Apelon Distributed Terminology System (DTS)

DTS Server Operations Guide

Table of Contents

Introduction

This guide describes the server administration, server configuration, and log file configuration processes required to set up and run the Apelon Distributed Terminology System (DTS). The guide is intended to help new and experienced System Administrators and support personnel perform all operations necessary to configure and maintain the Apelon DTS Server. A section on the **Tomcat Server**, which supports the DTS Browser client, also is included.

Apelon DTS Server Overview

Apelon DTS is a client/server solution that provides standalone and embeddable runtime access to terminology. Apelon DTS integrates standard vocabulary, including local enhancements, into an enterprise healthcare-computing environment. DTS establishes connections between a medical terminology and the conceptual framework on which it is built. Essential concepts of clinical descriptions from one application can be identified, then related to similar concepts from other information systems.

The client provides a collection of Java classes, each of which implements a well-defined and consistent subset of the functionality (e.g., searching over synonyms, taxonomic navigation, class-based queries, code translations, etc.). Each of these classes communicates by use of XML with server side components called QueryServers. The QueryServers are loaded on demand by the core ApelonServer, and are provided with resources such as JDBC-based connections as needed.

The core ApelonServer is configurable to support different usage scenarios. Worker threads, and all server resources in general, are pooled for reuse when possible. The numbers of connections allowed, sizes of thread pools, cache size, etc. all are configurable.

For example, a particular use case may require only navigation over taxonomies to support tree widgets in a browser-based application, with a small number of connections needing to execute large numbers of queries. In this case the server can be configured with a small thread pool, and only one QueryServer, the NavigationServer, loaded at startup. Another scenario might involve large volumes of connections looking up codes for clinical ordering transactions. In this case a small connection pool and a large thread pool could support a great number of simultaneous connections.

The server also provides a configurable logging facility, and a Command Line utility for checking the server's health and shutting it down remotely. The server is configured at startup through an XML properties file.

On the client side, different connection types are supported in a manner that abstracts away the details and allows the same Java classes to be used. In addition to the normal socket-based connection, there is a transient socket connection, a JDBC connection, and a secure connection, which requires user name and password.

The JDBC connection allows the server to be run in the same JVM. This is useful, for example, when running servlet based applications that may need to access DTS as a client running on the server side. If the database is on the same server this configuration may yield better performance.

DTS Server Startup

Start the Apelon DTS Server

After the DTS Knowledgebase has been prepared, but before any server administration can be performed, you must start the Apelon DTS Server. For a *Windows* installation, select **Start Apelon DTS Server** from the **Start** menu (**Start>Programs>Apelon> DTSInstall>Start Apelon DTS Server**) (where *DTSInstall* represents the folder name used when DTS was installed). For a Linux installation, execute ***StartApelonServer.sh*** in **bin/server** to start the Apelon DTS Server. The server starts automatically, after which you can run any of the other Apelon DTS components.

Note: If DTS Browser users are connecting to the database through a **Socket** type connection, the Apelon DTS and the Tomcat Servers both must be running. Only the Tomcat Server need be running for DTS Browser users connecting to the database through a **JDBC** connection. Note: When the modular classifier is used for modular classification, then only the Tomcat Server needs to be be running; refer to the *Ontylog Extension Namespaces and Extension Namespace Classification in DTS* document for more on modular classification.

```
Starting Apelon Server on 192.168.1.35:6666
Initializing socketServer
The validation of XML parser is on
RSS: Listening on port: 6666
Initializing adminServer
```

For AIX installations, navigate to **DTS/bin/server** and enter **StartApelonServer**.

Change the Apelon DTS Server Port Number, if Necessary

When you run the Apelon DTS Server, the server reads the **apelonserverprops.xml** file for the user name, password, host name for the Oracle database, and port number on which the server will listen. If you run the StartApelonServer command, and another program is using the port, you can change the default port number by editing the **apelonserverprops.xml** file.

1. Log into the machine directly or remotely.

2. Ensure you are in the **bin\server** subdirectory, where the **apelonserverprops.xml** file is located.

3. To edit the apelonserverprops.xml file, enter **vi apelonserverprops.xml** at the prompt. Note that **<property name="port"** has a default value of **"6666"**.

```
<apelprops>
    <property name="qs" value="com.apelon.dts.server.QueryServer" />
    <property name="pass" value="apelon" />
    <property name="user" value="apelon2" />
    <property name="host" value="192.168.1.4" />
    <property name="port" value="6666" />
    <property name="maxDbConns" value="15" />
    <property name="log" value="ApelonServerLog.xml" />
</apelprops>
```

4. Change the current value of **6666** to a new port number. For insert mode, enter **i**.

5. Save the file, and enter **:wq!**.

Server Shutdown

To shut down the DTS Server for a *Windows* installation, click **Close** in the **Start Apelon DTS Server** display window. To shut the DTS Server down for a Linux installation, type **CTRL-C** (if the DTS Server is running in the foreground, you can execute **ShutdownApelonServer.sh** from a different terminal window to shut it down).

Apelon DTS Server Configurations

Overview

Access to the Apelon DTS Server and the associated query servers is based on the configurations in the **apelonserverprops.xml** file (*DTSInstall***bin****server**). Within this file are properties and configurable values for accessing a database using Apelon DTS. **You must edit the server properties before starting the Apelon DTS Server**.

The Apelon DTS Server supports multiple query servers. The **apelonserverprops.xml** file includes properties and configurable values for running the DTS Server and sending queries to all of these servers. You can configure just the servers that your application(s) will need; if you will not use the services provided by one or more of the query servers, you should remove these sections from the **apelonserverprops.xml** file.

The client must direct the queries to the desired server through addition of a Header to each query. There is a matching rule between Headers and query server class; if the header is not specified, the queries will be sent to a default query server, which is the **QueryServer**.

The following table maps the query server that is required to service the DTS API query class(es) in an application.

Query Server	Header	Query Client
com.apelon.dts.server.QueryServer	DTS:001	com.apelon.dts.client.ConceptServer com.apelon.dts.client.SearchServer
com.apelon.dts.server.MatchServer	DTS:004	com.apelon.dts.client.MatchServer com.apelon.dts.client.match.MatchQuery
com.apelon.dts.server.SubConceptServer	DTS:005	com.apelon.dts.client.ClassQueryServer, com.apelon.dts.client.concept.OntylogClassQuery
com.apelon.dts.server.NavQueryServer	DTS:006	com.apelon.dts.client.NavServer, com.apelon.dts.client.concept.NavQuery
com.apelon.dts.server.SearchQueryServer	DTS:007	com.apelon.dts.client.SearchServer com.apelon.dts.client.concept.OntylogSearchQuery
com.apelon.dts.server.TranslateQueryServer	DTS:008	com.apelon.dts.client.TranslateServer
com.apelon.dts.server.ThesaurusConceptServer	DTS:009	com.apelon.dts.client.concept.ThesaurusConceptQuery
com.apelon.dts.server.AssociationServer	DTS:010	com.apelon.dts.client.association.AssociationQuery
com.apelon.dts.server.NamespaceServer	DTS:011	com.apelon.dts.client.namespace.NamespaceQuery
com.apelon.dts.server.TermServer	DTS:012	com.apelon.dts.client.term.TermQuery

Query Server	Header	Query Client
com.apelon.dts.server.DTSConceptServer	DTS:014	com.apelon.dts.client.concept.DTSConceptQuery
com.apelon.dts.server.OntylogConceptServer	DTS:015	com.apelon.dts.client.ConceptServer com.apelon.dts.client.concept.OntylogConceptQuery
com.apelon.dts.server.ClassifyServer	DTS:016	com.apelon.dts.client.classifier.ClassifyQuery
com.apelon.dts.server.OntylogExtConceptServer	DTS:017	com.apelon.dts.client.concept.OntylogExtConceptQuery
com.apelon.dts.server.ClassifyDetailsServer	DTS:018	com.apelon.dts.client.classifier.ClassifyDetailsQuery
com.apelon.dts.server.DTSCommonServer	DTS:019	com.apelon.dts.client.common.DTSCommonQuery
com.apelon.dts.server.SubsetServer	DTS:020	com.apelon.dts.client.subset.SubsetQuery

Configuration details for each of these servers are included later in the guide.

ApelonServer

Edit the highlighted property values for the **Apelon DTS Server**.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE apelonserverconfig SYSTEM "http://apelon.com/dtd/properties/apelonserverconfig.dtd">
<!-- ApelProps-->
<apelonserverconfig>
  <property name="port" value="6666"/>
  <property name="log" value="server/apelonserverlog.xml"/>
  <property name="validate_parser" value="false"/>
  <property name="authentication" value="false"/>
  <property name="user_admin" value="com.apelon.dts.server.DTSUsers"/>
  <connection default="true">
   <property name="type" value="oracle"/>
   <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver"/>
   <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]"/>
   <property name="user" value="dts"/>
   <property name="pass" value="dts"/>
   <property name="host" value="localhost"/>
   <property name="database_name" value="ORCL"/>
   <property name="database_port" value="1521"/>
  </connection>
  <!-- Example SQL Server connection (I-Net Sprinta driver) -->
  <!--
  <connection>
   <property name="type" value="sql2k" />
   <property name="jdbcDriver" value="com.inet.tds.TdsDriver" />
   <property name="url_template" value="jdbc:inetdae:[HOST]:[PORT]?database=[DATABASE]" />
   <property name="user" value="sa" />
   <property name="pass" value="" />
   <property name="host" value="localhost" />
   <property name="database_name" value="dts" />
   <property name="database_port" value="1433" />
  </connection>
  -->
  <!-- Example SQL Server connection (Microsoft driver) -->
  <!--
  <connection>
            <property name="type" value="sql2k" />
            <property name="jdbcDriver" value="com.microsoft.sqlserver.jdbc.SQLServerDriver" />
            <property name="url_template"
 value="jdbc:sqlserver://[HOST]:[PORT];databasename=[DATABASE];forwardReadOnlyMethod=serverCursor" />
            <property name="user" value="sa" />
            <property name="pass" value="" />
            <property name="host" value="localhost" />
            <property name="database_name" value="dts" />
            <property name="database_port" value="1433" />
  </connection>
```

port - The TCP port number to which the server listens.

log - The output file to which the log files are written in the *DTSInstall\bin\logs* directory.

validate_parser - Validates the XML format of queries and responses. In a production environment, set the value to **false** to turn off validation (and improve server performance).

authentication - Allows users to log in (with password) to a secure socket connection with the Apelon DTS Server. Set to **true** to require login; set to **false** (the default) if the user is not required to log into the server. Both client and server should set the property to true to enable a secure connection.

user_admin - Specifies the class that authenticates DTS users of a secure socket connection.

connection default - Allows you to set default connection values for all of the query servers in the **apelonserverprops.xml** file. Set to **true** (the default) to automatically configure all of the query server connections based on the default values specified here. These values become effective when the server is started. Note that you can enter alternate configuration values for any of the query servers; these alternate values will override the default values. Set to **false** if you want to configure each query server in the file individually.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

<u>**Optional Properties**</u>

You can add the following optional properties to the values for the **Apelon DTS Server** in the **apelonserverprops.xml** file, and override the default values as needed.

thread_pool_size - Number of threads the Apelon server can hold (default is **1000**).

thread_pool_monitor_sleep - Period of time for which the monitor thread sleeps (default is **1000** milliseconds).

thread_pool_min_free - Initial number of threads in the Apelon Server thread pool, and the number of threads to be added in the next expansion (default is **30**).

thread_pool_max_free - Maximum number threshold of threads in the Apelon Server thread pool; if idle thread is greater than this number, the monitor thread frees the idle threads (default is **60**).

max_client_workers - Maximum number of socket worker threads (default is **1000**).

max_accepts - Number of thread which calls Java accept() to accept client connections (default is **50**).

Socket Server Configurations

These discussions highlight configurations for Apelon Server socket connections.

Oracle Database Connection

Note the highlighted section from the **apelonserverprops.xml** file (*DTSInstall\bin\ server*).   The values shown are for an **Oracle** database connection (the default socket connection).

