



Web & Application Server Configuration for Contract Manager 12.0

This document contains confidential and proprietary information. The use, disclosure, reproduction, modification, transfer, or transmittal of this document for any purpose without the written permission of Primavera Systems is prohibited.

Table Of Contents

SUMMARY	2
WEB & APPLICATION SERVER MEMORY USAGE	3
DETERMINING PHYSICAL RAM REQUIREMENTS	3
ALLOCATING RAM TO THE CONTRACT MANAGER WEB & APPLICATION SERVERS	3
SERVER CONFIGURATION	6
<i>Running the Contract Manager Server as a service</i>	6
CONTRACT MANAGER CONFIGURATION	7
<i>Contract Manager Properties</i>	7
WEB SERVER CONFIGURATION	10
<i>Changing the Web Server Port Number</i>	10
<i>Contract Manager Server Port Number Usage</i>	10
<i>Web Server Logging</i>	10
APPLICATION SERVER CONFIGURATION	13
<i>Database Connection Configuration</i>	13
<i>Database Connection Pooling</i>	13
<i>Application Server Logging</i>	15
<i>Report Logging</i>	18
<i>Mail Configuration Issues</i>	18
CUSTOM DATABASE CONNECTION CONFIGURATION	19
<i>Port Assignment</i>	19
<i>Configuration</i>	20
<i>Port Range Determination</i>	21
SYSTEMS INTEGRATION	22
PRIMAVERA INTEGRATION	22
<i>Connecting Contract Manager to a Primavera Database</i>	22
<i>URL Connection</i>	23
BRAVA! [®] INTEGRATION	24

Summary

The purpose of this document is to provide details on how to change the configurable aspects of the Contract Manager Web and Application servers.

Architecture Changes for Contract Manager 12.0

Contract Manager 12.0 does not support viewing and printing reports using ActiveX and Applets anymore. Instead the reports would now be produced as .pdf documents which can be viewed and printed using Adobe Acrobat Reader. If Acrobat Reader is not present on the client machine then Contract Manager 12.0 would provide the option to download the generated .pdf files.

Contract Manager 12.0 ships with JBoss 4.0.5 as the application server.

The Contract Manager Web server accepts HTTP(S) requests and supplies HTTP(S) responses. It routes requests to the proper Java Servlet, which processes the request and passes it to the appropriate resource in the Contract Manager Application server. The response from the Application server is sent back to the Web Server and is directed to the appropriate Java Server Page (JSP). The Web Server uses the JSP and the data from the response to create the HTML page that is sent back to the requesting client.

Contract Manager uses Tomcat for a Web server. Tomcat is an open source Web server from the Apache organization. The Contract Manager installer will install a version of Tomcat that is integrated with the JBoss server. This means the Web & Application servers run in the same Java Virtual Machine (JVM). This improves performance because it eliminates the overhead encountered when the Web & Application servers have to communicate between separate JVMs. Because of this the Web and Application servers have to be installed on the same physical machine.

Contract Manager uses either JBoss or WebLogic for an Application server. JBoss is a high quality open source application server from the JBoss Group. JBoss is quickly becoming a dominant player in the application server market, winning Best Application Server from Java World Magazine in 2002. The Contract Manager installer will walk you through the process of installing this server. WebLogic is a Java-based application server from BEA Systems, Inc. The Contract Manager installer will not install WebLogic. It expects a WebLogic domain to be available that is specific to Contract Manager. There are also a number of other prerequisites for using Contract Manager with WebLogic. For more information regarding WebLogic prerequisites and installing Contract Manager with the WebLogic application server, see the "WebLogic Configuration" section of the Install.pdf file. The Application server pools database and system resources and provides a security layer so clients do not have direct connections to the database. The Application server also enforces a large percentage of Contract Manager's business rules and security settings ensuring the integrity of the data.

Web & Application Server Memory Usage

This section describes the how to determine the amount physical RAM that is required for the machine hosting the Contract Manager Web & Application servers. It also describes how to allocate the appropriate amount of RAM to those servers.

Determining Physical RAM Requirements

To achieve optimal performance and scalability the Contract Manager Web & Application servers need to have enough memory allocated to them to handle the maximum load that will be placed on them. The largest factors in determining the amount of memory required is the number of concurrent users and the total number of projects. The following chart gives the required physical RAM requirements based on concurrent users and total number of projects.

Web & Application Server Scaling		
Number of Concurrent Users *	Total Number of	
	Projects	Required RAM
1 - 10	< 100	512 MB
1 - 10	100 - 750	768 MB
1 - 10	750 - 1500	1 GB
1 - 10	> 1500	1.25 GB or more
11 - 50	< 100	768 MB
11 - 50	100 - 750	1 GB
11 - 50	750 - 1500	1.25 GB
11 - 50	> 1500	1.5 GB or more
51 - 100	< 100	1 GB
51 - 100	100 - 750	1.25 GB
51 - 100	750 - 1500	1.5 GB
51 - 100	> 1500	1.75 GB or more

* A note on concurrent users. The number of concurrent users means the number of people actively using the system at any one time.

