Oracle® Tuxedo

Using Oracle Tuxedo Distributed Caching (TDC)

12*c* Release 2 (12.2.2)

April 2016

Using Oracle Tuxedo Distributed Caching (TDC), 12*c* Release 2 (12.2.2)

Copyright © 1996, 2016, Oracle and/or its affiliates. All rights reserved.

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

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

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

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

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

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

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

This documentation is in pre-production status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

1. Overview

Caching Strategies 1-1

Data Caching for Clients and Servers 1-1

Result Caching for Oracle Tuxedo Services 1-2

Components and Deployment. 1-3

Administrator Tasks 1-4

Programmer Tasks 1-5

2. Oracle Tuxedo Distributed Caching (TDC) Administration

Configuring Oracle Coherence 2-1

tangosol-coherence-override.xml 2-2

coherence-cache-config.xml 2-3

Configuring Oracle Tuxedo Caching Server Module. 2-4

Configuring UBBCONFIG 2-4

Configuring Oracle Tuxedo Caching Server Module 2-5

Configure Oracle Tuxedo Distributed Caching (TDC) Property File 2-7

Configuring Data Caching for Clients and Servers 2-9

Configure Oracle Coherence 2-9

Start Oracle Coherence Cluster 2-10

Configure Oracle Tuxedo Caching Server Module. 2-10

Configuring Result Caching for Oracle Tuxedo Services 2-10

Configure Oracle Coherence 2-10

Start Oracle Coherence Cluster 2-10

Configure Oracle Tuxedo Caching Server Module. 2-10

Configure UBBCONFIG for TDC Result Caching. 2-10

3. Oracle Tuxedo Distributed Caching (TDC) Programming

Use Oracle Tuxedo Distributed Caching (TDC) Related ATMI APIs. 3-1

Put An Oracle Tuxedo Buffer to Cache 3-9
Get An Oracle Tuxedo Buffer from Cache. 3-10
Remove Oracle Tuxedo Buffer(s) from Cache. 3-11

4. Oracle Tuxedo Distributed Caching (TDC) Samples

Sample: Using Data Caching for Clients and Servers 4-1
Sample: Configure Oracle Coherence. 4-1
Sample: Start Oracle Coherence Cluster 4-4
Sample: Configure Oracle Tuxedo Caching Server Module. 4-4
Sample: Configure TMJAVASVR in UBBCONFIG. 4-6
Sample: Put An Oracle Tuxedo Buffer to Cache. 4-7
Sample: Get An Oracle Tuxedo Buffer from Cache 4-7
Sample: Remove An Oracle Tuxedo Buffer from Cache 4-8
Sample: Using Result Caching for Oracle Tuxedo Services 4-9
Sample: Configure VIEWTABLE. 4-9
Sample: Configure UBBCONFIG. 4-10
Sample: Set on Server Side 4-11
Sample: Set on Client Side 4-12

CHAPTER 1

Overview

This topic contains the following sections:

- Caching Strategies
- Components and Deployment
- Administrator Tasks
- Programmer Tasks

Caching Strategies

Oracle Tuxedo Distributed Caching (TDC) leverages Oracle Coherence and uses Oracle Tuxedo Caching Server Module (configured in Oracle Tuxedo Java server, which is working as a client of Oracle Coherence). This assures you taking all advantages that Oracle Coherence has for caching.

This feature supports the following caching strategies.

- Data Caching for Clients and Servers
- Result Caching for Oracle Tuxedo Services

Data Caching for Clients and Servers

When you enable data caching, you can store data in cache, and clients on other machines can retrieve the data from the cache. This offers you a new way of sharing data between clients and servers, especially sharing data for or from other servers.

This feature provides you

- High performance

  Storing and retrieving through cache is faster and lighter than before.

  Oracle Tuxedo also adopts other ways to achieve high performance, such as minimizing local buffer copies (ideally providing some zero copy use cases) and focusing on primarily read intensive operations (for example, make 2:1 read/write ratio or higher).

- Various buffer types support

  This feature supports many Oracle Tuxedo buffer types so that you do not need to manage to match your data with the data types that Oracle Tuxedo supports. The supported Tuxedo buffer types are `CARRAY`, `FML`, `FML32`, `MBSTRING`, `STRING`, `VIEW`, `VIEW32`, `XML`, `RECORD`, `X_C_TYPE`, `X_OCTET`, and `X_COMMON`.

Result Caching for Oracle Tuxedo Services

When you enable result caching, Oracle Tuxedo first reaches the result from the cache entry instead of reaching the backend service; if it fails to reach it or if the cache or cache entry is expired, Oracle Tuxedo reaches the backend service and the result is stored in the cache for Oracle Tuxedo to reach it next time.

This feature provides you

- High performance

  Synchronous services that return results that do not change often are good candidates to have their results cached by Oracle Tuxedo. This can improve performance by reducing network overhead to access the backend service.

  Oracle Tuxedo also adopts other ways to achieve high performance, such as minimizing local buffer copies (ideally providing some zero copy use cases) and focusing on primarily read intensive operations (for example, make 2:1 read/write ratio or higher).

- High availability and scalability

  By integrating with Oracle Coherence, Oracle Tuxedo takes advantage of its specialized scalable protocol and its creation of a cluster. A cluster can be seamlessly expanded to add more memory, processing power or both, and can avoid single point of failure as it transparently fails over if a cluster member fails. As a result this feature provides you a highly availability and scalability.

Also, taking advantage of Oracle Coherence, any cache entry can be replicated across two or more machines, and the data processing can be farmed out to where the data is and return results to you. This assures your data to be scalable.

- Various buffer types support

  This feature supports many Oracle Tuxedo buffer types so that you do not need to manage to match your data with the data types that Oracle Tuxedo supports. The supported Tuxedo buffer types are `CARRAY`, `FML`, `FML32`, `MBSTRING`, `STRING`, `VIEW`, `VIEW32`, `XML`, `RECORD`, and `X_COMMON`.