```
<apelonserverconfig>
  <property name="port" value="6666"/>
  <property name="log" value="server/apelonserverlog.xml"/>
  <property name="validate_parser" value="false"/>
  <property name="authentication" value="false"/>
  <property name="user_admin" value="com.apelon.dts.server.DTSUsers"/>
  <connection default="true">
  <property name="type" value="oracle"/>
  <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver"/>
  <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]"/>
  <property name="user" value="dts"/>
  <property name="pass" value="dts"/>
  <property name="host" value="localhost"/>
  <property name="database_name" value="ORCL"/>
  <property name="database_port" value="1521"/>
  </connection>
```

Microsoft SQL Server Connections

Note the highlighted sections from the **apelonserverprops.xml** file (*DTSInstall\bin\ server*). To establish a Microsoft SQL connection, modify the default property values as shown for either an **I-Net Sprinta** driver or a **Microsoft** driver.

```
<!-- Example SQL Server connection (I-Net Sprinta driver) -->
<!--
<connection>
  <property name="type" value="sql2k" />
  <property name="jdbcDriver" value="com.inet.tds.TdsDriver" />
  <property name="url_template" value="jdbc:inetdae:[HOST]:[PORT]?database=[DATABASE]" />
  <property name="user" value="sa" />
  <property name="pass" value="" />
  <property name="host" value="localhost" />
  <property name="database_name" value="dts" />
  <property name="database_port" value="1433" />
</connection>
-->
<!-- Example SQL Server connection (Microsoft driver) -->
<!--
<connection>
          <property name="type" value="sql2k" />
          <property name="jdbcDriver" value="com.microsoft.sqlserver.jdbc.SQLServerDriver" />
          <property name="url_template"
value="jdbc:sqlserver://[HOST]:[PORT];databasename=[DATABASE];forwardReadOnlyMethod=serverCursor" />
          <property name="user" value="sa" />
          <property name="pass" value="" />
          <property name="host" value="localhost" />
          <property name="database_name" value="dts" />
          <property name="database_port" value="1433" />
</connection>
```

To connect to an SQL Server database with a **named instance** (using either an **I-Net Sprinta** driver or a **Microsoft** driver) change the default **host** property value to reflect the database instance (e.g., <property name="**host**" value="**localhost/INSTANCE1**"/>).

Copy the **Sprinta.jar** file to the installed *DTSInstall\lib* directory.

Secure Socket Server Configurations

These discussions highlight configurations for Apelon Server secure socket connections.

Oracle Database Connection

Note the highlighted sections of the **apelonserverprops.xml** file (*DTSInstall\bin\server*). Change the value for the authentication property from **false** to **true**. The property values shown are for an **Oracle** database connection (the default connection).

```
<apelonserverconfig>
  <property name="port" value="6666"/>
  <property name="log" value="server/apelonserverlog.xml"/>
  <property name="validate_parser" value="false"/>
  <property name="authentication" value="true"/>
  <property name="user_admin" value="com.apelon.dts.server.DTSUsers"/>
  <connection default="true">
   <property name="type" value="oracle"/>
   <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver"/>
   <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]"/>
   <property name="user" value="dts"/>
   <property name="pass" value="dts"/>
   <property name="host" value="localhost"/>
   <property name="database_name" value="ORCL"/>
   <property name="database_port" value="1521"/>
  </connection>
```

Microsoft SQL Server Connections

Note the highlighted portions of the **apelonserverprops.xml** file (*DTSInstall\bin\server*). Change the value for the authentication property from **false** to **true**.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE apelonserverconfig SYSTEM "http://apelon.com/dtd/properties/apelonserverconfig.dtd">
<!-- ApelProps-->
<apelonserverconfig>
  <property name="port" value="6666"/>
  <property name="log" value="server/apelonserverlog.xml"/>
  <property name="validate_parser" value="false"/>
  <property name="authentication" value="true"/>
  <property name="user_admin" value="com.apelon.dts.server.DTSUsers"/>
  <connection default="true">
   <property name="type" value="oracle"/>
   <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver"/>
   <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]"/>
   <property name="user" value="dts"/>
   <property name="pass" value="dts"/>
   <property name="host" value="localhost"/>
   <property name="database_name" value="ORCL"/>
   <property name="database_port" value="1521"/>
  </connection>
```

To establish a Microsoft SQL connection, modify the default property values as shown for either an **I-Net Sprinta** driver or a **Microsoft** driver.

```
<!-- Example SQL Server connection (I-Net Sprinta driver) -->
<!--
<connection>
  <property name="type" value="sql2k" />
  <property name="jdbcDriver" value="com.inet.tds.TdsDriver" />
  <property name="url_template" value="jdbc:inetdae:[HOST]:[PORT]?database=[DATABASE]" />
  <property name="user" value="sa" />
  <property name="pass" value="" />
  <property name="host" value="localhost" />
  <property name="database_name" value="dts" />
  <property name="database_port" value="1433" />
</connection>
-->
<!-- Example SQL Server connection (Microsoft driver) -->
<!--
<connection>
          <property name="type" value="sql2k" />
          <property name="jdbcDriver" value="com.microsoft.sqlserver.jdbc.SQLServerDriver" />
          <property name="url_template"
value="jdbc:sqlserver://[HOST]:[PORT];databasename=[DATABASE];forwardReadOnlyMethod=serverCursor" />
          <property name="user" value="sa" />
          <property name="pass" value="" />
          <property name="host" value="localhost" />
          <property name="database_name" value="dts" />
          <property name="database_port" value="1433" />
</connection>
```

To connect to an SQL Server database with a **named instance** (using either an **I-Net Sprinta** driver or a **Microsoft** driver) change the default **host** property value to reflect the database instance (e.g., <property name="**host**" value="**localhost/INSTANCE1**"/>).

Copy the **Sprinta.jar** file to the installed *DTSInstall***\lib** directory.

QueryServer

Query Server provides backward compatibility with clients written using the DTS 2.4 API. The QueryServer works in association with the Apelon DTS Server, and services queries from the following classes:

com.apelon.dts.client.ConceptServer
com.apelon.dts.client.ClassQueryServer
com.apelon.dts.client.NavServer
com.apelon.dts.client.SearchServer
com.apelon.dts.client.Translate

Edit the highlighted property values for the **QueryServer**.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.QueryServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:001" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:001** points to the QueryServer.

> If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

<u>**Optional Property**</u>

You can add the following optional property for the QueryServer in the **apelonserverprops.xml** file, and override the default value as needed.

cs - Cache size in query server pool. Since the result of a client's query is stored into cache, if the data can be modified while the server is running, the cache size should be set to **0** (or the client may not get the latest data). The default value is **0**.

MatchServer

MatchServer provides backward compatibility with clients written using the DTS API. MatchServer services queries from classes ***com.apelon.dts.client.MatchServer*** and ***com.apelon.dts.client.match.MatchQuery***.

Class **MatchServer** provides for pattern matching and attribute searches; **MatchQuery** provides access to knowledgebase silos and pattern matching searches within the silos. Edit the highlighted property values for the **MatchServer**.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.MatchServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:004" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:004** points to the MatchServer.

> If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

SubConceptServer

SubConceptServer services queries from classes *com.apelon.dts.client.ClassQueryServer* and *com.apelon.dts.client.concept.OntylogClassQuery*. **ClassQueryServer** supports queries on subconcept relationships; **OntylogClassQuery** supports queries on subconcept relationships within an Ontylog concept hierarchy. Edit the highlighted property values for the **SubConceptServer**.

```
  <queryserver>
  <property name="qs" value="com.apelon.dts.server.SubConceptServer" />
  <property name="maxDbConns" value="15" />
  <property name="header" value="DTS:005" />
          <connection>
          <property name="type" value="oracle" />
          <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
          <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
          <property name="user" value="dts" />
          <property name="pass" value="dts" />
          <property name="host" value="localhost" />
          <property name="database_name" value="ORCL" />
          <property name="database_port" value="1521" />
          </connection>
  </queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:005** points to the SubConceptServer.

> If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

NavQueryServer

NavQueryServer services queries from classes *com.apelon.dts.client.NavServer* and *com.apelon.dts.client.concept.NavQuery*. Class **NavServer** provides for navigation over the concept hierarchy; **NavQuery** provides for navigation over the Ontylog concept hierarchy specifically. Edit the highlighted property values for the **NavQueryServer**.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.NavQueryServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:006" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:006** points to the NavQueryServer.

> If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

SearchQueryServer

SearchQueryServer services queries from classes *com.apelon.dts.client. SearchServer* and *com.apelon.dts.client.concept.SearchQuery*. Class **SearchServer** provides for pattern matching searches, index-based searches, and normalized searches; **OntylogSearchQuery** provides search methods for searching Ontylog and DTS concepts. Edit the highlighted property values for the **SearchQueryServer**.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.SearchQueryServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:007" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:007** points to the SearchQueryServer.

> If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

TranslateQueryServer

TranslateQueryServer services queries from class *com.apelon.dts.client. TranslateServer*. Class **TranslateServer** provides for translation of concept attributes between coding schemes. For example, if a concept has a property called **CPT Code** in one coding scheme, and another called **ICD9 Code** in the other coding scheme, it is possible to translate from one code value to the other. Edit the highlighted property values for the **TranslateQueryServer**.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.TranslateQueryServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:008" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:008** points to the TranslateQueryServer.

> If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

ThesaurusConceptServer

ThesaurusConceptServer services queries from class *com.apelon.dts.client.concept.* *ThesaurusConceptQuery*. Class **ThesaurusConceptQuery** provides for access to thesaurus concepts. Edit the highlighted property values for the **ThesaurusConceptServer**.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.ThesaurusConceptServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:009" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:009** points to the ThesaurusConceptServer.

If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

AssociationServer

AssociationServer services queries from class *com.apelon.dts.client. association. AssociationQuery*. Class **AssociationQuery** provides common methods for maintaining associations and association types. Edit the highlighted property values for the **AssociationServer**.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.AssociationServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:010" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:010** points to the AssociationServer.

If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

NamespaceServer

NamespaceServer services queries from class *com.apelon.dts.client.namespace. NamespaceQuery*. Class **NamespaceQuery** provides common methods for using namespaces. Edit the highlighted property values for the **NamespaceServer**.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.NamespaceServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:011" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:011** points to the NamespaceServer.

> If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

TermServer

TermServer services queries from class *com.apelon.dts.client. term.TermQuery*. Class **TermQuery** supports queries on terms (including their properties). Edit the highlighted property values for the **TermServer**.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.TermServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:012" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:012** points to the TermServer.

If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

DTSConceptServer

DTSConceptServer services queries from class *com.apelon.dts.client.concept. DTSConceptQuery*. Class **DTSConceptQuery** provides access to knowledgebase concepts. Edit the highlighted property values for the **DTSConceptServer**.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.DTSConceptServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:014" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:014** points to the DTSConceptServer.

If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

---

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

OntylogConceptServer

OntylogConceptServer services queries from classes *com.apelon.dts.client. ConceptServer* and *com.apelon.dts.client.concept.OntylogConceptQuery*. Class **ConceptServer** provides common methods for servers. The **ConceptServer** is a base class for the Classes **QueryServer**, **NavServer**, **SearchServer**, and **TranslateServer**. Edit the highlighted property values for the **OntylogConceptServer**.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.OntylogConceptServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:015" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:015** points to the OntylogConceptServer.

> If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

ClassifyServer

ClassifyServer services queries from class *com.apelon.dts.client.classifier.ClassifyQuery*. Class **ClassifyServer** services requests for classification for an Ontylog Extension namespace. Once a classification request is received, **DTS ClassifyServer** attempts to find the Modular Classifier URL for the Ontylog Extension namespace, prepares the request data, and sends the request data to the URL.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.ClassifyServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:016" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:016** points to the ClassifyServer.

> If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

OntExtensionConceptServer

OntExtensionConceptServer services queries from class *com.apelon.dts.client.concept. OntylogExtConceptQuery*. Class **OntylogExtConceptServer** services requests for addition, update, or deletion of attributes for defined concepts in an Ontylog Extension namespace.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.OntylogExtConceptServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:017" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:017** points to the OntylogExtConceptServer.

If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

ClassifyDetailsServer

ClassifyDetailsServer services queries from class *com.apelon.dts.client.classifier. ClassifyDetailsQuery*. Class **ClassifyDetailsQuery** services requests for detailed information regarding classification of an Ontylog Extension namespace.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.ClassifyDetailsServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:018" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:018** points to the ClassifyDetailsServer.

If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

DTSCommonServer

DTSCommonServer services queries from class *com.apelon.dts.client.common. DTSCommonQuery*. Class **DTSCommonQuery** services requests for common DTS data, and also provides a method to retrieve the current DTS schema version.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.DTSCommonServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:019" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:019** points to the DTSCommonServer.

> If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

SubsetServer

SubsetServer services all subset related queries and updates, such as adding, updating, deleting, retrieving, building, and exporting subsets. SubsetServer also handles queries such as retrieving concepts and namespaces from a specified subset.