Allocating RAM to the Contract Manager Web & Application Servers

Increasing the amount of physical RAM on the machine hosting the Contract Manager Web & Application servers will not substantially increase performance and/or scalability unless the amount of RAM made available to the Java Virtual Machine (JVM) is also increased.

The standard install will allocate 256MB of RAM to the JVM. If the Contract Manager Web & Application servers are the only applications running on a machine with 1GB of physical RAM, with the standard install, the Web & Application servers will only be able to use 256MB of RAM and about 500MB of RAM will be sitting idle.

An installation that is experiencing performance problems or just wants to maximize performance should adjust the JVM RAM setting according to the total amount of available physical RAM on the host machine.

As a general rule if the Contract Manager Web & Application servers are the only applications running on a machine then the amount of JVM RAM should equal the total physical RAM minus 256MB for the O/S.

For example:

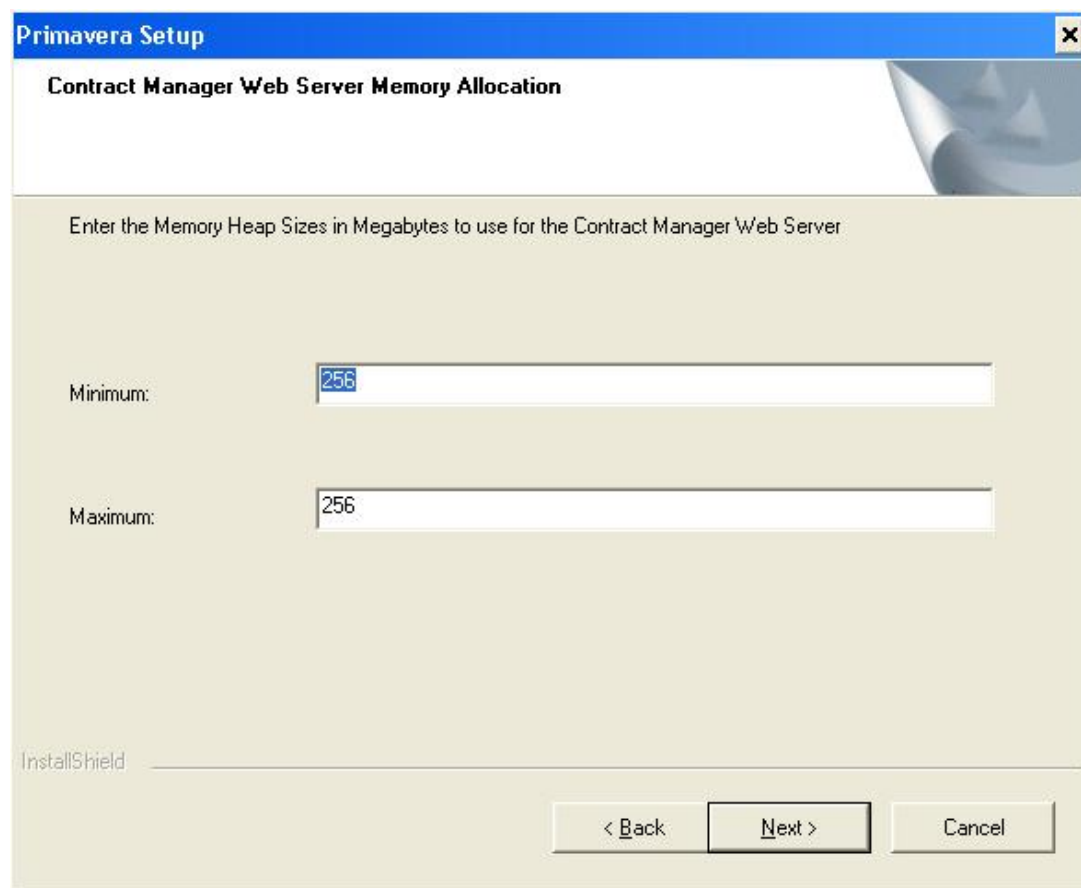
Physical RAM	1GB
Minus memory for the O/S	<u>-256MB</u>
Recommended JVM RAM	768MB

If another application is running on the same machine as the Contract Manager Web & Application servers then memory has to be available for that application.

For example:

Physical RAM	1GB
Minus memory for the O/S	-256MB
Minus memory for other application	<u>-128MB</u> (this will differ depending on what the application is)
Recommended JVM RAM	640MB

You can adjust the JVM RAM setting in the Contract Manager Server Configuration utility program. There is a minimum and maximum setting. It is advised to keep these numbers the same.