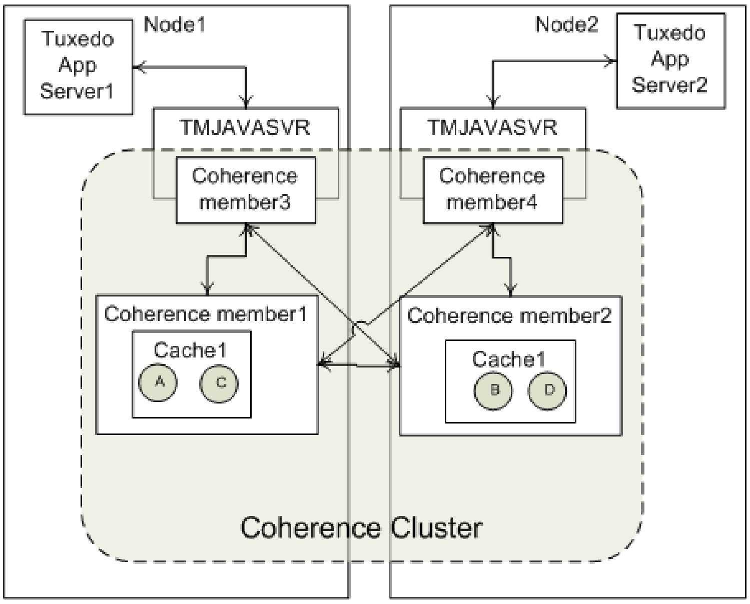
- Easy approach

  This feature can be transparent for you so you can use it without making any code changes, providing you an easy approach. Oracle Tuxedo encapsulates Oracle Coherence functions in Oracle Tuxedo Caching Server Module (in Oracle Tuxedo Java server container) so that you can just use Oracle Tuxedo Caching Server Module to implement all functions that pertain to this feature.

Components and Deployment

Figure 1-1 illustrates a typical deployment in an Oracle Tuxedo MP domain to use TDC based on Oracle Coherence.

Figure 1-1 Oracle Tuxedo Distributed Caching Components and Deployment



As you can see from this figure, `TMJAVASVR`, where Oracle Tuxedo Caching Server Module is configured, is taken as Oracle Coherence's client (member). You can directly use `TMJAVASVR` to cache results from multiple machines without worrying about how cluster members located in different machines communicate with each other.

Administrator Tasks

As Oracle Coherence and Oracle Tuxedo Caching Server Module are used, it requires you to configure them both before actually using TDC.

- Configuring Oracle Coherence
- Configuring Oracle Tuxedo Caching Server Module
- Configuring Data Caching for Clients and Servers
- Configuring Result Caching for Oracle Tuxedo Services

Programmer Tasks

- Use Oracle Tuxedo Distributed Caching (TDC) Related ATMI APIs
- Put An Oracle Tuxedo Buffer to Cache
- Get An Oracle Tuxedo Buffer from Cache
- Remove Oracle Tuxedo Buffer(s) from Cache

Oracle Tuxedo Distributed Caching (TDC) Administration

This topic contains the following sections:

- Configuring Oracle Coherence
- Configuring Oracle Tuxedo Caching Server Module
- Configuring Data Caching for Clients and Servers
- Configuring Result Caching for Oracle Tuxedo Services
- Configuring Cached Data Expiry Strategy
- Configuring for Propagating Execution Context ID (ECID) to Oracle Coherence

For a quick start, see the following samples.

- Sample: Using Data Caching for Clients and Servers
- Sample: Using Result Caching for Oracle Tuxedo Services

Configuring Oracle Coherence

Oracle Tuxedo Distributed Caching (TDC) uses Oracle Coherence as the caching engine, so Oracle Coherence must be installed and configured before you use Oracle Tuxedo Distributed Caching (TDC).

Configure the following files just like you configure on Oracle Coherence. See *Oracle Fusion Middleware Developing Applications with Oracle Coherence* for detailed instruction.

- tangosol-coherence-override.xml
- coherence-cache-config.xml

You can deploy the above configuration files into any path as long as this path is in the java class path and prior to where `coherence.jar` is. For example, you can put the configuration files into `${APPDIR}/config` and then start Oracle Coherence server like this:

Listing 2-1 Oracle Coherence Cluster Deployment

```
java -server -showversion $JAVA_OPTS -Dtangosol.coherence.mode=prod -cp
$APPDIR/config: ${COHERENCE_HOME}/lib/coherence.jar
com.tangosol.net.DefaultCacheServer
```

tangosol-coherence-override.xml

Listing 2-2 is an example; note the properties in bold. In this example, `coherence_tux` is the name of the Oracle Coherence cluster whose multicast port number is `51697` and unicast port number is `51687`.

Listing 2-2 tangosol-coherence-override.xml

```
<?xml version='1.0'?>

<coherence  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"


xmlns="http://xmlns.oracle.com/coherence/coherence-operational-config"


xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-operational-co
nfig coherence-operational-config.xsd">

  <cluster-config>

    <member-identity>

      <cluster-name
system-property="tangosol.coherence.cluster">coherence_tux</cluster-name>

    </member-identity>
    <unicast-listener>
      <address
system-property="tangosol.coherence.localhost">localhost</address>
      <port system-property="tangosol.coherence.localport">51687</port>
    </unicast-listener>
    <multicast-listener>
        <port system-property="tangosol.coherence.clusterport">51697</port>
    </multicast-listener>
  </cluster-config>
</coherence>
```

coherence-cache-config.xml

Listing 2-3 is an example; note the properties in bold. In this example, we create an Oracle Coherence cache named `tux_distributed`.