```
<queryserver>
<property name="qs" value="com.apelon.dts.server.SubsetServer" />
<property name="maxDbConns" value="15" />
<property name="header" value="DTS:020" />
        <connection>
        <property name="type" value="oracle" />
        <property name="jdbcDriver" value="oracle.jdbc.driver.OracleDriver" />
        <property name="url_template" value="jdbc:oracle:thin:@[HOST]:[PORT]:[DATABASE]" />
        <property name="user" value="dts" />
        <property name="pass" value="dts" />
        <property name="host" value="localhost" />
        <property name="database_name" value="ORCL" />
        <property name="database_port" value="1521" />
        </connection>
</queryserver>
```

maxDbConns - Maximum number of database connections that this application can have at one time.

header - **DTS:020** points to the SubsetServer.

If the **connection default** property was set to **true** under **apelonserverconfig**, the values for the following properties will default to those under **apelonserverconfig**.

user - User name for the database account.

password - Password for accessing the database.

host - IP address of the machine on which the database resides.

database_name - Name of the Microsoft SQL Server database or Oracle instance.

database_ port - Database connection port.

Log Configuration

Log files record the Apelon DTS Server activity, and provide information to assist you in troubleshooting. The error log file lists the errors the server has encountered since it was started.

Apelon DTS utilizes Apache's log4j package to perform logging. One of the advantages of log4j is its manageability. Once the log statements have been inserted into the code, they can be controlled with configuration files. They can be selectively enabled or disabled, and sent to different and multiple output targets in user-chosen formats.

Log4j provides a low-maintenance way, at runtime, of altering the amount of logging and debugging information generated. What this means is that logging output is completely controllable while the application is still running. Log4j is fully configurable at runtime using external configuration files. In addition, the log files can automatically rollover when they reach a user specified size. For more information about log4j, visit the log4j Web site at http://jakarta.apache.org/.

The following is an example of an error log output file.

```
2001-05-22 11:03:11,103 INFO  [main] samples.dts.TestSearch - () - PROPERTY BY NAME: UMLS_CODE
2001-05-22 11:03:11,113 INFO  [main] samples.dts.TestSearch - () - PROPERTY BY ID: UMLS_CODE
2001-05-22 11:03:11,113 INFO  [main] samples.dts.TestSearch - () - SearchConcept with name matching: PROCEDURE*\\
2001-05-22 11:03:11,153 INFO  [main] samples.dts.TestSearch - () - SearchConcept with SNOMED_CODE property  matching P1-21*
2001-05-22 11:03:11,173 ERROR [main] samples.dts.TestSearch - () - Error in searchConceptProperty:
java.lang.NullPointerException
```

apelonserverlog.xml file

The configuration information for logging is contained in the **apelonserverlog.xml** file (*DTSInstall\bin\server*). In this file you can customize logging for any resource by specifying the desired values for the associated parameters. Three main components in log4j, *categories*, *appenders,* and *layouts*, work together to enable developers to log messages. Developers can log messages according to message type and priority, control (at runtime) how messages are formatted, and where they are reported.

Open the **apelonserverlog.xml** file, which is illustrated for your reference.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">

<log4j:configuration>
  <appender name="CONSOLE" class="org.apache.log4j.ConsoleAppender">
   <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="%m%n"/>
   </layout>
  </appender>

  <appender name="LOGTOFILE" class="org.apache.log4j.RollingFileAppender">
   <param name="File" value="../logs/ApelonServer.log" />
   <param name="MaxFileSize" value="10000000" />
   <param name="MaxBackupIndex" value="25" />
   <param name="Append" value="true" />
   <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="%d{yyyy-MM-dd HH:mm:ss.SSS} %-5p %c [%t] %C.%M() - %m%n"/>
   </layout>
  </appender>

  <!-- In order to use this NTEventLogAppender you need the NTEventLogAppender.dll in your system path -->
  <appender name="NTEVENTLOG" class="org.apache.log4j.nt.NTEventLogAppender">
   <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="%d{yyyy-MM-dd HH:mm:ss.SSS} %-5p %c [%t] %C.%M() - %m%n"/>
   </layout>
  </appender>

  <category name="apelon.data" additivity="false">
   <priority value="info" />
   <appender-ref ref="LOGTOFILE" />
   <appender-ref ref="CONSOLE" />
  </category>
  <category name="apelon.data.db" additivity="false">
   <priority value="info" />
   <appender-ref ref="LOGTOFILE" />
   <appender-ref ref="CONSOLE" />
  </category>
  <category name="apelon.data.xml" additivity="false">
   <priority value="info" />
   <appender-ref ref="LOGTOFILE" />
   <appender-ref ref="CONSOLE" />
  </category>
  <category name="apelon.data.io" additivity="false">
   <priority value="info" />
   <appender-ref ref="LOGTOFILE" />
   <appender-ref ref="CONSOLE" />
  </category>
  <category name="apelon.data.client" additivity="false">
   <priority value="info" />
   <appender-ref ref="LOGTOFILE" />
   <appender-ref ref="CONSOLE" />
  </category>
  <category name="apelon.data.server" additivity="false">
   <priority value="info" />
   <appender-ref ref="LOGTOFILE" />
   <appender-ref ref="CONSOLE" />
  </category>
  <category name="apelon.system.logging" additivity="false">
   <priority value="info" />
   <appender-ref ref="LOGTOFILE" />
   <appender-ref ref="CONSOLE" />
  </category>
  <root>
   <priority value="info" />
   <appender-ref ref="LOGTOFILE" />
   <appender-ref ref="CONSOLE" />
  </root>
</log4j:configuration>
```

Configure Appenders

Log4j allows logging requests to print to multiple destinations. An output file destination is called an **Appender**. Appenders can exist for the console, files, GUI components, remote socket servers, NT Event Loggers, and remote UNIX Syslog daemons. This is the physical location of the log file. If not defined, output log files default to console.

You can configure three appenders in the **apelonserverlog.xml** file.

- A console appender, **CONSOLE**

- One RollingFile appender (a flat file that rolls over when it reaches a default size) **LOGTOFILE**

- The Windows NT event log, **NTEVENTLOG**

Note the highlighted **<appender name** properties in the portion of the file that is illustrated.

```
<appender name="CONSOLE" class="org.apache.log4j.ConsoleAppender">
  <layout class="org.apache.log4j.PatternLayout">
   <param name="ConversionPattern" value="%m%n"/>
  </layout>
</appender>

<appender name="LOGTOFILE" class="org.apache.log4j.RollingFileAppender">
  <param name="File" value="../logs/ApelonServer.log" />
  <param name="MaxFileSize" value="10000000" />
  <param name="MaxBackupIndex" value="25" />
  <param name="Append" value="true" />
  <layout class="org.apache.log4j.PatternLayout">
   <param name="ConversionPattern" value="%d{yyyy-MM-dd HH:mm:ss.SSS} %-5p %c [%t] %C.%M() - %m%n"/>
  </layout>
</appender>

<!-- In order to use this NTEventLogAppender you need the NTEventLogAppender.dll in your system path -->
<appender name="NTEVENTLOG" class="org.apache.log4j.nt.NTEventLogAppender">
  <layout class="org.apache.log4j.PatternLayout">
   <param name="ConversionPattern" value="%d{yyyy-MM-dd HH:mm:ss.SSS} %-5p %c [%t] %C.%M() - %m%n"/>
  </layout>
</appender>
```

Configure and Associate Layouts with Appenders

In addition to customizing the log message output destination, you can configure the output format of the messages. You do this by associating a layout with an appender. The layout is represented by a **conversion pattern**, which determines the format for the logged messages.

<param name="**ConversionPattern**" value="**%m%n**"/>

Note the following table of symbols that can be included in a conversion pattern.

Conversion Pattern Symbol	Format Definition
%d	Date of the log request
%r	Number of milliseconds elapsed since start of program
%-5p	Priority of the log request
[%t]	Thread making the log request
%C	Category name associated with the log request
(%x)	Nested diagnostic context (NDC)
%m	Message of the statement
(%F:%L)	Outputs the filename and line number generating the message
\n	Go to a new line at the end of the message

The default output format is based on Log4j's PatternLayout, and defaults to a conversion pattern of "**%m%n**" for the **CONSOLE** appender (message text, then go to a new line at the end of the message).

A log file with the conversion pattern "**%d %-5p [%t] %C – (%x) - %m\n** will produce message output in the following format:

```
2002-05-06 15:30:28,817 INFO [Thread-82]
com.apelon.apelonserver.admin.AdminServer$RequestHandler - () - SOCKET IN:editconfig
```

The first field is the date. The second field is the priority. The third field is the thread outputting the log statement. The fourth field is the rightmost two components of the category making the log request. The fifth field (just before the '-' and in parentheses) is the nested diagnostic context (NDC). Note the nested diagnostic context may be empty. The text after the '-' is the message of the statement. See *View the Log File*.

The configuration properties for the appender **LOGTOFILE** are defined:

```
<appender name="LOGTOFILE" class="org.apache.log4j.RollingFileAppender">
   <param name="File" value="../logs/ApelonServer.log" />
   <param name="MaxFileSize" value="10000000" />
   <param name="MaxBackupIndex" value="25" />
   <param name="Append" value="true" />
   <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="%d{yyyy-MM-dd HH:mm:ss.SSS} %-5p %c [%t] %C.%M() -
%m%n"/>
   </layout>
  </appender>
```

File - The file name to which the log is written. Output files are in *DTSInstall***bin\logs**.

MaxFileSize - The log file has a maximum file size of 10MB (default value="**10000000**").

MaxBackupIndex - A maximum of 25 backup files, named **AppLog.1**, **AppLog.2**, etc., will be created before the log rolls over to AppLog.1 again (default value="**25**").

Append - If **true** new messages will be appended to the existing file messages. If false the existing log file will be overwritten.