Server Configuration

Configuration files are used to control different aspects of the Web & Application server's environment and how the servers behave. Listed here are the most commonly used configuration files for the Contract Manager Web & Application servers.

Configuration Files		
File Name	Location	Purpose
run.bat	Primavera\ContractManager\Jboss\bin	Sets environment variables and starts the servers
exponline.properties	Primavera\ContractManager\Jboss\server\Contract Manager\lib\ext\com\ primavera\exponline\common	Configures the how Contract Manager uses the Web & Applications servers
log4j.xml	Primavera\ContractManager\Jboss\server\Contract Manager\conf	Defines how the servers display and log auditing information
expedition-ds.xml	Primavera\ContractManager\Jboss\server\Contract Manager\deploy	Configuration of database connection pools
conf\jboss-service.xml	Primavera\ContractManager\Jboss\server\Contract Manager\conf	JBoss configuration settings
server.xml	Primavera/ContractManager/jboss\server\Contract Manager\deploy\jbossweb-tomcat50.sar	Tomcat configuration settings

Running the Contract Manager Server as a service

Contract Manager 12.0 will install the Contract Manager server as an automatic service. Being a service, there is not a console displayed for this server. The messages that were being displayed on the console are now being written to the Primavera/ContractManager/expconsole.log file.

Steps for running the server from the desktop instead of as a service

1. Stop the Contract Manager Service and change the Startup Type to "Manual".
2. Start the server by running the Primavera/ContractManager/expedition.bat file.

Contract Manager Configuration

This section describes configuration changes that are made to the Contract Manager properties file.

Contract Manager Properties

The tokens below are values obtained from the user via the installer and entered in the `exponline.properties` file by the installer:

Tokens obtained from the Installer	
Token	Description
\$\$DOMAINNAME\$\$	The value of the Contract Manager Web Server Computer name
\$\$INSTALLPATH\$\$	The value entered by the installer in the Destination folder for the Contract Manager Web Server e.g. <code>c:\primavera\expwebserver</code>
\$\$COMPANYABBREV\$\$	Up to 4 characters of the value entered by the installer in the Company Name field.
\$\$SMTPSERVER\$\$	The value entered by the installer in the SMTP Server field
\$\$EXPDBTYPE\$\$	The value entered by the installer when selecting the Contract Manager database type, e.g. Sybase or Oracle.
\$\$P3EDBTYPE\$\$	The value entered by the installer when selecting the P3e database type
\$\$REPORTIMAGEFOLDER\$\$	The value entered by the installer.

The following table lists the purpose of each of the properties in the `exponline.properties` file.

Tokens obtained from the Installer	
Property	Description
<code>WebServerName=\$\$DOMAINNAME\$\$</code>	This value is this computer name, e.g. <code>EXPSEVER</code>
<code>WebApplicationName=exponline</code>	This is the identifier that is used in the Contract Manager URL. For Contract Manager 9.0 this value must be "exponline" without the quotes
<code>WebProtocol=http</code>	Defines the protocol used to communicate to the Web Server, HTTP is the default. See separate document, <i>Implementing HTTPS in Jetty</i> , for instructions on how to use HTTPS.
<code>WebPort=80</code>	This sets the port number used by the Contract Manager Web Server. Port 80 is the default, to change this value see instructions later in this document.
<code>InstallPath=\$\$INSTALLPATH\$\$</code>	Value entered via Installer on where the Web Contract Manager was installed.
<code>admindefault=jdbc/expadminPoolDS</code>	Used to define the Expadmin DB connection. Do not change this value.
<code>DatabaseSiteName=\$\$COMPANYABBREV\$\$</code>	This up to a 4 character string used to make the primary keys at each site of a company unique.
<code>ServerContext.INITIAL_CONTEXT_FACTORY=</code>	This property is for future use. A blank value in is used for the JBoss server. For another type of application server this value would be changed.
<code>ServerContext.PROVIDER_URL=</code>	This property is for future use. A blank value in is used for the JBoss server. For another type of application server this value would be changed
<code>LookUpQualifier=</code>	This property is for future use. A blank value in is used for the JBoss server. For another type of application server this value would be changed