Listing 2-3 coherence-cache-config.xml

```
<?xml version="1.0"?>
<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"

xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-cache-config
coherence-cache-config.xsd">
  <caching-scheme-mapping>
    <cache-mapping>
      <cache-name>tux_distributed</cache-name>
      <scheme-name>distributed</scheme-name>
    </cache-mapping>
  </caching-scheme-mapping>
  <caching-schemes>
    <distributed-scheme>
      <scheme-name>distributed</scheme-name>
      <service-name>DistributedCache</service-name>
        <lease-granularity>member</lease-granularity>
      <backing-map-scheme>
        <local-scheme/>
      </backing-map-scheme>
      <autostart>true</autostart>
    </distributed-scheme>
  </caching-schemes>
</cache-config>
```

Configuring Oracle Tuxedo Caching Server Module

- Configuring UBBCONFIG
- Configuring Oracle Tuxedo Caching Server Module
- Configure Oracle Tuxedo Distributed Caching (TDC) Property File

Configuring UBBCONFIG

Oracle Tuxedo Caching Server Module runs in Tuxedo Java server, so `TMJAVASVR` must be configured in `UBBCONFIG`. `TMJAVASVR` uses multi-threaded configuration to improve performance. Multi-instances configuration is also used to gain higher availability.

Listing 2-4 UBBCONFIG for TMJAVASVR

```
*RESOURCE
...
```

```
MODEL          SHM

...

*MACHINES

"m1"           LMID=L1

...

*GROUPS

JGRP1  LMID=L1 GRPNO=10


...

TMJAVASVR    SRVGRP=JGRP1 SRVID=10

              MINDISPATCHTHREADS=4 MAXDISPATCHTHREADS=4 MIN=2 MAX=2

              CLOPT="-- -c /home/scott/tuxedo/dom1/config/tdcsvr_coh.xml"

...
```

Configuring Oracle Tuxedo Caching Server Module

Oracle Tuxedo Distributed Caching (TDC) introduces Oracle Tuxedo Caching Server Module, which converts the caching request to Oracle Coherence and sends back the reply. It works as a client of Oracle Coherence.

You should configure this Oracle Tuxedo Caching Server Module in Oracle Tuxedo Java Server Configuration File; the package that Oracle Tuxedo Java server uses to enable TDC is `com.oracle.tuxedo.tjtdc.jar` located in `${TUXDIR}/udataobj/tuxj/tdc`.

Listing 2-5 is an example of Oracle Tuxedo Java Server Configuration File that enables TDC. `-f` option of the `<server-clopt>` element is newly added for this feature and is used to specify the path of TDC property file. For more information of other elements and options, see "Java Server Configuration Schema File for version 2.0" in *Setting Up An Oracle Tuxedo Application*.

- In general, you should not change the code lines in bold. The class `com.oracle.tuxedo.tdc.TCache4Coherence` is the main class used by TDC which is located in `${TUXDIR}/udataobj/tuxj/tdc/com.oracle.tuxedo.tjtdc.jar` .

- It is necessary to change the code lines in Italic due to different environment. Note that `<APPDIR>` must be replaced with the real path to make the file work.

Notes:

- One or more Oracle Tuxedo Caching Server Module can be configured on a Tuxedo machine.
- TDC supports inter-domain caching, as long as the Oracle Tuxedo Caching Server Modules of different Tuxedo domains are members of the same Oracle Coherence cluster.
- The class path `${APPDIR}/config</classpath>` must be prior to the class path `${COHERENCE_HOME}/lib/coherence.jar`.
- When using JAVA8 on the platforms where IPv4 and IPv6 stacks are validated simultaneously, you need to specify IPv4 stack explicitly:

  ```
  <java-config>
  …
  <jvm-options>-Djava.net.preferIPv4Stack=true</jvm-options>
  …
  </java-config>
  ```

Listing 2-5 Configuring Oracle Tuxedo Caching Server Module in Oracle Tuxedo Java Server Configuration File

```
<?xml version="1.0" encoding="UTF-8"?>
<TJSconfig version="2.0">
    <java-config>
        <jvm-options>-XX:MaxPermSize=192m</jvm-options>
        <jvm-options>-server</jvm-options>
<jvm-options>-Dtangosol.coherence.distributed.localstorage=false</jvm-options>
        <jvm-options>-Dtangosol.coherence.mode=prod</jvm-options>
    </java-config>
<tux-config>
        <server-clopt>-f <APPDIR>/config/tdcsvr_coh.conf</server-clopt>
```

```
    </tux-config>
    <classpath-config>
    </classpath-config>
    <tux-resources>
    </tux-resources>
    <jdbc-resources>
    </jdbc-resources>
    <tux-server-config>
        <classpath>${TUXDIR}/udataobj/tuxj/tdc/com.oracle.tuxedo.tjtdc.jar
</classpath>
        <classpath>${TUXDIR}/udataobj/tuxj/tdc/dms.jar</classpath>
        <classpath>${TUXDIR}/udataobj/tuxj/tdc/ojdl.jar</classpath>
        <classpath>${APPDIR}/config</classpath>
        <classpath>${COHERENCE_HOME}/lib/coherence.jar</classpath>
        <server-class name="com.oracle.tuxedo.tdc.TCache4Coherence">
        </server-class>
    </tux-server-config>
</TJSconfig>
```

Configure Oracle Tuxedo Distributed Caching (TDC) Property File

It is required to add a new property file which specifies all properties about the caches for TDC. Oracle Tuxedo Caching Server Module advertises services using the name of these caches. You can define many Oracle Tuxedo caches in a single property file. See Oracle Tuxedo Distributed Caching (TDC) Property File Properties for this file's properties.

Listing 2-6 shows a template of the property file. In this template, two Oracle Tuxedo cache names are configured: `tc1` and `tc2`. `tc1` uses Oracle Coherence cache `tux_distributed` and `tc2` uses Oracle Coherence cache `tux2_distributed`.

You can find this template in `$TUXDIR/udataobj/tuxj/tdc/tdcsvr_coh.conf.template`. You can change the name and the location of this property file; if so, make sure you make the same change for the `<server-clopt> -f` option in Oracle Tuxedo Java Server Configuration File (see Listing 2-5 for an example).