Categories

Categories define a hierarchy and give the programmer run-time control on which statements are printed or not. Categories are assigned Priorities. A log statement is printed depending on its priority/category combination. You can enable or disable logging requests based on the priority assigned in each category. See *Disable Logging Requests*. The priority and category names can be seen in the logged messages, so this will help you determine if lesser security messages can be omitted from the log without losing useful information.

Priorities

The **priority** code really is a severity code, ranging from **DEBUG** messages at the lowest level to **FATAL** at the top. The set of possible priorities includes **debug, info, warn, error**, and **fatal**.

- The **DEBUG** priority designates fine-grained informational events that are most useful to debug an application.
- The **INFO** priority designates informational messages that highlight the progress of the application at coarse-grained level.
- The **WARN** priority designates potentially harmful situations.
- The **ERROR** priority designates error events that might still allow the application to continue running.
- The **FATAL** priority designates very severe error events that will presumably lead the application to abort.

Each category defined in the **apelonserverlog.xml** file is assigned a priority, and references appenders that establish the output destinations. Note the category definition shown, with its priority value (info) and appender-refs (**LOGTOFILE** and **CONSOLE**).

```
</category>
  <category name="apelon.data.server" additivity="false">
   <priority value="info" />
   <appender-ref ref="LOGTOFILE" />
   <appender-ref ref="CONSOLE" />
  </category>
```

Assigning a priority to a category means that messages of that priority, and higher, will be shown.

Disable Logging Requests

Logging requests are made by invoking one of the printing methods for a category. These printing methods are the priorities assigned in log4j: Debug, Info, Warn, Error, and Fatal. The printing method is based on the priority of a logging request. A logging request is enabled if its priority is higher than, or equal to, the priority of its category; otherwise, the request is disabled. This rule assumes that priorities are ordered as follows: **Debug** < **Info** < **Warn** < **Error** < **Fatal**.

For example, if the category **apelon.data.db** is assigned a priority of **warn**, the statement **apelon.data.db.warn** is a logging request of priority **warn**. As such, error messages with a priority of **warn**, **error** or **fatal** will print in the log file. To disable logging for the category, assign the category the highest priority, **fatal**, so only error messages defined as **fatal** print to the log file.

Rolling Log Files

Rolling files enable you to remove the old logs without affecting the running service. A rolling file collects log data until it reaches the size set in the rollover file size setting.

The first time that size is reached, the current file is renamed to *Filename.1* and a new file is created. The default value in the configuration parameters allows 25 rolling log files to be created in this manner. When that maximum number is reached, the process begins over again by renaming the current file to *Filename.1*.

```
<appender name="LOGTOFILE" class="org.apache.log4j.RollingFileAppender">
  <param name="File" value="../logs/ApelonServer.log" />
  <param name="MaxFileSize" value="10000000" />
  <param name="MaxBackupIndex" value="25" />
  <param name="Append" value="true" />
  <layout class="org.apache.log4j.PatternLayout">
   <param name="ConversionPattern" value="%d{yyyy-MM-dd HH:mm:ss.SSS} %-5p %c [%t] %C.%M() - %m%n"/>
  </layout>
</appender>
```

In the configuration parameters shown, the values for the rolling log file are as follows: the maximum size limit on a rollover log file is 10MB (**10000000**) and **25** backup files can be created before the log rolls over to *Filename 1* again.

View the Log File

The **apelonserver.log** file is created in the **bin\logs** subdirectory, and lists errors the server has encountered since the server was started. It also contains informational messages about the server, such as date and time that the server was started, thread, and category name. The log output can be customized in many ways for layout and destination.

To view the output, right-click the appropriate file in the **bin\logs** directory, then select a text editor with which to open the log file. WordPad is preferred over Notepad, as Notepad includes no formatting (the layout will not display accurately).

Below is an example of the output that would be recorded in the **apelonserver.log** destination file using the conversion pattern "%d %-5p [%t] %c{2} %x - %m/n" (see *Configure and Associate Layouts With Appenders*). In this instance, the priority is defined as Info.

```
2001-05-22 11:43:26,451 INFO [Thread-29] com.apelon.common.server.RemoteServerSocket$RSSClient - (T-990546066831
WEEK.ct.apelon.com/192.168.1.130) - SOCKET IN:<?xml version="1.0" ?><!DOCTYPE fetchType SYSTEM "query.dtd"><fetchType>
<roleType value="all"/></fetchType>
```

%d	=	2001-05-22 11:43:26	=	date and time
%-5p	=	INFO	=	priority
[%t]	=	[Thread-29]	=	thread making request
%c	=	com.Apelon.common.server. RemoteServerSocket$RSSClient	=	category
{2}	=	(T-990546066831 WEEK 192.168.1.130)	=	nested diagnostic context
%m	=	SOCKET IN:<?xml version fetchType>	=	message

Below is an example of the output that would be recorded in the **AdminServerLog** file with the same layout configuration as above.

```
2001-05-22 11:43:06,993 INFO [Thread-31] com.apelon.common.admin.AdminServer - () - SOCKET IN:getcurrthreadcount
2001-05-22 11:43:07,003 INFO [Thread-31] com.apelon.common.admin.AdminServer - () - SOCKET OUT:30
```

Secure Server Mode

The Apelon DTS Server can be operated in a **Secure Server Mode**. When operating in Secure Server Mode, the DTS Server will authenticate the clients with which it communicates, providing more controlled access to the server.

In order to use the Secure Server Mode, you must configure the **apelonserverprops.xml** file to activate the feature, and users must exist on the system. In Secure Server Mode only users who have an authorized login for the server can access server features.

Activate Secure Server Mode

When installed, the Apelon DTS Server defaults to the non-secure operating mode. Follow this procedure to activate the Secure Server Mode.

1. Go to *DTSInstall***\bin\server** and open the file **apelonserverprops.xml** in Notepad.

2. Locate the line: **<property name="authentication" value="false" />**. If this line is not present, add it below the last property name line.

3. Change **false** to **true**.

4. Save the apelonserverprops.xml file.

5. Restart the Apelon DTS Server. The server will now be running in Secure Server Mode, and it will always come up in Secure Server Mode when it is restarted.

Manage DTS Users

If the Apelon DTS Server is to be operated in Secure Server Mode, all users must log in to use the server. This requires a System Administrator to create valid users, and assign each user a login before the user can connect to the server.

Apelon DTS provides the tools to create users, and to assign and maintain each user's edit permissions within local namespaces. A GUI-based **User Manager** tool is available from the DTS **Start** menu to perform user maintenance.

For a *Windows* installation, the System Administrator also can run the file **UserManager.bat** in the *DTSInstall***bin\admin** subdirectory to maintain users. For a Linux installation, the System Administrator can execute **UserManager.sh** in **bin/admin** to maintain users.

Each method for maintaining users is discussed.

Manage DTS Users With User Manager GUI Application

Create User/Assign Namespace Permissions

Follow this procedure to create a new DTS user, and also to assign the user edit permissions in local namespaces.

1. Select **User Manager** from the DTS **Start** menu (**Start>Programs>Apelon> *DTSInstall*>User Manager**). The *Apelon DTS User Manager* window displays.
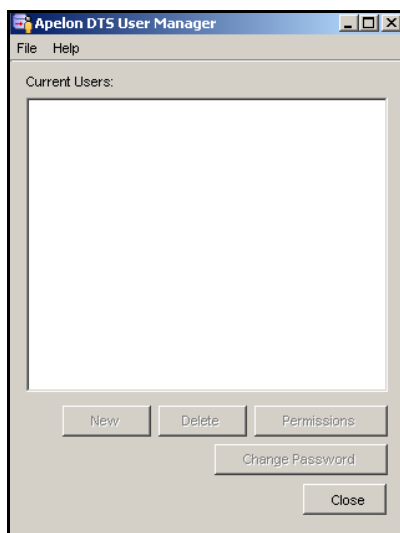
2. At this point you must connect to the DTS database. Select **Connect** from the *Apelon DTS User Manager* window **File** menu. The *Connect to Database* window displays.
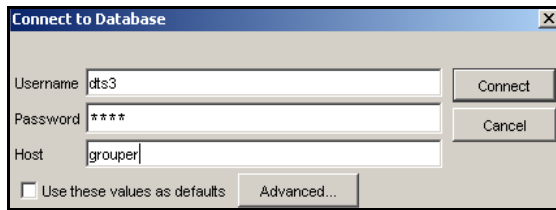
3. Enter your *Username*, *Password*, and *Host* name (for the machine where your database is installed). To recall these values for future sessions, click *Use these values as the defaults*.

4. If your configuration requires you to modify the default Oracle instance and port designations, or to choose SQL Server, click on **Advanced**. The *Advanced Setup* window displays.
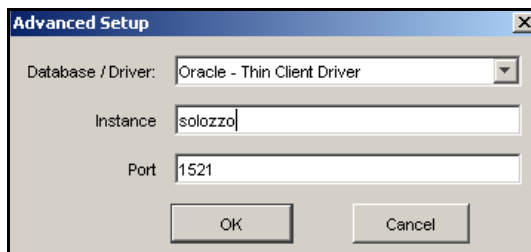

If you choose **Oracle – Thin Client Driver** in the *Database/Driver* field, you can accept the default database name in the *Database name/instance* field, and the default port number in the *Port* field, or override the defaults to connect to the desired database. Click **OK**.

  - If you indicated during DTS installation that you maintain your DTS knowledgebase in a *Microsoft* SQL Server database, and you have obtained and installed the **iSprinta Enterprise** driver, then **SQL Server – Sprinta Driver** is included as an option in the *Database/Driver* field.

  - If you indicated during DTS installation that you maintain your DTS knowledgebase in a *Microsoft* SQL Server database, and selected the Microsoft driver that is provided with DTS, then **SQL Server – Microsoft** is included as an option in the *Database/Driver* field.

In the displayed *Database/Driver* field, specify your SQL Server/driver selection. In the displayed *Instance* field, specify the SQL Server named instance (if you are using a default instance, leave the field blank).

5. Click **Connect** on the *Connect to Database* window. The *Apelon DTS User Manager* window displays again. Current DTS users are listed in the **Current Users** area.
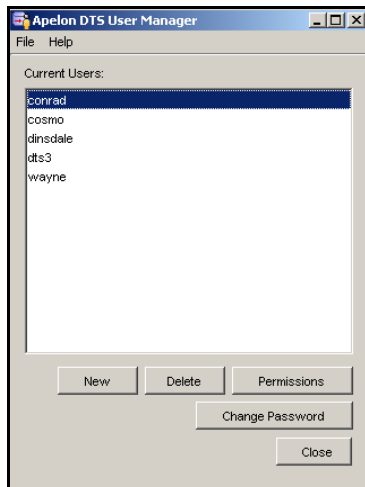
6. Click **New**. The *Add New User* window displays.
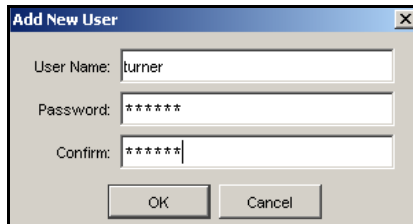
7. Specify the *User Name* and *Password* for the new user (the password displays as asterisks). Retype the new password in the *Confirm* field (the password displays again as asterisks).