JDBCLookUpQualifier=java:/	This is the prefix used for JDBC lookups. For this release the value must be java:/.
PrintDebugLevel=0	This is flag used to determine how much debug information will be displayed to the console and written to the log. A value of 0 means no debug information will be displayed. A value of 6 shows all debug information.
SMTPServer=\$\$SMTPSERVER\$\$	Value is the user's SMTP Server for outgoing mail.
DatabaseType=\$\$EXPDBTYPE\$\$	Specifies the type of DB used for Contract Manager. The valid values are either "sybase", "oracle" or "mssql" without the quotes.
P3eDatabaseType=\$\$P3EDBTYPE\$\$	Specifies the type of DB used for Primavera
xmlsessiontimeout=3600	This value defines how long the XML API can sit idle before the session is timed out and the user has to log back in. The value is in seconds the default is 1 hour.
sessiontimeout=3600	This value defines how long the browser client can sit idle before the session is timed out and the user has to log back in. The value is in seconds the default is 1 hour.
CharacterEncoding=UTF-8	Must be blank or UTF-8, both provide the same result.
ReportImagesLocation=\$\$REPORTIMAGEFOLDER\$\$	This folder is the location of the image files used by the Reports & Forms.
Standalone=yes	A "yes" value is set when Contract Manager Clients will be run from a browser on this Server computer. A "no" value is set when Contract Manager Clients will be run from a browser from other computers.
GridToExcel= \$\$INSTALLPATH\$\$jboss-3.2.5/server/ContractManager/lib/ext/com/primavera/exponline/client/gridtoexcel.xsl	This value is the full path to the installed gridtoexcel.xsl file.
CompressionEnabled	Setting this to true allows JSPs served by application server to be compressed using GZIP compression.
app_server_name	Values can either be jboss or weblogic
database_port	Database Server Port Number e.g. 2638 for Sybase
database_host_name	Database Server Machine Name or IP
WebLogicDomain_Home	Weblogic Domain Home Path e.g C:\bea\user_projects\domains\CMDomain
BravaServerName	The machine name the Brava server is on
BravaServerPort	The port number for the Brava server
BravaMarkups	The file directory where Brava is storing its markups
BravaClient	The type of brava viewer that should be used to view the files. Valid values are activex or applet. You should use activex when the installation is BravaX and applet when the installation is BravaJ.
SybaseStartingClientPort	This defines the first client port number. All subsequent client port assignments will be made consecutively from this number.
SybaseTotalPossibleConnections	This number represents the total of the maximum number of connections of all of the connection pools.
SybaseMinimumNumberOfConnections	This represents the total of the min-pool-size for each connection pool defined for JBoss.

SybaseConnectionsDebugFile

This property will turn on connection debug logging. It will create the specified file and log all connections and the Socket cleanup operations.

Web Server Configuration

This section describes some of the configurations that can be made to the Contract Manager Web Server.

Changing the Web Server Port Number

To change the port number that the Web Server listens to for HTTP traffic is to run the Server Configuration Utility and enter the new port number in the Port number dialog screen. Weblogic Ports can be configured using WebLogic admin console e.g. `http://<machine_name>:7001/console`

Contract Manager Server Port Number Usage

This table lists the default ports used by the Contract Manager servers and where these ports are configured:

Port Usage		
Port Number	Usage	Configured In
25	SMTP	exponline.properties
80	HTTP	Contract Manager Server Configuration Utility
443	HTTPS	server.xml
1098	JNDI RMI	confjboss-service.xml
1099	JNDI JNPService	confjboss-service.xml
1521	Oracle Port	expedition-ds.xml
2638	Sybase Database Server	expedition-ds.xml
4444	RMI Object Port – Using JRMP	confjboss-service.xml
4445	JBoss Pooled Invoker	confjboss-service.xml
7001	WebLogic default HTTP Port	Weblogic Ports can be configured using weblogic admin console e.g <code>http://<machine_name>:7001/console</code>
7002	WebLogic default HTTPS Port	Weblogic Ports can be configured using weblogic admin console e.g <code>http://<machine_name>:7001/console</code>
8009	Apache Coyote AJP Connector	server.xml
8083	JBoss Class Loading	confjboss-service.xml
N + # of DB Connections	Sybase client connections, 1 per DB connection	This range is assigned randomly by Sybase or is configurable. See Custom Database Socket Configuration for configuration details.

Web Server Logging

** This section is applicable to JBoss application server only. Weblogic logs can be configured using WebLogic admin console e.g. `http://<machine_name>:7001/console`*

This logs all of the requests made to the Contract Manager server. It contains data on when the requests were made, the TCP addresses that accessed the server, what the requests were, what the results codes were, and the pages that were served.

