ for each process but can be generalized as a simple count of the records processed in the batch run tree. For example, in the Billing background process in Customer Care and Billing/Enterprise Tax Management, throughput is the number of bills processed per minute. If you opt to use this method, it is recommended that you graph a curve of throughput vs. number of threads. The graph should display a curve that is steep at first but then flattens as more threads are added. Eventually adding more threads will cause the throughput to decline. Through this type of analysis you can determine the optimum number of threads to execute for any given process.

Use a third party batch scheduler

You can use an external scheduler to execute Customer Care and Billing/Enterprise Tax Management processes as well any other related (e.g. backups) and non-related process. Typically the introduction of such a scheduler allows a site to automate a lot of their processes to reduce manual effort.

The Customer Care and Billing/Enterprise Tax Management product does not have advanced batch-scheduling capabilities typically required by sites. All sites tend to use a third party tool for scheduling to manage Customer Care and Billing/Enterprise Tax Management jobs as well as “external” related jobs (such as backups). In choosing a batch scheduling tool that you can use with Customer Care and Billing/Enterprise Tax Management the following must be supported:

- **UNIX command line support** – The batch scheduler should be able to run a UNIX command line. An added bonus is the ability to parameterize the command line.
- **UNIX return code support** – The batch scheduler should be able to act upon the value of a standard UNIX return code. For example the process will return zero (0) for successful execution and non-zero for not successful.
- **Manage Job Dependencies** – The main advantage of a batch scheduler for Customer Care and Billing/Enterprise Tax Management is to manage the relationships between jobs and ensure that jobs are only started after dependant jobs are completed successfully.
- **(Optional) Limit the number of batch jobs executing at one time** – The ability for a batch scheduler to submit and control the number of active jobs is ideal to ensure the resources on a machine are maximized not exhausted. This is not a mandatory requirement but experience has shown that throughput is maximized if the scheduler manages the number of active jobs competing for the CPU.

Oracle Utilities Customer Care and Billing/Enterprise Tax Management V2.x and above includes an internal scheduler that can be used to submit background processes but it has the following limitations:

- It contains no failover facilities.
- It can only be run on one server only.
- It can only run Customer Care and Billing/Enterprise Tax Management jobs.

Customers have used the following tools successfully with Customer Care and Billing/Enterprise Tax Management⁷:

- **UC-4** - UC4 Software (<http://www.uc4.com/>)
- **BMC Control-M for Distributed Systems** - BMC Software (<http://www.bmc.com>)
- **CA-AutoSys** - Computer Associates (<http://www3.ca.com>)

⁷ There is no preferred solution as typically this software is used beyond just Customer Care and Billing/Enterprise Tax Management.

- **Tivoli Workload Scheduler** - IBM (<http://www-306.ibm.com/software/tivoli/products/scheduler/>)
- **COSBatch** - Open Ssystems Management Ltd. (<http://www.cosbatch.com/>)

DATABASE BEST PRACTICES

The Database Server is responsible for the storage and management of data. There are a number of practices that sites find useful for maintaining the health of the Database Server.

Resolve SQL Errors in Batch Jobs/Tuxedo/Hibernate

When a database error occurs in Customer Care and Billing/Enterprise Tax Management code the output from the error (including the full SQLCA) is output to the stdout file for the Tuxedo or background process. Examination of these files on a regular basis can help identify problems in the database and detect any configuration errors.

For V1.x, the online Tuxedo connections, the log files are located under **\$SPLSYSTEMLOGS/stdout*.txt**. For the batch they are located in **\$SPLAPP** (unless otherwise configured).

Regularly Calculate Database Statistics

When any table in the system grows (or shrinks) by a larger than normal rate, the access paths to that table may change causing inefficiencies. For the database to make the correct decision, it uses a set of statistics to assess all available paths. This is an important factor in performance. It is therefore recommended that database statistics be recalculated, using the **ANALYZE TABLE/DBMSTATS** (ORACLE) or **runstats** (DB2), on a regular basis to maintain up to date statistics.

The frequency will depend on the volume and size of your database. It is recommended that statistics most tables be calculated once a week at minimum unless their growth factors do not affect the path chosen by the DBMS.

A discussion on the statistics calculation is outlined in “**Oracle Utilities Customer Care and Billing/Enterprise Tax Management Performance Troubleshooting Guide**”.

Checking error logs can detect problems in the database before the end user reports them.