8. Click **OK**. The *Apelon DTS User Manager* window displays again, listing the new user.
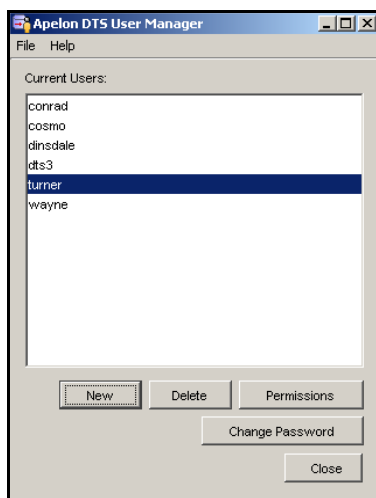
9. At this point you can assign local namespace edit permissions for the new user, and/or edit permissions for an existing user. Highlight a username listed under **Current Users** on the *Apelon DTS User Manager* window.
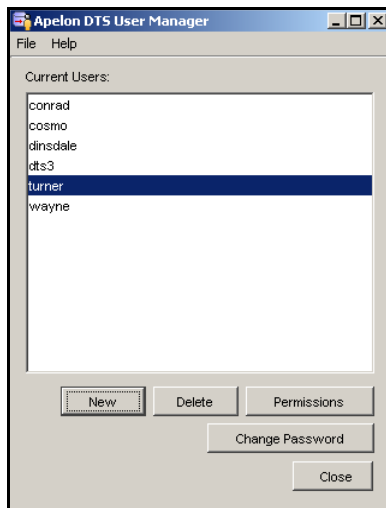
10. Click **Permissions**. The *Select Permissions* window displays. The username you selected is listed, as well as the local namespaces in your DTS Knowledgebase for which edit permissions can be assigned..
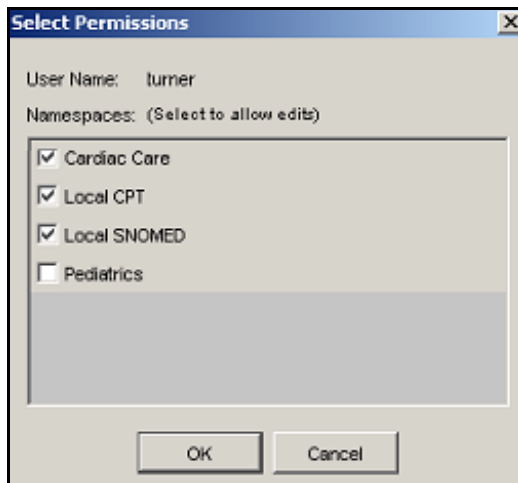
On this window you can select the **local namespace(s)** in the DTS Knowledgebase for which the user will have edit capability.

11. For the new or existing user, click each checkbox representing the local namespace for which the user will have edit permission. To remove namespace edit permission for a user, click the checkbox adjacent to the listed namespace to remove the check mark.

12. Click **OK** to create or update the user's edit permissions in the selected local namespaces. Click **Cancel** to ignore the new or updated permissions.

13. Click **Close** to exit the *Apelon DTS User Manager* window.

Change User Password

1. To change an existing user's password, highlight the username on the *Apelon DTS User Manager* window.
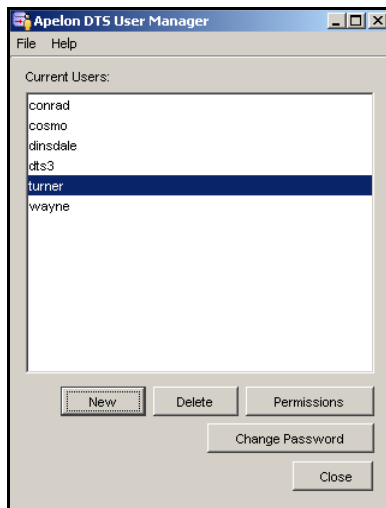
2. Click **Change Password**. The *Change Password* window displays.
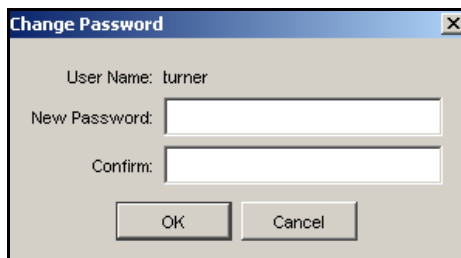
3. Enter the *New Password* for the user (the new password displays as asterisks). Retype the new password in the *Confirm* field (the new password displays again as asterisks).

4. Click **OK** to update the user password. Click **Cancel** to ignore the modification.

5. Click **Close** to exit the *Apelon DTS User Manager* window.

Delete DTS User

1. To delete an existing DTS user, highlight the username on the *Apelon DTS User Manager* window.
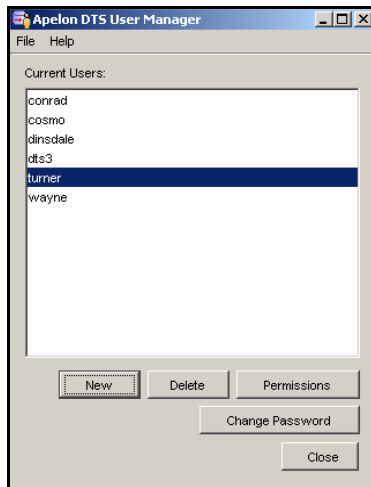
2. Click **Delete**. The following confirmation window displays.
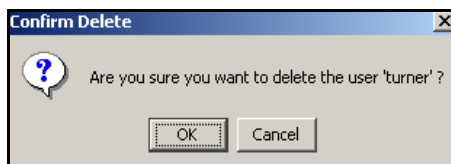
3. Click **Cancel** to ignore the deletion. Click **OK** to delete the username from the list of **Current Users** on the *Apelon DTS User Manager* window.

4. Click **Close** to exit the *Apelon DTS User Manager* window.

Manage DTS Users With UserManager.bat

Create/Maintain Users

In addition to the GUI-based **User Manager** tool, the System Administrator can run the file **UserManager.bat** in the *DTSInstall***\bin\admin** subdirectory to maintain users. This program allows the System Administrator to create the user table in the Apelon database, and then **Add**, **Edit**, **Delete**, or **View** the existing users; you must shut down, then restart the server after adding, editing, or deleting a user. **UserManager.bat** also allows you to assign and edit user write permissions within all available namespaces in your database.

> For a Linux installation, the System Administrator must execute **UserManager.sh** in **bin/admin** to start the User Manager and maintain users. After the System Administrator executes **UserManager.sh to** start the User Manager, the procedures for creating and maintaining DTS users are identical to those used when **UserManager.bat** is run with *Windows*.

- Before running RemoteAdmin on AIX with a secure socket server, **UserManager** on AIX should be run to generate a valid DTS Username/Password.

- Before running a DTS client on *Windows* with a secure socket server, **UserManager.bat** installed on *Windows* should be run to generate a valid DTS Username/Password.

View Writable Namespaces

After you create a new user you must assign write permission to that user for access to one or more namespaces within your knowledgebase. Follow this procedure to view all writeable namespaces; it is from this list that you will select namespaces in which the new user will have write permission..

1. Start **UserManager** and log into the database. When connected the **UserManager** menu displays.

2. Enter **8**, **View all writable namespaces**.

```
Do you want to -
[1] Add DTS user,
[2] Edit DTS user,
[3] Delete DTS user,
[4] Add permission,
[5] Edit permission,
[6] Delete permission,
[7] View all DTS users,
[8] View all writeable namespaces,
[9] View the writeable namespaces for the specified DTS user,
[10] Create DTS users table,
[11] Create permission table,
[12] Exit
>> 8_
```

A list displays of all namespaces for which write permissions can be assigned.

```
*** List of All Writeable Namespaces ***
(1) Name - SNOMED,          Id - 30
(2) Name - SNOMED_HIST_REL,      Id - 31
(3) Name - CPT2003,         Id - 20
(4) Name - ICD2003,         Id - 10
(5) Name - UMLSAssociation,      Id - 2000
(6) Name - ULT,  Id - 1057
(7) Name - CCS,  Id - 1005
(8) Name - OMS,  Id - 1044
(9) Name - DSM4,            Id - 1013
(10) Name - LNC,            Id - 1026
(11) Name - ICD10,          Id - 1018
(12) Name - MSH,            Id - 1033
(13) Name - SRC,            Id - 1056
(14) Name - NCI,            Id - 1039
(15) Name - AOD,            Id - 1002
(16) Name - ICPC2P,         Id - 1023
(17) Name - SPN,            Id - 1055
(18) Name - HL7,            Id - 1017
(19) Name - CST,            Id - 1010
(20) Name - VANDF,          Id - 1060
(21) Name - UWDA,           Id - 1059
(22) Name - NIC,            Id - 1042
(23) Name - NCBI,           Id - 1038
(24) Name - ICD10AM,        Id - 1019
(25) Name - NOC,            Id - 1043
(26) Name - NAN,            Id - 1037
(27) Name - CSP,            Id - 1009
(28) Name - ICPC2E,         Id - 1022
(29) Name - MTH,            Id - 1034
(30) Name - PDQ,            Id - 1046
(31) Name - HCPCS,          Id - 1015
(32) Name - ICPC,           Id - 1021
(33) Name - UMD,            Id - 1058
(34) Name - RXNORM,         Id - 1052
(35) Name - HHC,            Id - 1016
(36) Name - WHO,            Id - 1061
(37) Name - MDR,            Id - 1029
(38) Name - STYRT,          Id - 40
************************
```

Note the count (quantity) of namespaces for which you intend to grant write user permissions, as well as the IDs (numbers) of each of those namespaces.

Add Namespace Permissions to User

Follow this procedure to add namespace write permissions to users. You need to know the number of namespaces for which you are assigning write permissions, as well as the ID for each namespace (refer to the *View Writable Namespaces* procedure).

1. Start **UserManager** and log into the database. When connected the **UserManager** menu displays.

2. Enter **4**, **Add permission**.

```
Do you want to -
[1] Add DTS user,
[2] Edit DTS user,
[3] Delete DTS user,
[4] Add permission,
[5] Edit permission,
[6] Delete permission,
[7] View all DTS users,
[8] View all writeable namespaces,
[9] View the writeable namespaces for the specified DTS user,
[10] Create DTS users table,
[11] Create permission table,
[12] Exit
>> 4
```

3. When prompted, specify the quantity of namespaces for which you will be granting write permission for the user. Also specify the namespace ID for each namespace assigned. A success message displays indicating that the permissions have been assigned to the user.

```
Enter the user name
>> wayne
How many namespaces can this user update?
>> 1
Enter a permitted namespace id for the user 'wayne'
>> 30
SUCCESS: Permission for user 'wayne' updating the namespace '30' added to the da
tabase.
```

View Writable Namespaces for a User

In order to edit or delete the writable namespaces assigned to a user, you must be able to list the namespaces currently assigned. Follow this procedure to list namespaces for which write permissions are assigned currently.

1. Start **UserManager** and log into the database. When connected the **UserManager** menu displays.

2. Enter **9**, **View the writable namespaces for the specified DTS User**.

```
Do you want to -
[1] Add DTS user,
[2] Edit DTS user,
[3] Delete DTS user,
[4] Add permission,
[5] Edit permission,
[6] Delete permission,
[7] View all DTS users,
[8] View all writeable namespaces,
[9] View the writeable namespaces for the specified DTS user,
[10] Create DTS users table,
[11] Create permission table,
[12] Exit
>> 9
```

3. When prompted, specify the name of the user for whom you want writable namespaces listed. The name and ID of each assigned namespace is listed.

```
Enter the user name
>> wayne
*** List of Writeable Namespaces for 'wayne' ***
<1> Name - SNOMED_HIST_REL,      Id - 31
************************
```

Edit Namespace Write Permissions for a User

Follow this procedure to edit the namespace write permissions currently assigned to a user. You need to know the IDs of the old and new namespaces assigned to the user (refer to the *View Writable Namespaces for a User* procedure).