Listing 2-6 TDC Property File Template

```
#**********global level settings**********

#* Encoding setting, optional, defaults to "no"
#* It can be overridden by the cache level setting.
#options.encoding=yes

#* Logging setting, optional, defaults to "no"
#* There is no cache level logging setting.
#options.logging=yes

#**********cache level settings**********

#* Configurations for Tuxedo cache "tc"
#* Encoding setting, optional, defaults to "no"
#cache.options.encoding.tc=yes
#* Cache used in Oracle Coherence, required
coh.cache.name.tc=tux_distributed

#* Configurations for Tuxedo cache "tc2"
#* Encoding setting, optional, defaults to "no"
#cache.options.encoding.tc2=yes
#* Cache used in Oracle Coherence, required
```

```
coh.cache.name.tc2=tux_distributed
```

Oracle Tuxedo Distributed Caching (TDC) Property File Properties

Oracle Tuxedo TDC property file is a file in a simple line-oriented format.

Properties are processed in terms of lines. There are two kinds of line, natural lines and logical lines.

A natural line is defined as a line of characters that is terminated either by a set of line terminator characters (`\\n` or `\\r` or `\\r\\n`) or by the end of the stream. A natural line may be either a blank line, a comment line, or hold all or some of a key-element pair.

A logical line holds all the data of a key-element pair, which may be spread out across several adjacent natural lines by escaping the line terminator sequence with a backslash character `\\`.

A natural line that contains only white space characters is considered blank and is ignored. A comment line has an ASCII `'#'` or `'!'` as its first non-white space character; the key contains all of the characters in the line starting with the first non-white space character and up to, but not including, the first unescaped `'=', ':'`. Any white space after the key is skipped; if the first non-white space character after the key is `'='` or `':'`, then it is ignored and any white space characters after it are also skipped. All remaining characters on the line become part of the associated element string; if there are no remaining characters, the element is the empty string `""`.

If there are several properties with the same key, the last property of them will be used.

Table 2-1 lists the supported TDC property file properties.

Table 2-1 Oracle Tuxedo Distributed Caching (TDC) Property File Properties

Property	Description
`options.encoding`	Values can be • `yes` (all caching data must be encoded) • `no` (default) This value can be overridden by `cache.options.encoding.[cachename]`.
`cache.options.encoding.[cachename]`	`yes` indicates all caching data in Oracle Tuxedo cache `[cachename]` must be encoded. It should be enabled when caching users are located in machines that have different data representation. If this property is not set, `options.encoding` value is used.
`coh.cache.name.[cachename]`	Indicates the used cache name in the Coherence cluster for the Oracle Tuxedo cache `[cachename]`. This is necessary for TDC property file if Oracle Coherence is used for TDC.
`options.logging`	The value of `yes` indicates all caching requests will be logged into coherence log. It is set to `no` by default.

Notes:

- `[cachename]` must be 78 characters or less in length. A service with name `[cachename]` will be advertised by Oracle Tuxedo Java Server.
- When you set `options.logging=yes`, the log level is 6 (and you cannot change this default value). Therefore, in Oracle Tuxedo Coherence configuration file, you should specify the element `<severity-level>` no less than 6 in order to print logging information.

Use TMIB to Dynamically Configure Oracle Tuxedo Distributed Caching (TDC) Property File Properties

You can use TMIB to dynamically configure TDC property file properties.

Table 2-2 Use TMIB to Configure Oracle Tuxedo Distributed Caching (TDC) Property File Properties

Property	Description
`options.encoding`	Reboot the client to take effect.
`cache.options.encoding.[cachename]`	Reboot the client to take effect.
`coh.cache.name.[cachename]`	Dynamically takes effect.
`options.logging`	Dynamically takes effect.

Oracle Tuxedo Java Server introduces a new class called `T_TJS`, which is used for this feature. Using this, you can modify, add, and delete a TDC property. Note that, this dynamic configuration does not affect the cached data in Coherence; removing a cache mapping does not mean clearning the cached data.

After dynamic configuration, the comments in the original property file are removed, and the original sequence of the properties cannot be kept in the new property file. The original property file is backed up with the extension name `.bak`.

See Listing 2-7 for an interface example.

Listing 2-7 Interface to configure TDC Property by TMIB

```
SRVCNM   .TMIB

TA_CLASS      T_TJS         /*must be T_TJS*/

TA_OPERATION  SET           /*must be SET*/

TA_SRVGRP     xxxx          /*TMJAVASVR Group Identifier*/

TA_SRVID             xxxx          /*the server id of TMJAVASVR which
support TDC*/

TA_TJS_CLASS  T_TJSAPPCMD   /* must be T_TJSAPPCMD */

TA_COMMAND    TA_TDC_UPDATECFG   /*must be TA_TDC_UPDATECFG */

TA_TDC_PROPERTY          xxxx         /* the TDC property*/

TA_TDC_PRO_UPDATEMODE  xxxx         /* the operation on TDC property*/
```

We use `TA_TDC_PROPERTY` attribute to set the property, and use `TA_TDC_PRO_UPDATEMODE` attribute to indicate the operation mode which can be set as `delete`, `new`, or `modify`. The value for `TA_TDC_PROPERTY` is key-value pair of TDC property when operation is `new` or `modify`, and is the key of TDC property when operation is `delete`. Note that, the separator for key-value pair only can be "=".

See Listing 2-8 for a sample to add properties to TDC property file (Run command: "`ud32 -C tpsysadm<setProperty`"); see Listing 2-9 for a sample to delete properties from TDC property file.