The details of how this information is logged are configured in the server.xml file:

```
<!-- Access logger -->  
  <Valve className="org.apache.catalina.valves.AccessLogValve"  
    prefix="localhost_access" suffix=".log"  
    pattern="common" directory="{jboss.server.home.dir}/log"/>
```

Values for the **pattern** attribute are made up of literal text strings, combined with pattern identifiers prefixed by the "%" character to cause replacement by the corresponding variable value from the current request and response. The following pattern codes are supported:

- **%a** - Remote IP address
- **%A** - Local IP address
- **%b** - Bytes sent, excluding HTTP headers, or '-' if zero
- **%B** - Bytes sent, excluding HTTP headers
- **%h** - Remote host name (or IP address if resolveHosts is false)
- **%H** - Request protocol
- **%l** - Remote logical username from identd (always returns '-')
- **%m** - Request method (GET, POST, etc.)
- **%p** - Local port on which this request was received
- **%q** - Query string (prepending with a '?' if it exists)
- **%r** - First line of the request (method and request URI)
- **%s** - HTTP status code of the response
- **%t** - Date and time, in Common Log Format
- **%u** - Remote user that was authenticated (if any), else '-'
- **%U** - Requested URL path
- **%v** - Local server name

The shorthand pattern name "common" corresponds to **%h %l %u %t "%r" %s %b**".

This functionality is fully documented on the Apache site:

<http://jakarta.apache.org/tomcat/tomcat-5.0-doc/config/valve.html#Access%20Log%20Valve>

Application Server Configuration

Database Connection Configuration

Once Contract Manager is installed on a machine there are three utility programs available to assist you with configuring connections to your database server.

Database Connection Configuration Utilities	
Utility	Description
CA.exe	This utility can be found in the database folder of the Contract Manager CD1. This utility can be used to create a new database group, upgrade an existing database group or convert an existing database group to use a different database type,
Server Configuration	Captures the following information about the database server and updates the appropriate Contract Manager sever files: <ul style="list-style-type: none"> • Server type (Sybase or Oracle) • Machine name • Port number • Password
Set Database Connection	Reads the populategroups.bat file created by the Server Configuration utility and updates appropriate Contract Manager server files with the information.

Database Connection Pooling

** This section is applicable to JBoss application server only. WebLogic connection pooling can be configured using WebLogic admin console e.g. http://<machine_name>:7001/console*

When the Application server starts it creates pools of connections for each database the Contract Manager Application server is using. Database connection pools are used so that database connections are shared across users. Each user doesn't have to make a dedicated connection to the database. A pool of connections is made and when a request needs a connection it uses one from the pool and returns it to the pool when the request has completed.

The size of the connection pool is dependent on the number of concurrent users for a each particular database. Remember all users need to connect to the ExpAdmin database, but all users may not be connecting to all of the groups. There doesn't have to be a one to one relationship between concurrent users and database connections but you should allow the size of the pool to grow to that size if it needs to.

For example:

For 50 concurrent users, 10 using Group A and 40 using Group B:

ExpAdmin Pool Min Size: 25
ExpAdmin Pool Max Size: 100

Group A Pool Min Size: 5
Group A Pool Max Size: 20

Group B Pool Min Size: 25
Group B Pool Max Size: 80

This configuration sets the minimum for each pool to be less than the number of concurrent users, but allows the number of connections to grow to twice the number of concurrent users in case some request is using multiple connections.

The actual amount of connections kept alive should never be less than the minimum size and will grow and shrink up to the maximum size setting depending on usage.

As a rule of thumb it is better to allocate more connections needed than less. If a user has to wait for a connection because all connections in the pool are being used and the pool is at max size, then the user will experience a performance delay.

The pools are defined in the expedition-ds.xml file. The installer will run the Server Configuration utility will run the Set Database Connections utility which will overwrite this file. The default pool sizes are minimum of 25 and maximum of 75. These are generous settings, but it shouldn't be a problem unless you are short on resources on your Web & Application or database servers.

This is an example of the XML in the expedition-ds.xml file that defines a connection pool for the ExpAdmin database. **If you change any of these values directly in this file, you will need to re-adjust them every time you run the Server Configuration utility. The utility removes all entries from the file then polls the database server for the groups available and adds an entry for each group with the default settings.**

```
<datasources>
  <local-tx-datasource>
    <jndi-name>jdbc/EXPADMIN</jndi-name>
    <driver-class>com.sybase.jdbc2.jdbc.SybDataSource</driver-class>
    <connection-url>jdbc:sybase:Tds:serverName:2638?ServiceName=expadmin</connection-url>
    <user-name>exp</user-name>
    <password>sql</password>
    <min-pool-size>25</min-pool-size>
    <max-pool-size>75</max-pool-size>
  <idle-timeout-minutes>20</idle-timeout-minutes>
    <exception-sorter-class-name>
      org.jboss.resource.adapter.jdbc.vendor.SybaseExceptionSorter
    </exception-sorter-class-name>
  </local-tx-datasource>
```

This XML also defines that all connections that have been idle for more than 20 minutes should be refreshed. This is done so that connections are refreshed before the database decides a connection is idle and disconnects it. The Sybase DB default is 4 hours (-ti setting).