1. Start **UserManager** and log into the database. When connected the **UserManager** menu displays.

2. Enter **5**, **Edit permission**.

```
Do you want to -
[1] Add DTS user,
[2] Edit DTS user,
[3] Delete DTS user,
[4] Add permission,
[5] Edit permission,
[6] Delete permission,
[7] View all DTS users,
[8] View all writeable namespaces,
[9] View the writeable namespaces for the specified DTS user,
[10] Create DTS users table,
[11] Create permission table,
[12] Exit
>> 5
```

3. When prompted, indicate the user for whom you want to modify namespace write permissions. You then must specify the ID for the old namespace, and the new one that will replace it.

```
Which DTS user's permission do you want to change?
>> wayne
What is the old permission? (old namespace_id)
>> 30
What is the new permission? (new namespace_id)
>> 31
SUCCESS: Permission for 'wayne' on '31' updated in the database
```

A success message displays indicating that the user's namespace permission was changed.

Delete Namespace Write Permissions

Follow this procedure to delete a namespace write permission currently assigned to a user. You need to know the ID of each assigned namespace that will be deleted (refer to the *View Writable Namespaces For a User* procedure).

1. Start **UserManager** and log into the database. When connected the **UserManager** menu displays.

2. Enter **6**, **Delete permission**.

```
Do you want to -
[1] Add DTS user,
[2] Edit DTS user,
[3] Delete DTS user,
[4] Add permission,
[5] Edit permission,
[6] Delete permission,
[7] View all DTS users,
[8] View all writeable namespaces,
[9] View the writeable namespaces for the specified DTS user,
[10] Create DTS users table,
[11] Create permission table,
[12] Exit
>> 6
```

3. When prompted, indicate the name of the user for whom you are deleting write permission in a namespace, then indicate the ID for that namespace. When the confirmation message displays, enter **yes** to confirm the deletion.

```
Which DTS user's permission needs to be deleted?
username >> wayne
namespace_id >> 31
Are you sure, you want be delete permission for user 'wayne' on namespace '31'?
<yes/no>
>> y
SUCCESS: Permission for user 'wayne' on namespace '31' deleted from the database
```

4. Restart the server to make the deletion effective.

Edit User Password

Follow this procedure to change the user's password. Note that you also can refresh user information using the RemoteAdmin tool (refer to the *Reset User Information* discussion later in the guide).

1. Start **UserManager** and log into the database. When connected the **UserManager** menu displays.

2. Enter **2**, **Edit DTS user**.

```
Do you want to -
[1] Add DTS user,
[2] Edit DTS user,
[3] Delete DTS user,
[4] Add permission,
[5] Edit permission,
[6] Delete permission,
[7] View all DTS users,
[8] View all writeable namespaces,
[9] View the writeable namespaces for the specified DTS user,
[10] Create DTS users table,
[11] Create permission table,
[12] Exit
>> 2
```

3. When prompted, enter the name of the user whose password is to be changed.

```
Which DTS user's password do you want to change?
>> edgar
```

4. You are prompted for the new password for the user. Enter the new password, observing the 32-character limit. For security, only asterisks display. A success message displays indicating that the new user password was saved.

```
Which DTS user's password do you want to change?
>> edgar
What is the new password for 'edgar' ? (Maximum 32 characters)
Password: **********
SUCCESS: User 'edgar' updated in the database
```

5. Restart the server to make the password change effective (or, refresh the user information using the RemoteAdmin tool; see the *Reset User Information* discussion).

Delete Users

Follow this procedure to remove a user from the system.

1. Start **UserManager** and log into the database. When connected the **UserManager** menu displays.

2. Enter **3**, **Delete DTS user**.

```
Do you want to -
[1] Add DTS user,
[2] Edit DTS user,
[3] Delete DTS user,
[4] Add permission,
[5] Edit permission,
[6] Delete permission,
[7] View all DTS users,
[8] View all writeable namespaces,
[9] View the writeable namespaces for the specified DTS user,
[10] Create DTS users table,
[11] Create permission table,
[12] Exit
>> 3
```

3. When prompted, enter the name of the user to be deleted. A confirmation message displays; enter **yes** to confirm deletion. A success message displays indicating that the user was deleted.

```
Which DTS user needs to be deleted?
>> carmine
Are you sure, you want be delete DTS user 'carmine'? (yes/no)
>> y
SUCCESS: User 'carmine' deleted from the database
```

4. Restart the server to make the deletion effective (or, refresh the user information using the RemoteAdmin tool; see the *Reset User Information* discussion).

View Users

At any time the System Administrator can view a list of all users. Follow this procedure to list current users.

1. Start **UserManager** and log into the database. When connected the **UserManager** menu displays.

2. Enter **7**, **View all DTS users**.

```
Do you want to -
[1] Add DTS user,
[2] Edit DTS user,
[3] Delete DTS user,
[4] Add permission,
[5] Edit permission,
[6] Delete permission,
[7] View all DTS users,
[8] View all writeable namespaces,
[9] View the writeable namespaces for the specified DTS user,
[10] Create DTS users table,
[11] Create permission table,
[12] Exit
>> 4
```

3. When prompted, indicate if you want the list displayed on the screen (**1**) or written to a file.

```
The user list will be very long if the number of user is large.
Do you want to view the list -
[1] on the screen?
[2] from a file?
>> _
```

To write the users list to a file, enter **2**, then specify the name of the file that will be created in the **bin\admin** subdirectory (the default file name is **userList.txt**). You can open this file at your convenience and view DTS users.

```
Enter the file name (Hit Enter to use "userList.txt"), the file will be at Apelo
n DTS\bin\admin
>> _
```

If you selected to view users on the screen, the list of current DTS users displays.

```
*** List of DTS Users ***
<1> wayne
<2> edsel
<3> edgar
<4> Abercrombie
************************
```

Remote Administration Server

Included with DTS is a remote server administration tool. This tool allows an administrator to connect to the server, and check server resources and connections from a remote desktop. This allows the server to be shut down or otherwise controlled if it is running in daemon mode (as a system background task).

Using remote server administration, the administrator can conduct ping tests, display database connection count, shut down the server, reset user information, and perform other functions. Apelon provides two methods by which you can access the server remotely, **RemoteAdmin** for the AIX machine, and **RemoteAdmin.bat** for the *Windows* environment.

> To access the server remotely for a Linux installation, the administrator should run **admin/ RemoteAdmin.sh** in **bin/admin**.

Note: If the Remote Administration Server is run using a Secure Socket Server Connection, you will be prompted "**Is authentication turned on for the server? (yes/no)**" if you answer **yes**, you will need to enter your DTS **Username** and **Password**.

Command Line Parameters

You can list a variety of performance statistics about the Apelon DTS Server, including Java thread counts, connection counts, client connection size, and server performance. To determine the administrative options that can be tested remotely, execute **RemoteAdmin** from the AIX machine or execute the **RemoteAdmin.bat** file located in the **bin\admin** subdirectory. The console window lists and describes the various Command Line options for testing the Apelon DTS Server.

```
Connects to an ApelonServer at a given host on a port

-? this message -ping  returns a timestamp indicating the server is alive
-status returns stats indicating thread counts, active DB connections, etc..
-ss   use secure socket connection, requires next two options.
-user user authorized to talk to server
-pass user password
-shutdown [-noprompt] -- shut down the server  completely. By default shutdown w
ill prompt a user to confirm shutdown. To avoid prompt, -shutdown -noprompt shou
ld be used
-resetusers ask the server to reload users tables
-----
Hit enter to terminate.
```

NOTE: You cannot run any commands if you initiated Remote Administrator by double-clicking **RemoteAdmin.bat** through *Windows*. This only produces a list of the available options. You must re-enter Remote Administrator through the Command prompt for it to respond to commands.

Test the Server

To test the server using the Command Line parameters in the **RemoteAdmin.bat** file (or on AIX, the RemoteAdmin shell script) initiate the Command prompt, then enter the **RemoteAdmin** command, server host and port, and the specific parameter you wish to test. The location of the Command prompt varies with different versions of *Windows* (e.g., in one of the submenus of the **START** menu).

Server Status

Using **RemoteAdmin** you can display server status information, including current active database connections and **thread counts**. Threads enhance performance and functionality by allowing a program to efficiently perform multiple tasks simultaneously; thread counts indicate how the server is responding. Listed information includes the following:

- **Active Thread Count** - indicates the number of threads currently processing a client request.
- **Active Database Connection Count** - tells you the number of database connections being used within an established pool of allowable database connections.
- **Current Number of Threads** - indicates the number of threads allowed to process client requests.
- **Client Connection Size** is the number of reusable socket connection objects available. When the Apelon DTS Server receives a request from a client through a socket connection, once the request is served, the Apelon DTS Server saves the socket connection object for the next request. The number of saved socket connection objects is the Client Connection Size.

1. From the Command prompt, navigate to *DTSInstall***bin\admin**.

2. From the **bin\admin** subdirectory, connect to the server by entering the **RemoteAdmin** command, **server name**, and the **port** on which the server is listening. Enter the Command Line parameter **-status**.

   ```
   C:\Program Files\Apelon\DTS\bin\admin>remoteadmin kelvin 6666 -status
   ```

3. Press **Enter**. The resulting list indicates the active thread count, active database connection count, number of threads, and the client connection size.

   ```
   C:\Program Files\Apelon\DTS\bin\admin>remoteadmin kelvin 6666 -status

   Default Logging Loaded from a Jar.
   Active thread count is 2
   Active database connection count is 0
   Current number of threads 30
   Client connection size is 0
   ```

Ping Test

A Ping test lets you determine if the server is responding. The Ping tool checks for the presence of your site, and measures the amount of time required for a response to return to the server.

1. From the Command prompt, navigate to *DTSInstall***bin****admin**.

2. From the **bin\admin** subdirectory, connect to the server by entering the **RemoteAdmin** command, **server name**, and the **port** on which the server is listening. Enter the Command Line parameter -**ping**.

```
C:\Program Files\Apelon\DTS\bin\admin>remoteadmin kelvin 6666 -ping
```

3. Press **Enter**. The parameter **-ping** returns the date and time the ping test was completed.

```
C:\Program Files\Apelon\DTS\bin\admin>remoteadmin kelvin 6666 -ping

Default Logging Loaded from a Jar.
Ping Server: Fri Nov 12 15:46:43 EST 2004
```

Reset User Information

Use the remote server administration tool to modify user information (username and password). The DTS Server recognizes the changes to the user information immediately.

1. From the Command prompt, navigate to *DTSInstall***bin****admin**.

2. From the **bin\admin** subdirectory, connect to the server by entering the **RemoteAdmin** command, **server name**, and the **port** on which the server is listening. Enter the Command Line parameter **-resetusers**.

```
C:\Program Files\Apelon\DTS 3.4\bin\admin>remoteadmin.bat localhost 6666 -ss -us
er admin -pass admin -resetusers_
```

3. Press **Enter**. The username and password are reset immediately. Note that if a user is logged into the DTS Editor, and the password is changed using UserManager and reset as described here, the user will be required to log into the DTS Editor again using the updated logon information.

    Note also that if the **role** of a DTS Workflow user is changed, the password should be changed as well. After the password is changed, the user will need to log into the DTS Editor again using the updated logon information.

Shutdown

There are two server shutdown options from **RemoteAdmin**. The first shutdown option process includes a warning message indicating that all servers will be shut down (including the **admin** server) and cannot be restarted with RemoteAdmin; confirmation is required to complete shutdown. The second shutdown option does not include a confirmation message; this process shuts down the server immediately with no confirmation required.

Shutdown (Non-secure Socket Connection)

Follow this procedure to shut down the server when there is a non-secure socket connection.