Listing 2-8 Edit a ud32 Input File: setproperty

```
SRVCNM   .TMIB

TA_CLASS         T_TJS

TA_OPERATION     SET

TA_SRVGRP        JGRP1

TA_SRVID         10

TA_TJS_CLASS     T_TJSAPPCMD

TA_COMMAND       TA_TDC_UPDATECFG

TA_TDC_PROPERTY coh.cache.name.cache1=tux_distributed

TA_TDC_PROPERTY cache.options.encoding.cache1=yes

TA_TDC_PROPERTY options.encoding=yes

TA_TDC_PROPERTY options.logging=yes

TA_TDC_PRO_UPDATEMODE   new
```

Listing 2-9 Delete Properties from TDC Property File

```
SRVCNM   .TMIB

TA_CLASS         T_TJS
```

```
TA_OPERATION     SET

TA_SRVGRP        JGRP1

TA_SRVID         10

TA_TJS_CLASS     T_TJSAPPCMD

TA_COMMAND       TA_TDC_UPDATECFG

TA_TDC_PROPERTY coh.cache.name.cache1

TA_TDC_PROPERTY cache.options.encoding.cache1

TA_TDC_PROPERTY options.encoding

TA_TDC_PROPERTY options.logging

TA_TDC_PRO_UPDATEMODE   delete
```

Configuring Data Caching for Clients and Servers

Follow these steps for configuring data caching for clients and servers.

- Configure Oracle Coherence
- Start Oracle Coherence Cluster
- Configure Oracle Tuxedo Caching Server Module

Configure Oracle Coherence

For how to configure Oracle Coherence, see Configuring Oracle Coherence.

Start Oracle Coherence Cluster

If there is no Oracle Coherence Cluster is running, start your own cluster, making sure you configure the path of the configuration files into the Java Class Path and ahead of where `coherence.jar` is. See Listing 2-1 for an example.

Configure Oracle Tuxedo Caching Server Module

See Configuring Oracle Tuxedo Caching Server Module for instruction.

Configuring Result Caching for Oracle Tuxedo Services

Follow these steps for configuring result caching for Oracle Tuxedo services.

- Configure Oracle Coherence
- Start Oracle Coherence Cluster
- Configure Oracle Tuxedo Caching Server Module
- Configure UBBCONFIG for TDC Result Caching

You can also use MIB to dynamically make changes for TDC.

- Use MIB to Dynamically Make Changes for TDC Result Caching

Configure Oracle Coherence

See Configuring Oracle Coherence for instruction.

Start Oracle Coherence Cluster

If there is no Oracle Coherence Cluster is running, start your own cluster, making sure you configure the path of the configuration files into the Java Class Path and ahead of where `coherence.jar` is. See Listing 2-1 for an example.

Configure Oracle Tuxedo Caching Server Module

See Configuring Oracle Tuxedo Caching Server Module for instruction.

Configure UBBCONFIG for TDC Result Caching

Configure TDC Result Caching on `UBBCONFIG`. See *Section 5 - File Formats, Data Descriptions, MIBs, and System Processes Reference* for more information.

- SERVICES Section
- CACHING Section

SERVICES Section

Specify `CACHING=string_value` as the name of the caching criteria used for caching for this service.

CACHING Section

Specify this CACHING section on UBBCONFIG.

- CACHING_CRITERIA_NAME required_parameters (Mandatory)
- CACHENAME=string_value (Mandatory)
- KEY_BUFTYPE="type1[:subtype1[,subtype2 . . .]][;type2[:subtype3[, . . .]]] . . ." (Mandatory)
- KEY="[mykey1][_$request_][mykey2]" string_value (Optional)
- KEY_FIELD= "field1[+field2[+field3[+¡-]]]" (Optional)

Listing 2-10 shows an example, where

- Svccache1 uses Oracle Tuxedo cache tc1. Other configurations are default. (It means KEY=$service+$request and KEY_BUFTYPE=STRING).
- Svccache2 uses Oracle Tuxedo cache tc1. The request is used to figure out the key when caching the response data. Other configurations are default. (It means KEY_BUFTYPE=STRING).
- Svccache3 uses Oracle Tuxedo cache tc1. The fixed string key1 is used as the key. Other configurations are default.
- Svccache4 uses Oracle Tuxedo cache tc1. The buffer type of the request message to the service is VIEW32 whose subtype is mystruct1. The value of the field name in the subtype mystruct1 is used as the key.
- Svccache5 uses Oracle Tuxedo cache tc1. The buffer types of the FML32 and VIEW32: mystruct1 have the same field1 and field2 (name and data type should be the same; value can be different); the request message will use the values of field1 and field2 as the key.

Listing 2-10 UBBCONFIG CACHING Section Configuration

```
...

*CACHING

Svccache1

        CACHENAME="tc1"
```

```
Svccache2

        CACHENAME="tc1"

        KEY="$request"


Svccache3

        CACHENAME="tc1"

        KEY="key1"


Svccache4

        CACHENAME="tc1"

        KEY="$request"

        KEY_BUFTYPE="VIEW32:mystruct1"

        KEY_FIELD="name"


Svccache5

        CACHENAME="tc1"

        KEY="mykey_$request"

        KEY_BUFTYPE="FML32;VIEW32:mystruct1"

        KEY_FIELD="field1+field2"


...
```

Use MIB to Dynamically Make Changes for TDC Result Caching

You can use MIB to dynamically make changes for TDC Result Caching. The following sub-sections explain TDC Result Caching related MIB attributes. See *Section 5 - File Formats, Data Descriptions, MIBs, and System Processes Reference* for more information.

- `T_SERVICE` Class Definition
  - `TA_CACHINGNAME`
- `T_CACHING` Class Definition
  - `TA_CACHING_NAME`
  - `TA_CACHING_CACHENAME`
  - `TA_CACHING_KEY`
  - `TA_CACHING_KEY_BUFTYPE`
  - `TA_CACHING_KEY_FIELD`
  - `TA_STATE`

Configuring Cached Data Expiry Strategy

Use <expiry-delay> Subelement Supported by Coherence

In Oracle Coherence, you can add a subelement called <expiry-delay> to <local-scheme> causes entries to automatically expire a cache if it is not updated for a given time interval, and, when the expired the cache invalidates the entry, remove it from the cache. See for more information.