Database statistics are important for the performance of all SQL in the product. Keeping them up to date ensures the database has the most up to date information to make the appropriate access path decisions.

Ensuring I/O across devices is becoming less of a problem with progressive versions of the database handling this automatically and the introduction of intelligent SAN technology.

Ensure I/O is spread evenly across available devices

One of the key practices that is key to performance of a database is the elimination of “hot spots” in the disk architecture by ensuring that I/O is spread across all available devices. This is known as the “Database Topology”. For example, placing the database physical files on a single disk is not optimal as multiple concurrent requests queue to use the disk and would result in higher than expected disk wait times. By spreading the load across disks, the opportunity for wait times is minimized and increases throughput. It is therefore recommended that the disk architecture be designed for the physical database files so that as much I/O as possible is spread across all disks.

A discussion on the database topology and its implications is outlined in “**Oracle Utilities Customer Care and Billing/Enterprise Tax Management Performance Troubleshooting Guide**”.

Use the Correct NLS settings (ORACLE)

For non-western languages the NLS settings ensure the correct character set is shown.

One of the configuration settings that can affect the sorting and processing of date data is setting the correct language for the connection to the database. For ORACLE customers it is recommended that the NLS setting is correct for your region is set correctly at installation time. Refer to the NLS documentation (Globalization Support Guide) for ORACLE for details of the valid settings for your region.

*Note: Additionally for UTF8 customers ensure that the **tuxStringEncoding** (V1.x) in the **web.xml** file or **spl.runtime.cobol.encoding** (v2.x) in the **spl.properties** file, is set correctly to display the correct character set.*

Consider Partitioning

Partitioning Large tables can increase throughput significantly.

One of the features of the databases that Customer Care and Billing/Enterprise Tax Management supports is the ability to partition large tables onto “logical” tables to improve I/O. The “**Database Design Standards**” document in the SDK documentation outlines the tables that are candidates for using partitioning.

Note: The document mentions DB2 tables. The same applies to ORACLE implementations as well.

It is recommended that if partitioning is used that the number of partitions is set to a number greater than the maximum number of threads of any background processes that access the file. Threads of a background process should not cross partition boundaries as this can cause performance issues. Ensure all threads operate within a partition.

Keeping plans and packages in synchronization with the database statistics collection can result in efficient database access.

The AppViewer provides the data dictionary for Customer Care and Billing/Enterprise Tax Management. If you require the data model in a tool that can import the model from the database it is possible to generate the model.

Rebind regularly (DB2)

For DB2 customers, if the database statistics are rebuilt using **runstats**, the plan and collections MAY need to be rebound to reflect the updated statistics. Schedule the rebind as soon as possible after **runstats** has been executed.

Use the **rebind.sh** script to rebind the Customer Care and Billing/Enterprise Tax Management collections.

Building the Data Model

One of the common questions regarding Customer Care and Billing/Enterprise Tax Management is the availability of the total data model in a particular tool (such as Visio or similar). The Customer Care and Billing/Enterprise Tax Management product contains a large number of tables (> 170 as of V2.1) and it is generally impractical to display a full model due to its size (to make it legible).

There are a number of sources of information that can replace a full data model and present the data model information into “bite sized chunks”:

- The data model information is contained in the Data Dictionary component of the Application Viewer. With V1.x of Customer Care and Billing/Enterprise Tax Management it was pre-populated and with V2 it must be generated to view the full dictionary.
- The Conversion documentation, available in word as well as as online help contain a summary set of data models that basically outline the major entities in the product.
- Each of the Business Process manuals for the product outline the functionality and contain data models specifically for that component.

Why is there no referential integrity built into the database?

Note: The scripts in this section have been designed for ORACLE database only. Sites using DB2 or SQL Server should use the language equivalents of these scripts.

Typically referential integrity of a database is managed by the database itself. In Customer Care and Billing/Enterprise Tax Management this is not so as the business objects contain ALL the business logic including referential integrity. The reasons for this are varied:

- From a maintenance cost point of view, all the code is in one place. This reduces maintenance effort.
- Databases implement “all or nothing” referential integrity. This means that referential integrity is checked whether the data has changed or not. From a performance point of view this is potentially wasting time. The business objects in Customer Care and Billing/Enterprise Tax Management decide when to enforce referential integrity rules.