Application Server Logging

** This section is applicable to JBoss application server only. Weblogic logs can be configured using WebLogic admin console e.g. http://<machine_name>:7001/console*

The log4j.xml file governs the logging in JBoss. It configures where information is logged to, the level of information that is logged, and how the logs are maintained. Log4j is an open source logging utility available from Apache.

This file defines two root categories, "CONSOLE" and "FILE". These categories control what is logged to the server console and log files respectively. The amount of information that is output to these categories is controlled by the logging levels.

Logging Levels

Log4j Logging Levels	
Level	Description
ALL	The ALL has the lowest possible rank and is intended to turn on all logging.
DEBUG	The DEBUG Level designates fine-grained informational events that are most useful to debug an application.
ERROR	The ERROR level designates error events that might still allow the application to continue running.
FATAL	The FATAL level designates very severe error events that will presumably lead the application to abort.
INFO	The INFO level designates informational messages that highlight the progress of the application at coarse-grained level.
OFF	The OFF has the highest possible rank and is intended to turn off logging.
WARN	The WARN level designates potentially harmful situations.

The product is shipped to log in WARN mode. This means that only exceptions and warning messages will be logged. The logging level can be configured in the log4j.xml file.

Console Logging

When the Contract Manager server is run as a service the console output gets written to the Primavera/Contract Manager/expconsole.log file.

The following XML controls the logging to the console.

- The logging level is represented in the "Threshold" tag.
- The layout of the console log is Time : Level : Category : Message. To see a complete description of layout configuration options, look at the PatternLayout class documentation at this address:

<http://logging.apache.org/log4j/docs/api/index.html>

```
<!-- ===== -->
<!-- Append messages to the console -->
<!-- ===== -->

<appender name="CONSOLE" class="org.apache.log4j.ConsoleAppender">
  <errorHandler class="org.jboss.logging.util.OnlyOnceErrorHandler"/>
  <param name="Target" value="System.out"/>
</appender>
```

```

    <param name="Threshold" value="WARN"/>
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%d{ABSOLUTE} %-5p [%c{1}] %m%n"/>
    </layout>
  </appender>

```

File Logging

The following XML controls the logging to the console.

- The logging level is represented in the "Threshold" tag.
- When the size of the log file reaches the MaxFileSize it will create a new file.
- MaxBackupIndex indicates howmany backup files will be kept before it starts overwriting existing files.
- The layout of the console log is Date: Time : Level : Category : Message. To see a complete description of layout configuration options, look at the PatternLayout class documentation at this address:

<http://logging.apache.org/log4j/docs/api/index.html>

```

<!-- ===== -->
<!-- Preserve messages in a local file -->
<!-- ===== -->

<!-- A size based file rolling appender -->
<appender name="FILE" class="org.jboss.logging.appender.RollingFileAppender">
  <errorHandler class="org.jboss.logging.util.OnlyOnceErrorHandler"/>
  <param name="File" value="${jboss.server.home.dir}/log/server.log"/>
  <param name="Append" value="false"/>
  <param name="MaxFileSize" value="5120KB"/>
  <param name="MaxBackupIndex" value="1"/>
  <param name="Threshold" value="WARN"/>
  <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="%d %-5p [%c] %m%n"/>
  </layout>
</appender>

```

Contract Manager Debug Information

To see debug messages from the Contract Manager application running on JBoss:

- The output level should be changed to INFO
- The PrintDebugLevel in exponline.properties should be set to 6.

NOTE: Turning on all debugging information will create very large log files and will cause some performance degradation.

Boot Log

When the Contract Manager server is started the server startup information is written to the Primavera/ContractManager/boot.log file.

Report Logging

The Contract Manager server can create a Report Log that will contain debug information about every "report" that the server processes. Reports means any process that renders a report or form created by InfoMaker. That includes reports, forms, and PDFs that are e-mailed.

To create this log, add the following line to the exonline.properties file:

```
ReportLogPath=c:/reportlogs
```

C:/reportlogs must be a valid directory.

Mail Configuration Issues

An issue faced with configuring SMTP mail to work with Contract Manager is Corporate Mail/Spam filters that block TCP/IP addresses without a successful reverse domain lookup.

Some email relays might reject mail from Contract Manager in the case where the reverse DNS lookup fails. The solution for this is to add the Contract Manager IP address to the access list on the SMTP relay or to have a valid DNS entry for the Contract Manager server.

Custom Database Connection Configuration

This functionality is for Sybase configurations on JBoss application server only.

This section describes how to use a custom Contract Manager Socket factory for each connection JBoss makes to the Sybase DB. The advantages of using this custom factory are:

1. Allows customers to define a range of port numbers for these connections.
2. Allows connection audit information to be turned on.

Port Assignment

When this custom factory is enabled every request by JBoss to create a connection to the Sybase DB will be intercepted by our custom Socket factory. This Socket factory will create a Socket with the client port assigned within a specified range. This specified range is determined by settings in our properties file.