Listing 2-11 Example configutation for a Cache with Expiring Entries

```
<caching-schemes>

<local-scheme>

<scheme-name>local</scheme-name>

<expiry-delay>10ms</expiry-delay>

</local-scheme>

</caching-schemes>
```

See *Oracle Fusion Middleware Developing Applications with Oracle Coherence* for more information.

Automatically Delete the Cached Data Supported by TDC

In the feature of Result Caching for Oracle Tuxedo Services, Oracle Tuxedo supports two methods to delete all the cached data related to a service.

When an Oracle Tuxedo application server is shutdown, Oracle Tuxedo deletes all the services in the server. If one service does not exist in the whole Oracle Tuxedo application system, Oracle Tuxedo deletes all the cached data which is based on the caching entry of the service. Now, Oracle Tuxedo only supports the "`tmshutdown -s servername`" command to trigger this function.

When shutdown Oracle Tuxedo Caching Server Module, Oracle Tuxedo tries to delete all the service cached data from all the caches which are configured in TDC property file, and tries to remove all the entries in cache with a key prefix `tdc_svc_` (a prefix of the internal caching key).

Configuring for Propagating Execution Context ID (ECID) to Oracle Coherence

Oracle Coherence can use the Execution Context ID (ECID) in its logs. This globally unique ID can be attached to requests between Oracle components. ECID allows you to track log messages pertaining to the same request when multiple requests are processed in parallel. Oracle Coherence logs will include ECID only if you already have an activated ECID prior to calling Oracle Coherence operations. ECID may be passed from another component or obtained in the client code.

Working as a client of Oracle Coherence, Oracle Tuxedo TDC enables you to propagate ECID to Oracle Coherence.

Enabling ECID

- Enable ECID in Oracle Tuxedo

  Oracle Tuxedo has two flags which you can add to OPTIONS in UBBCONFIG to control ECID.

  - ECID_CREATE

    Indicates that the ECID (Execution Context Identifier) creation function is enabled. In this case, boundary nodes (including Native/WS/Jolt clients and domain gateways) can generate the ECID.

  - ECID_USERLOG

If the identifier ECID_USERLOG is set and the ECID is not a null string, ECID will be appended to the userlog.

- Enable ECID in Oracle Coherence

  See *Oracle Coherence documentation* for instructions.

Enabling ECID for TDC

Prepare `tangosol-coherence-override.xml` in `${APPDIR}/config`. More specifically,

- `<destination>` element is used to configure path and file name for emitting log messages to a file. The specified path must already exist.
- Add ecid into `< message-format >` element to enable ECID.
- `<severity-level>` element can be used to change log level.

Listing 2-12 Enabling ECID for TDC

```
<?xml version='1.0'?>

<coherence  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"


xmlns="http://xmlns.oracle.com/coherence/coherence-operational-config"


xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-operationa
l-config coherence-operational-config.xsd">

<cluster-config>

       <member-identity>

       <cluster-name
system-property="tangosol.coherence.cluster">coherence_tux</cluster-name>

       </member-identity>

       <unicast-listener>

       <address
system-property="tangosol.coherence.localhost">localhost</address>

       <port system-property="tangosol.coherence.localport">51687</port>
```

```
        </unicast-listener>
        <multicast-listener>
        <port system-property="tangosol.coherence.clusterport">51697</port>
        </multicast-listener>
</cluster-config>

<logging-config>
        <destination
system-property="tangosol.coherence.log">/tmp/coherence.log</destination>
        <severity-level
system-property="tangosol.coherence.log.level">9</severity-level>
        <message-format>{date}/{uptime} {product} ecid={ecid} {version}
&lt;{level}&gt;(thread={thread},member={member}):{text}</message-format>
</logging-config>
</coherence>
```

Oracle Tuxedo Distributed Caching (TDC) Programming

This topic contains the following sections:

- Use Oracle Tuxedo Distributed Caching (TDC) Related ATMI APIs
- Put An Oracle Tuxedo Buffer to Cache
- Get An Oracle Tuxedo Buffer from Cache
- Remove Oracle Tuxedo Buffer(s) from Cache

Use Oracle Tuxedo Distributed Caching (TDC) Related ATMI APIs

Table 3-1 Oracle Tuxedo Distributed Caching (TDC) Related ATMI APIs

Name	Description
`tpgetcache(3c)`	Get an Oracle Tuxedo Cache handle according to the configuration
`tpcacheput(3c)`	Put an Oracle Tuxedo typed buffer into a cache, associating that buffer with a key
`tpcacheget(3c)`	Get the Oracle Tuxedo typed buffer associated with the key from a cache
`tpcacheremove(3c)`	Remove the cache entry associated with the parameter key from a cache

Table 3-1 Oracle Tuxedo Distributed Caching (TDC) Related ATMI APIs

Name	Description
`tpcachemremove(3c)`	Remove cache entries associated with the parameter keyarray from a cache
`tpcacheremoveall(3c)`	Remove all cache entries from a cache

For more information about the above APIs, see *Section 3c - C Functions*.

Put An Oracle Tuxedo Buffer to Cache

TDC APIs `tpgetcache` and `tpcacheput` can be used in an Oracle Tuxedo client or server to put an Oracle Tuxedo buffer associated with a key into an Oracle Tuxedo cache.

For more information about `tpgetcache` and `tpcacheput`, see Use Oracle Tuxedo Distributed Caching (TDC) Related ATMI APIs.