1. From the Command prompt, navigate to *DTSInstall***bin****admin**.

2. From the **bin****admin** subdirectory, connect to the server by entering the **RemoteAdmin** command**, server name**, and the **port** on which the server is listening. Enter the Command Line parameter -**shutdown**.

```
C:\Program Files\Apelon\DTS\bin\admin>remoteadmin kelvin 6666 -shutdown
```

3. Press **Enter**. A message displays indicating that all servers will shut down, and prompts you for confirmation.

```
WARNING: this shutdown command will shut down all servers including admin server
.Once you shut down, you can't restart any serverthrough admin server.
Do you still want to shut down? Enter yes or no.
```

Enter **yes** to shut down servers, or **no** to cancel the shutdown (**servers keep running!** displays if you enter **no**).

To shut down servers without confirmation, enter **-shutdown -noprompt**.

```
C:\Program Files\Apelon\DTS\bin\admin>remoteadmin kelvin 6666 -shutdown -noprompt
t_
```

Press **Enter**. A message displays indicating successful shut down of servers.

```
.AdminClient kelvin 6666 -shutdown -noprompt
Default Logging Loaded from a Jar.
shutdown server successful
Press any key to continue . . .
```

Shutdown (Secure Socket Connection)

Follow this procedure to shut down the server when there is a secure socket connection.

1. From the Command prompt, navigate to *DTSInstall***bin****admin**.

2. From the **bin****admin** subdirectory, connect to the server by entering the **RemoteAdmin** command, the **server name**, the **port** on which the server is listening, **-ss** (for **secure socket**) the **username**, and the user **password**.

The **DTS Username** and **Password** were generated by running the UserManager utility on AIX. Note: For *Windows* client, you can create the DTS **Username** and **Password** by running **UserManager.bat**, installed on the *Windows* platform, or through the User Manager GUI utility.

Enter the Command Line parameter **-shutdown**.

```
C:\Program Files\Apelon\DTS\bin\admin>remoteadmin swing 6666 -ss -user admin -pa
ss admin -shutdown
```

3. Press **Enter**. A message displays indicating that all servers will shut down, and prompts you for confirmation.

```
WARNING: this shutdown command will shut down all servers including admin server
.Once you shut down, you can't restart any server through admin server.
Do you still want to shut down? Enter yes or no.
```

Enter **yes** to shut down the server, or **no** to cancel the shutdown.

If you entered **-shutdown –noprompt**, shutdown occurs without the confirmation.

Tomcat Server

Configure the Tomcat Server

Tomcat is a server application that is separate from the DTS Server. Tomcat is designed specifically to be a Web server, and is used with the Apelon DTS program suite to support operation of the DTS Browser. The DTS Browser is a client service used to perform searches of an Apelon Knowledgebase across either a company intranet or through the Internet. Tomcat must be running in order for you to access the Knowledgebase using the DTS Browser from any location.

Tomcat requires little user intervention in the course of its operation. The System Administrator needs only to start the Tomcat Server to provide Intra/Internet access to the Knowledgebase.

The Tomcat Server is pre-configured when delivered, and should require no changes. If configuration changes become necessary, the following procedures can be used to change the port number or DTS Browser path.

Configure the Tomcat Port Number

The Tomcat installation is preset to use port 8081. In some installations it may be necessary to modify this port for proper Tomcat operation. For example, if you have multiple versions of DTS (e.g., **DTS 3.4**, **DTS 3.5**, etc.) installed on the same machine, you should modify the port number for the current installed version.

1. Navigate to *DTSInstall***tomcat\conf**.

2. Locate the file **server.xml** and open it in Notepad.

3. Locate the following section:

```
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
<Connector className="org.apache.catalina.connector.http.HttpConnector"
           port="8081" minProcessors="5" maxProcessors="75"
           enableLookups="true" redirectPort="8443"
           acceptCount="10" debug="0" connectionTimeout="60000"/>
```

4. Change the port number in: **port="8081"** to the port number required.

5. Save the file.

6. Restart the Tomcat Server to implement the change.

Note: If you change the port number, the DTS Browser will be running at
http://localhost:[yourPortNumber]/[dtstreebrowser]/index.jsp.
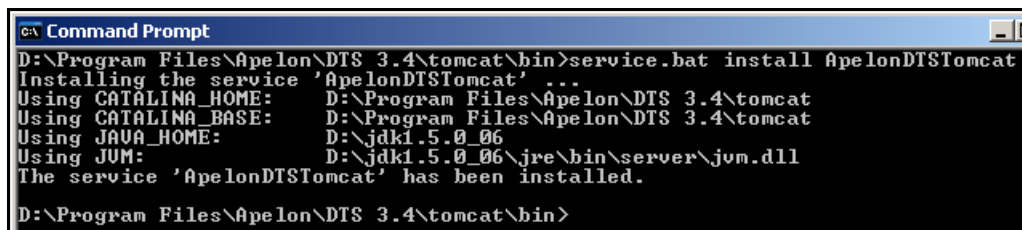
Run Tomcat Server

Tomcat must be running in order for any user to access the Knowledgebase using the **DTS Browser** from any location. To start the Tomcat Server on a *Windows* installation, select **Programs>Apelon>***DTSInstall***>Start Tomcat** from the *Windows* **Start** menu. For a Linux installation, execute *runtomcat.sh* in **bin/browser** (using the **start** argument) to start the Tomcat Server. Once started, Tomcat should not require any Administrative support.

To stop the Tomcat Server on a *Windows* installation, select **Start>Programs>Apelon>** *DTSInstall***>Stop Tomcat** from the *Windows* **Start** menu. For a Linux installation, execute *runtomcat.sh* in **bin/browser** (using the **stop** argument) to stop Tomcat.

Install and Run Tomcat as a Service

Follow this procedure to install the files necessary to run Tomcat (i.e., the version of Tomcat bundled with DTS) as a service, then start the Tomcat service. This procedure is provided for reference purposes; Apelon does not support running Tomcat as a service in any version of DTS.

1. Open the **tomcatservicefiles.zip** file provided by Apelon, then extract the **tomcat5.exe** and **tomcat5w.exe** files within to *DTSInstall***\tomcat\bin**.

2. Open the Command window, navigate to DTS\tomcat\bin, and enter **service.bat install ApelonDTSTomcat** (this installs **ApelonDTSTomcat** as a service).
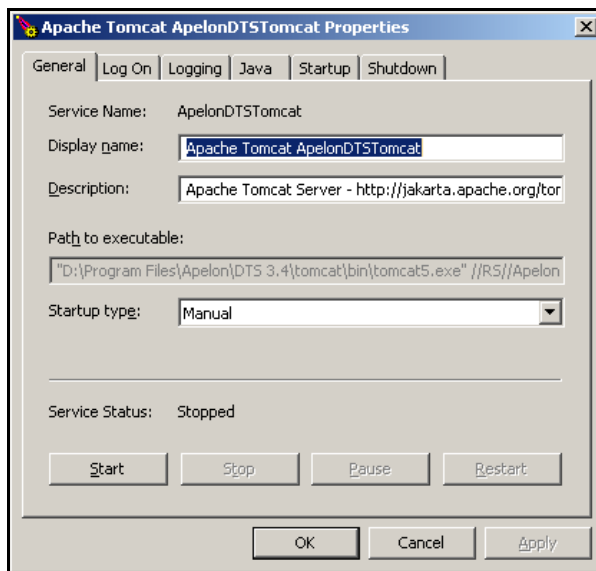

3. In the *Windows* Explorer, make a copy of the **tomcat5.exe** file and rename the new file **ApelonDTSTomcat.exe**, then make a copy of the **tomcat5w.exe** file and rename the new file **ApelonDTSTomcatw.exe**.
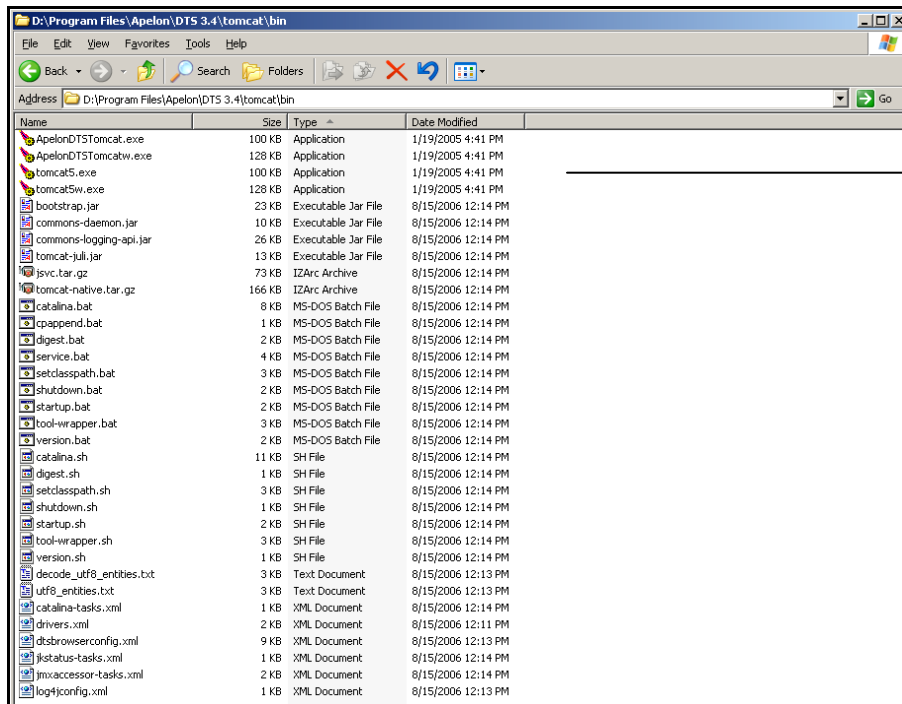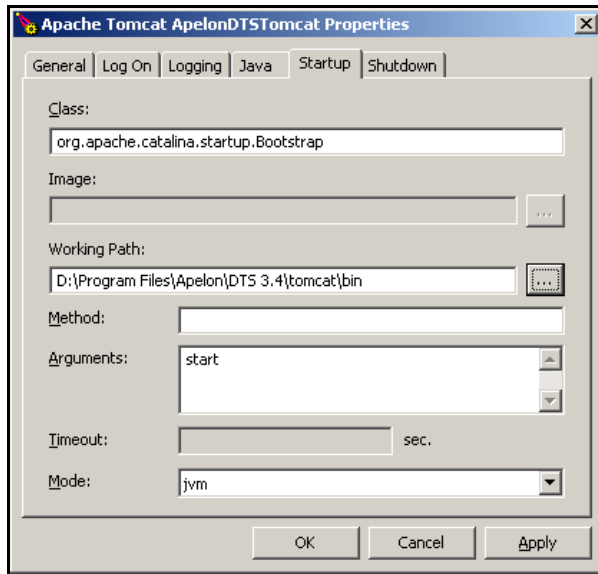
Rename Files

4. Double click **ApleonDTSTomcatw.exe**. The *Apache Tomcat ApelonDTSTomcat Properties* window displays; change the service *Display name* here, as needed.
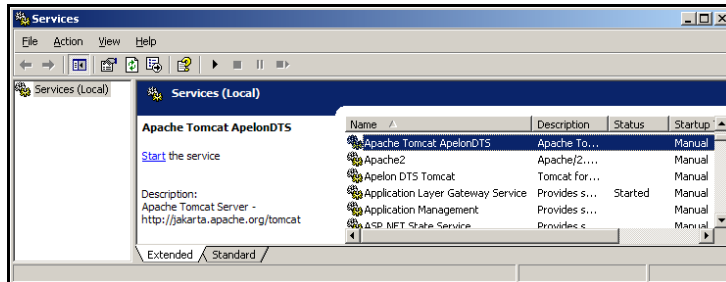

5. Click the **Startup** tab. For the *Working Path*, enter ***DTSInstall\Tomcat\bin***.

Click **OK.** Setup for service **ApelonDTSTomcat** is now completed.