When a socket is created, the Socket factory keeps a pointer to it and remembers which port was assigned to it. Every 10 minutes a scan of the Sockets is executed to look for closed Sockets. If a closed Socket is found then the associated port number is released and available for another connection.

This illustrates how ports are assigned for 5 DB connections:

Initial Creation of Connection pool

Create Socket on port 4000	Socket created successfully on port 4000	
Create Socket on port 4001	Socket created successfully on port 4001	
Create Socket on port 4002	Socket created successfully on port 4002	
Create Socket on port 4003	Port 4003 is in use by other program	Socket created
successfully on port 4004		
Create Socket on port 4005	Socket created successfully on port 4005	

Refresh of Connections by JBoss

Create Socket on port 4006	Socket created successfully on port 4006
Create Socket on port 4007	Socket created successfully on port 4007
Create Socket on port 4008	Socket created successfully on port 4008
Create Socket on port 4009	Socket created successfully on port 4009
Create Socket on port 4010	Socket created successfully on port 4010

Release port 4000

Release port 4001

Release port 4002

Release port 4003 * This is releasing our pointer to this port because we are not holding an open connection to it. This will allow subsequent connection attempts to try this port again. This is particularly important when the server is cycled quickly and the old connections are still holding ports open.

Release port 4004

Release port 4005

Configuration

To turn this functionality on you have to modify two configuration files:

expedition-ds.xml
exponline.properties

expedition-ds.xml

This file defines the JBoss database connection pools. The following line has to be added to each datasource definition:

```
<connection-property name="SYB SOCKET_FACTORY">  
    com.primavera.exponline.server.base.SybaseSocketFactory  
</connection-property>
```

exponline.properties

Mandatory Configuration Settings

The following properties have to be added to enable this functionality:

SybaseStartingClientPort=4000

This defines the first client port number. All subsequent client port assignments will be made consecutively from this number.

SybaseTotalPossibleConnections=150

This number represents the total of the maximum number of connections of all of the connection pools. For example if you 2 connection pools defined in expedition-ds.xml and both have the following pool sizes:

```
<min-pool-size>25</min-pool-size>  
<max-pool-size>75</max-pool-size>
```

Then the total possible connections is 150 (75 * 2)

Optional Configuration Setting

SybaseMinimumNumberOfConnections=50

This represents the total of the min-pool-size for each connection pool defined for JBoss. This enables debugging for minimum pool size settings.

SybaseScanInterval=10

This set how often the Socket scan is run. The number is in minutes, 10 is the default value. NOTE: The Socket scan interval value has to be less than the

```
<idle-timeout-minutes>20</idle-timeout-minutes>
```

setting for each datasource in the expedition-ds.xml file.

SybaseConnectionsDebugFile=C:/primavera/debug/connections.txt

This property will turn on connection debugging. It will create the specified file and log all connections and the Socket cleanup operations. *This file should only be used for debug purposes and not for long-term production use.*

If the total number of open Sockets (DB Connections) drops below the minimum defined in the property, a warning message will written to the connections debug file:

Thu Jun 03 13:14:09 EDT 2004 WARNING Minimum Connection Count =50 Current number of Open Connections=20

This check will be performed every time the Socket scan is done (default is every 10 minutes).

NOTE: The path specified for this file must be a valid path and the user the Contract Manager server is running under must have privileges to create a file on this path. The Contract Manager server will not start correctly if either of these conditions is not met.

Port Range Determination

This is how the port range is determined:

Starting Port	SybaseStartingClientPort	4000
Number of ports needed	SybaseTotalPossibleConnections	$(150 * 2) + 5$
Number of ports reserved		305
Port Range		4000 - 4304

The number of connections is doubled to allow new ports to be assigned before the old ones are released. The 5 extra ports are included to allow for up to 5 ports in the specified range to be in use by other programs.

Systems Integration

This section reviews configuration to integrate with other software systems.

Primavera Integration

This section describes the integration points Contract Manager has with Primavera and the underlying configuration needed to make them work. There are two types of interfaces to Primavera:

P3e Integration Points	
Integration Type	Description
Direct Connection to the Primavera Database	A connection is made directly to the Primavera database and information is passed between the two systems. You can see the field mapping for this interface in an install document titled "Primavera Integration Field Mappings".
URL Connection	A Primavera URL is called from the Contract Manager client and Primavera pages are displayed. This interface requires the direct database connection parameters to be populated.

The Contract Manager and Primavera systems are installed separately and do not share any resources except the possibility that they could both utilize the same Oracle or MSSQL database server.

Connecting Contract Manager to a Primavera Database.