Listing 3-1 Put An Oracle Tuxedo Buffer Associated with A Key into An Oracle Tuxedo Cache

```
...
        TCACHE* mycache = NULL;

        char mykey[128];

        char* databuf = NULL;


        tpinit(NULL);

        databuf = tpalloc("STRING", NULL, 256);


        mycache = tpgetcache("tc");

        strcpy(mykey, "myname");

        strcpy(databuf, "scott");

        tpcacheput(mycache, mykey, databuf, 0, 0L);
        tpfree(databuf);
...
```

Get An Oracle Tuxedo Buffer from Cache

TDC APIs `tpgetcache` and `tpcacheget` can be used in an Oracle Tuxedo client or server to get an Oracle Tuxedo typed buffer from an Oracle Tuxedo cache according to a key.

For more information about `tpgetcache` and `tpcacheget`, see Use Oracle Tuxedo Distributed Caching (TDC) Related ATMI APIs.

Listing 3-2 Get An Oracle Tuxedo Buffer from an Oracle Tuxedo Cache According to the Key

```
...
        TCACHE* mycache = NULL;
        char mykey[128];
        char* databuf = NULL;

        tpinit(NULL);
        databuf = tpalloc("STRING", NULL, 256);

        mycache = tpgetcache("tc");
        strcpy(mykey, "myname");
        tpcacheget(mycache, mykey, &databuf, NULL, 0L);

        tpfree(databuf);
...
```

Remove Oracle Tuxedo Buffer(s) from Cache

TDC APIs `tpgetcache` and `tpcacheremove` can be used in an Oracle Tuxedo client or server to remove an Oracle Tuxedo buffer associated with a key from an Oracle Tuxedo cache. See Listing 3-3 for an example.

Also, you can use TDC API `tpcachemremove` to remove several Oracle Tuxedo buffers at one time, or use TDC API `tpcacheremoveall` to remove all the Oracle Tuxedo buffers from an Oracle Tuxedo cache. For more information about these TDC APIs, see Use Oracle Tuxedo Distributed Caching (TDC) Related ATMI APIs.

Listing 3-3 Remove An Oracle Tuxedo Buffer Associated with A Key from An Oracle Tuxedo Cache

```
...
TCACHE* mycache = NULL; char mykey[128];
char* databuf = NULL;


tpinit(NULL);
databuf = tpalloc("STRING", NULL, 256);


mycache = tpgetcache("tc"); strcpy(mykey, "myname");
tpcacheremove(mycache, mykey,0L);


tpfree(databuf);
...
```

Oracle Tuxedo Distributed Caching (TDC) Samples

This topic contains the following samples:

- Sample: Using Data Caching for Clients and Servers
- Sample: Using Result Caching for Oracle Tuxedo Services

Sample: Using Data Caching for Clients and Servers

Suppose `${APPDIR}` is `/home/scott/tuxedo/dom1`.

- Sample: Configure Oracle Coherence
- Sample: Start Oracle Coherence Cluster
- Sample: Configure Oracle Tuxedo Caching Server Module
- Sample: Configure TMJAVASVR in UBBCONFIG
- Sample: Put An Oracle Tuxedo Buffer to Cache
- Sample: Get An Oracle Tuxedo Buffer from Cache
- Sample: Remove An Oracle Tuxedo Buffer from Cache

Sample: Configure Oracle Coherence

- Prepare `tangosol-coherence-override.xml` in `${APPDIR}/config`. See Listing 4-1.

Configure Oracle Coherence cluster `coherence_tux` whose multicast port number is `51697` and unicast port number is `51687`.

- Prepare `coherence-cache-config.xml` in `${APPDIR}/config`. See Listing 4-2.

  Configure Oracle Coherence cache `tux_distributed`.

Listing 4-1 Prepare tangosol-coherence-override.xml

```
<?xml version='1.0'?>

<coherence  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"


xmlns="http://xmlns.oracle.com/coherence/coherence-operational-config"


xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-operational-co
nfig coherence-operational-config.xsd">

  <cluster-config>

    <member-identity>

      <cluster-name
system-property="tangosol.coherence.cluster">coherence_tux</cluster-name>

    </member-identity>

    <unicast-listener>

      <address
system-property="tangosol.coherence.localhost">localhost</address>

      <port system-property="tangosol.coherence.localport">51687</port>

    </unicast-listener>

    <multicast-listener>

        <port system-property="tangosol.coherence.clusterport">51697</port>

    </multicast-listener>

  </cluster-config>

</coherence>
```

Listing 4-2 Prepare coherence-cache-config.xml

```
<?xml version="1.0"?>

<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

              xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"


xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-cache-config
coherence-cache-config.xsd">

  <caching-scheme-mapping>

    <cache-mapping>

      <cache-name>tux_distributed</cache-name>

      <scheme-name>distributed</scheme-name>

    </cache-mapping>

  </caching-scheme-mapping>


  <caching-schemes>

    <distributed-scheme>

      <scheme-name>distributed</scheme-name>

      <service-name>DistributedCache</service-name>

        <lease-granularity>member</lease-granularity>

      <backing-map-scheme>

        <local-scheme/>

      </backing-map-scheme>

      <autostart>true</autostart>

    </distributed-scheme>

  </caching-schemes>

</cache-config>
```

Sample: Start Oracle Coherence Cluster

If no Oracle Coherence cluster is running, you can start your own cluster.

```
java -server -showversion $JAVA_OPTS -Dtangosol.coherence.mode=prod -cp
${APPDIR}/config: ${COHERENCE_HOME}/lib/coherence.jar
com.tangosol.net.DefaultCacheServer
```

Sample: Configure Oracle Tuxedo Caching Server Module

Preparing tdcsvr_coh.xml for Oracle Tuxedo Caching Server Module

Preparing `tdcsvr_coh.xml` for Oracle Tuxedo Caching Server Module in `${APPDIR}/config`.

See Listing 4-3, where the `<server-clopt>-f /home/scott/tuxedo/dom1/config/tdcsvr_coh.conf</server-clopt>` property specifies the TDC property file.