- Most of the referential rules in Customer Care and Billing/Enterprise Tax Management are optional. If there is a value in the foreign key field it is checked, if there is no value (blanks, zero or nulls) then the referential integrity is not checked unless it is a mandatory column. This is not possible in database imposed referential integrity.
- If the database controlled referential integrity then the application has no control on when it is imposed in the course of a transaction. Business object controlled referential integrity allows finer levels of control on when referential integrity is enforced in the transaction flow.
- Each database implements referential integrity in a slightly different way. To reduce maintenance costs, code differences are kept to a minimum.

Business Object enforced referential integrity is more efficient as far as Customer Care and Billing/Enterprise Tax Management is concerned and translates to superior performance across many database types.

Building the Data Model

All is not lost though. Customer Care and Billing/Enterprise Tax Management maintains its own data dictionary in the form of meta-data that is used by the Oracle Utilities Software Development Kit, ConfigLab and Archiving.

If you insist that you want the data model in a tool or adorning a large wall then the following is recommended process to be used to generate the data model using the meta-data:

- **Empty a copy of the database (truncated the tables).** None of the relationships are expressed as constraints in the physical database; this is because ALL the referential integrity (RI) and validation is done in the code based business objects. More importantly most of the constraints are data conditional (if there is data in the column, then RI applies; no value, no RI) so a loaded database might actually break “database strict” RI rules. Remove the data to prevent constraint violations using a valid method for the database (for example, **TRUNCATE TABLE... REUSE STORAGE** for ORACLE et al).
- **Create constraints from the meta-data structure.** The two ORACLE pl/sql scripts below can be used to achieve this. The names of the constraints is already documented in the meta data as well. Run the utility and created the constraints in the database.

Function to join

```
create or replace function join
(
    p_cursor sys_refcursor,
```



```

        p_del varchar2 := ','
    ) return varchar2
is
    l_value  varchar2(32767);
    l_result varchar2(32767);
begin
    loop
        fetch p_cursor into l_value;
        exit when p_cursor%notfound;
        if l_result is not null then
            l_result := l_result || p_del;
        end if;
        l_result := l_result || l_value;
    end loop;
    return l_result;
end join;
/
show errors;

```

Script to Create Constraints

```

set serverout on size 1000000
set echo off
set feedback off
set linesize 300
spool constraints.sql
declare
cursor c1 is
select tbl_name, CONST_ID, REF_CONST_ID, table_name from
ci_MD_CONST, user_indexes
where CONST_TYPE_FLG='FK'
and TRIM(index_name)=SUBSTR(REF_CONST_ID, 5, 7)
and TRIM(tbl_name) in (select TRIM(table_name) from
user_tables)

```

```

order by tbl_name, CONST_ID;
stmt varchar2(400);
field_list varchar2(300);
begin
for r1 in c1 loop
stmt := 'alter table ' || trim(r1.tbl_name) || ' add
constraint ' || trim(r1.const_id);
dbms_output.put_line(stmt);
select join(cursor(select trim fld_name from
ci_md_const_fld where const_id = r1.const_id order by
seq_num)) into field_list from dual;
stmt := 'foreign key (' || field_list || ')';
dbms_output.put_line(stmt);
select join(cursor(select trim(fld_name) from
ci_md_const_fld where const_id = r1.ref_const_id order
by seq_num)) into field_list from dual;
stmt := 'references ' || trim(r1.table_name) || ' ('
|| field_list || ')';
dbms_output.put_line(stmt);
end loop;
end;
/
spool off;
@constraints.sql
exit;

```

- **Load the data model in the tool of your choice.** Load the data model with the constraints in the desired tool. This should build the data model.

Note: This may take a while for the WHOLE data model.

- **Removed the newly created constraints.** This is to return the database back to the original condition.

```

set serverout on size 100000
set echo off
set feedback off
set linesize 300

```

```
spool drop_constraints.sql
```

```
select 'ALTER ' || tbl_name || ' drop constraint ' ||  
CONST_ID || ';' from ci_MD_CONST where  
CONST_TYPE_FLG='FK' order by tbl_name, CONST_ID;
```

```
spool off;
```

```
@drop_constraints.sql
```

```
exit;
```

- Reload the database. You then have the data model in your tool and the database returned to its original state.



Best Practices in IT Operations for Oracle Utilities Customer Care and Billing/Enterprise Tax Management
September 2007

Author: Anthony Shorten, Principal Product Manager, Utilities Global Business Unit

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Copyright © 2007, Oracle. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice.

This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle, JD Edwards, SPL WorldGroup, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.