When installing Contract Manager the user will be prompted for parameters to make a connection to the P3e database. These parameters will be used to make a direct connection to the Primavera database and are used for the URL connection.

Primavera Database Connection Parameters	
Parameter	Description
Service Name	The name of the Primavera MS SQL DB Server, or for Oracle, the SID. Example: P3E35 or P3EORA9, respectively.
Machine	The name of machine where the Primavera Database is located. It can be the machine name or a URL.
DB Name	The Primavera DB name. For Oracle, this is usually the same as the SID, for MS SQL Server it is the value in the jdbc connection url following "databasename=" in the myPrimavera configuration.
Username and Password	The username and password to get admin or privileged access to the Primavera Server.

These parameters will be used by the installer to create an entry in the expedition-ds.xml file. This entry will create a database connection pool for the Primavera connections. Here is an example of the expedition-ds.xml entry for JBoss application server:

```
<local-tx-datasource>
  <jndi-name>jdbc/PRIMAVERASDK_PE</jndi-name>
  <driver-class>oracle.jdbc.driver.OracleDriver</driver-class>
  <connection-url>jdbc:oracle:thin:@servername:1521:polaris4</connection-url>
  <user-name>uid</user-name>
  <password>psw</password>
  <min-pool-size>6</min-pool-size>
  <max-pool-size>6</max-pool-size>
  <exception-sorter-class-name>
    org.jboss.resource.adapter.jdbc.vendor.OracleExceptionSorter
  </exception-sorter-class-name>
</local-tx-datasource>
```

The highlighted text above should be replaced by the parameters entered in the installer.

To tie an Contract Manager project to a Primavera project the "Schedule" field must be updated in the Project Settings dialog for each Contract Manager project.

URL Connection

The URL connection to Primavera allows Contract Manager users click on a Primavera URL from within Contract Manager. This will create a separate browser instance and make a URL connection to Primavera. The UID & PSW from the Contract Manager user must match exactly to the UID & PSW of the Primavera user for this connection to succeed. Also, the Primavera database connection parameters must be populated for the URL connection to succeed.

There is one connection parameter, for the URL connection, entered in the installer:

P3e URL Connection Parameters	
Parameter	Description
Primavision URL	The name of the machine hosting the Primavision app, and usually includes the port number.

The following information is posted to the Primavision server to make a URL connection:

Parameters passed to the Primavision server	
Parameter	Description
User Name	Contract Manager's user name
Password	Encrypted Contract Manager's password
Language	This will be standard 2 character language code. (en, fr, de,zh etc.)
Project Id	This is a number uniquely identifying a project in the Primavera database
Database	Computer name of the machine on which database is running, Primavera Database Server port number, and Primavera Service Name

Dbname	The name of the MSSQL database or Oracle SID that contains the corresponding project and user information.
Dbhost	Name of the machine hosting the database. This must match the host value from the connection string for Oracle and the host value preceding the port number in the case of MSSQL. MSSQL eg., jdbc:JSQLConnect://odessa:1433/database=EXPADMIN Odessa is the dbhost Oracle eg., jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=38.150.137.14)(PORT=1521)))(CONNECT_DATA=(SID=alxd06))) 38.150.137.14 is the dbhost

Brava!® Integration

Contract Manager 12.0 supports both BravaX 5.3 and BravaJ. The ability to choose BravaX or BravaJ for integration is provided by BravaClient property in the exponline.properties file. If this property is not specified and other Brava integration properties are present in the exponline.properties file, Contract Manager would assume integration with BravaX 5.3 and try to launch the Brava activeX control when attempting to view attachments. To integrate Contract Manager with a Brava! server, the following properties must be added to the exponline.properties file (default location `\Primavera\ContractManager\jboss\server\expedition\lib\ext\com\primavera\exponline\common`):

Contract Manager Properties for Brava Integration

Property	Description
BravaServerName	Name of the machine that is running the Brava! Server
BravaServerPort	Port number that Brava!'s instance of Tomcat is listening on
BravaMarkups	The file directory where Brava! Is keeping its markups
BravaClient	Indicates what type of Brava installation to use. Valid values are "activex" and "applet".

Example:

```
BravaServerName=machinename
BravaServerPort=8080
BravaMarkups=\\machinename\markup
BravaClient=activex
```

The BravaMarkups property must use a UNC path to point to the markups directory. As well, if the Contract Manager Web Server is running as a service, it must be modified to run as a domain user account with access to the Brava! markups directory. The Brava! markups directory must be shared.

Notes:

- The Contract Manager Web Server does not validate Brava! values added to exponline.properties. If the values are not accurate, Brava! accessibility from Contract Manager will be affected.

- The Contract Manager Web Server needs to be restarted before the integration will take effect.

For more information on using Brava! with Contract Manager, refer to the Brava! Integration Overview in online help.