6. To start Tomcat as a service, access the *Services* window (**Control Panel>Administrative Tools>Services**), then locate and start **Apache Tomcat ApelonDTS** service.

**Start Tomcat Service**

Password Encryption and Decryption

Your Apelon DTS installation includes the **Apelon Encrypter/Decrypter**. The **Encrypter** allows you to encrypt passwords in selected XML configuration files in which password authentication is established. The **Decrypter** allows you to convert encrypted password values to clear text values in the encrypted XML file.

Note that the Decrypter resets the password values that existed in the original XML file. The Encrypter/Decrypter tool should be made available to System Administration personnel only in order to preserve password security.
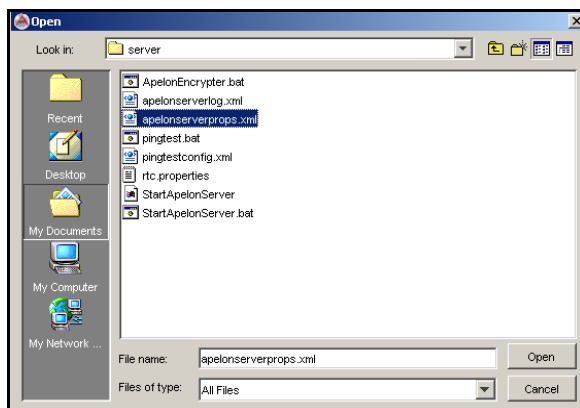
Note the procedure for running the Encrypter/Decrypter.

1. For a *Windows* installation, run **ApelonEncrypter.bat** in *DTSInstall***\bin\server**. For a Linux installation, run **ApelonEncrypter.sh** in **bin/server**.
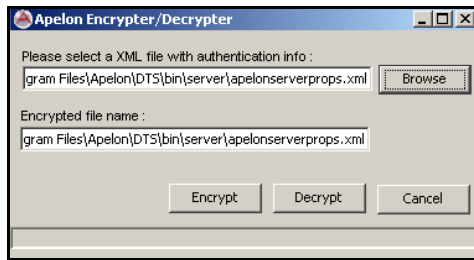
   The *Apelon Encrypter/Decrypter* window displays.
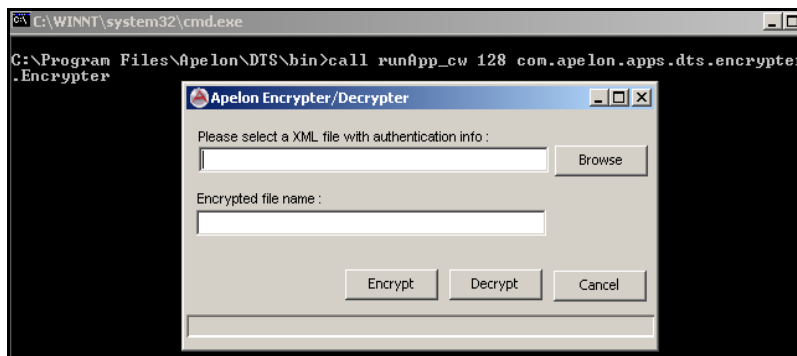
2. Click **Browse** to search for the XML file for which you want to encrypt passwords. The following *Browse* window displays.


3. Browse to the desired XML file that includes password authentication, then click **Open** to select it. The *Apelon Encrypter/Decrypter* window displays again, reflecting your file selection, and the location where it resides.

4. Click **Encrypt**. If the file is encrypted already, a message displays indicating this; select another file for encryption.

   If the XML file was not encrypted previously, the *Apelon Encrypter/Decrypter* window indicates that the file you selected has been encrypted successfully.
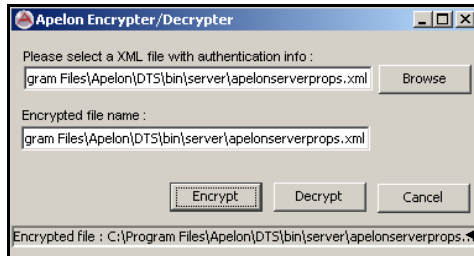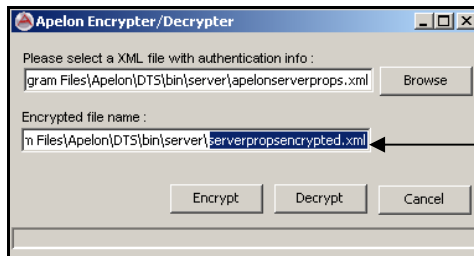

Encryption Completed

   All values in the selected XML file with the property name "**pass**" are encrypted at this point. Note the highlighted encrypted value.

```
<property name="pass" value="enc(xlgILoPPcTvGWAgug89xO8ZYCC6Dz3E7WA9AUZUaZSI=" />
```

   The prefix **enc(** indicates an encrypted password value.

5. You have the options of creating a new encrypted XML file with a new file name (i.e., the original file is left untouched) and also specifying another location for the renamed, encrypted file. Make any desired changes in the *Encrypted file name* field, then click **Encrypt**. (note the illustration).

Specify New Encrypted File Name

The new encrypted file is created in the location you designated.

6. To decrypt an encrypted XML file, browse to the desired file, then click **Decrypt**. The *Apelon Encrypter/Decrypter* window indicates that the file you selected has been decrypted successfully.
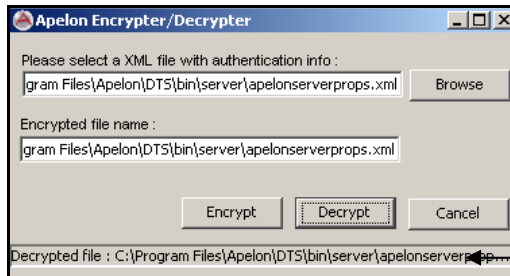
Decryption Completed

All of the encrypted values in the selected XML file with the property name "**pass**" are replaced by clear text values at this point.

Common Error Messages

Error messages can be frustrating, but they're not impossible to overcome. In fact, you can avoid most error messages by using a few simple techniques. We have catalogued some of the most common Apelon DTS error messages and provided practical solutions that will prevent you from getting them.

If you are having trouble with Apelon DTS Server, you should first check the error logs. The error logs can be found in the **bin\logs** subdirectory, and are named AppLog and AdminServerLog.

Error Message	Possible Cause	Solution
0403-006 Execute permission denied.	The correct permissions were not assigned to the StartApelonServer file.	Refer to the *Assign Execute Permission on AIX* discussion for procedures on providing the **StartApelonServer** file the appropriate execute permissions.
IO error creating socket when Testing Remote Admin server.	I/O error occurs when creating the socket and establishing the connection. The IP address of the host or route to the host could not be determined.	Ensure that the correct server name and port are entered.
Receive a "connect" error attempting to run the client.	Apelon DTS Server is not running.	Start the Apelon DTS Server before running the client.
Client server cannot connect to the database.	Oracle is not enabled.	Ensure Oracle is running so the server can access the database.

Report Problems

Report problems to Apelon by accessing the Apelon e-customer site at http://support.apelon.com and completing the Support Request Form. A second option is to send an e-mail message to **support@apelon.com**. Please specify the version of Apelon DTS you are using. It is helpful to include log configurations files, if any. A short example reproducing the problem is very much appreciated.

Appendix A – The DTS Server on AIX

Assign Execute Permission on AIX

Before attempting to start the DTS server on AIX, you must give the **StartApelonServer** file execute permission. If execute permission is not assigned, an error message displays when you start the server.

```
-> StartApelonServer
ksh: StartApelonServer: 0403-006 Execute permission denied.
apelon@millennium:/apps/apelon/OntylogDTS/bin
->
```

1. At the Command prompt, type **ls –l** and press **Enter**. The current permissions for all the files in the **bin/admin** subdirectory display. Note that permissions for **StartApelonServer** are for read and write.

```
-rw-r--r--    1 apelon    staff        1858 Jul 23 11:22 ApelonServerLog.xml
-rw-r--r--    1 apelon    staff         494 Jul 23 11:22 ApelonServerProps.xml
-rw-r--r--    1 apelon    staff         235 Jul 23 11:22 RemoteAdmin
-rw-r--r--    1 apelon    staff         470 Jul 23 11:22 StartApelonServer
-rw-r--r--    1 apelon    staff         156 Jul 23 11:22 apelprops.dtd
apelon@millennium:/apps/apelon/OntylogDTS/bin
```

2. Assign permission to execute StartApelonServer. At the prompt, type **chmod +x StartApelonServer** and press **Enter**.

3. To verify that permission has been granted, type **ls –l**.

```
-rwxr-xr-x    1 apelon    staff         470 Jul 23 11:22 StartApelonServer
```

4. Run the Apelon DTS Server. At the prompt, enter **StartApelonServer**.

Run the Apelon DTS Server with Telnet

After all the files have been copied to the AIX server, you can run the Apelon DTS Server using Telnet.

1. Begin Telnet. At the prompt, type **Telnet** and press **Enter**.

2. Type **open**, then type the Telnet **IP Address** of the AIX machine.

3. Enter the Telnet **user name** and **password**.

4. Change the directory path to the **bin/server** subdirectory on the AIX server. Use the **pwd** command to ensure the correct location.

```
apelon@millennium:/apps/apelon
-> pwd
/apps/apelon
apelon@millennium:/apps/apelon
-> cd OntylogDTS4
apelon@millennium:/apps/apelon/OntylogDTS4
-> cd bin
apelon@millennium:/apps/apelon/OntylogDTS4/bin
->
```

5. Run the Server by entering the StartApelonServer command and the properties file. At the prompt, enter **StartApelonServer** and press **Enter**.

```
initing socketServer
setting class for queries class com.apelon.dts.server.QueryServer
initing admin server
starting socketServer
```

Assign Execute Permission on an AIX Machine to Access Remote Admin

Before you can access **RemoteAdmin** from an AIX machine, it is necessary to change the permissions of the file to grant execute permission (this is not necessary if using **RemoteAdmin.bat** in *Windows*).

1. Start Telnet. At the prompt, enter **Telnet** and press **Enter**.

2. Type **open** and the **IP Address** of the AIX machine.

3. Enter the Telnet **user name** and **password**.

4. Change to the **bin** directory to which the files have been copied.

5. At the Command prompt, type **ls –l** and press **Enter**. The current permissions for all files in the **bin\admin** directory display. Note that the permissions for RemoteAdmin are for read/write only.

```
-> ls -l
total 56
-rw-r--r--    1 apelon    staff        1858 Jul 13 10:51 ApelonServerLog.xml
-rw-r--r--    1 apelon    staff        1211 Jul 13 10:51 ApelonServerProps.xml
-rw-r--r--    1 apelon    staff           0 Jul 23 16:22 AppLog
-rw-r--r--    1 apelon    staff         235 Jul 23 15:54 RemoteAdmin
-rwxr-xr-x    1 apelon    staff         338 Jul 13 10:51 StartApelonServer
-rw-r--r--    1 apelon    staff         206 Jul 13 10:51 apelprops.dtd
-rwxr-xr-x    1 apelon    staff         338 Jul 13 10:51 StartApelonServer
-rw-r--r--    1 apelon    staff         206 Jul 13 10:51 apelprops.dtd
apelon@millennium:/apps/apelon/OntylogDTS/bin
-> _
```

6. Assign permission to execute RemoteAdmin. At the prompt type "**chmod +x RemoteAdmin**" and press **Enter**.

7. To verify that permission has been granted, type "ls –l".

```
-rwxr-xr-x    1 apelon    staff         235 Jul 23 15:54 RemoteAdmin
```