Listing 4-3 Configure Oracle Tuxedo Caching Server Module in Oracle Tuxedo Java Server Configuration File

```
<?xml version="1.0" encoding="UTF-8"?>

<TJSconfig version="2.0">

    <java-config>

        <jvm-options>-XX:MaxPermSize=192m</jvm-options>

        <jvm-options>-server</jvm-options>

<jvm-options>-Dtangosol.coherence.distributed.localstorage=false</jvm-options>

        <jvm-options>-Dtangosol.coherence.mode=prod</jvm-options>

    </java-config>

<tux-config>

       <server-clopt>-f
/home/scott/tuxedo/dom1/config/tdcsvr_coh.conf</server-clopt>

    </tux-config>

    <classpath-config>

    </classpath-config>
    <tux-resources>
    </tux-resources>
    <jdbc-resources>
    </jdbc-resources>
    <tux-server-config>
        <classpath>${TUXDIR}/udataobj/tuxj/tdc/com.oracle.tuxedo.tjtdc.jar
</classpath>
       <classpath>${TUXDIR}/udataobj/tuxj/tdc/dms.jar</classpath>
       <classpath>${TUXDIR}/udataobj/tuxj/tdc/ojdl.jar</classpath>
       <classpath>${APPDIR}/config</classpath>
       <classpath>${COHERENCE_HOME}/lib/coherence.jar</classpath>
       <server-class name="com.oracle.tuxedo.tdc.TCache4Coherence">
       </server-class>
    </tux-server-config>
</TJSconfig>
```

Note: The class path `${APPDIR}/config</classpath>` must be prior to the class path `${COHERENCE_HOME}/lib/coherence.jar`.

Prepare tdcsvr_coh.conf for TDC Property File

Prepare `tdcsvr_coh.conf` for TDC property file in `${APPDIR}/config`.

See Listing 4-4, where it configures Oracle Tuxedo cache `tc`, which actually uses Oracle Coherence cache `tux_distributed`.

Listing 4-4 Configure Oracle Tuxedo Distributed Caching (TDC) Property File

```
#* global option encoding setting
#options.encoding=no
```

```
#* configurations for Tuxedo cache "tc"


#* option encoding setting

cache.options.encoding.tc=no

#* physical cache used in Oracle Coherence

coh.cache.name.tc=tux_distributed
```

Sample: Configure TMJAVASVR in UBBCONFIG

Configure `TMJAVASVR` in `UBBCONFIG`.

See Listing 4-5, where multi-threaded configuration is enabled and `TMJAVASVR` configuration file `tdcsvr_coh.xml` is set.

Listing 4-5 Configure TMJAVASVR in UBBCONFIG

```
*RESOURCES

...

MODEL   SHM

...

*MACHINES

"m1"    LMID=L1

...

*GROUPS

JGRP1   LMID=L1  GRPNO=10


...

TMJAVASVR     SRVGRP=JGRP1  SRVID=10

              MINDISPATCHTHREADS=4  MAXDISPATCHTHREADS=4  MIN=2  MAX=2
```

```
            CLOPT="-- -c /home/scott/tuxedo/dom1/config/tdcsvr_coh.xml"
...
```

Sample: Put An Oracle Tuxedo Buffer to Cache

Put an Oracle Tuxedo `STRING` buffer associated with a key `mykey` into an Oracle Tuxedo cache `tc`.

Listing 4-6 Put an Oracle Tuxedo buffer

```
...
      TCACHE* mycache = NULL;
      char mykey[128];
      char* databuf = NULL;

      tpinit(NULL);
      databuf = tpalloc("STRING", NULL, 256);

      mycache = tpgetcache("tc");
      strcpy(mykey, "myname");
      strcpy(databuf, "scott");
      tpcacheput(mycache, mykey, databuf, 0, 0L);

      tpfree(databuf);
...
```

Sample: Get An Oracle Tuxedo Buffer from Cache

Get an Oracle Tuxedo typed buffer from an Oracle Tuxedo cache `tc` according to a key `mykey`.

Listing 4-7 Get an Oracle Tuxedo buffer

```
...
      TCACHE* mycache = NULL;
      char mykey[128];
      char* databuf = NULL;

      tpinit(NULL);
      databuf = tpalloc("STRING", NULL, 256);

      mycache = tpgetcache("tc");
      strcpy(mykey, "myname");
      tpcacheget(mycache, mykey, &databuf, NULL, 0L);

      tpfree(databuf);
...
```

Sample: Remove An Oracle Tuxedo Buffer from Cache

Remove an Oracle Tuxedo buffer associated with the key `mykey` from Oracle Tuxedo cache `tc`.

Listing 4-8 Remove An Oracle Tuxedo Buffer Associated with A Key from An Oracle Tuxedo Cache

```
...
TCACHE* mycache = NULL; char mykey[128];
char* databuf = NULL;

tpinit(NULL);
databuf = tpalloc("STRING", NULL, 256);
```

```
mycache = tpgetcache("tc"); strcpy(mykey, "myname");

tpcacheremove(mycache, mykey,0L);


tpfree(databuf);

...
```

Sample: Using Result Caching for Oracle Tuxedo Services

- Sample: Configure VIEWTABLE
- Sample: Configure UBBCONFIG
- Sample: Set on Server Side
- Sample: Set on Client Side

Sample: Configure VIEWTABLE

Configure your `VIEWTABLE`. See Listing 4-9, where the buffer type is `VIEW` and the subtype is `mystruct1`.

Listing 4-9 Configure VIEWTABLE

```
...

VIEW mystruct1

# type     cname     fbname                count  flag  size    null

string     name      -                      1      -     31      -

string     address   -                      1      -    255      -

char       age        -                     1     -     -        -

END

...
```

Sample: Configure UBBCONFIG

Configure your `UBBCONFIG`. See Listing 4-10, where

- `TMJAVASVR` is configured

  Multi-threaded configuration is enabled and the configuration file `tdcsvr_coh.xml` is set.

- Caching is enabled

  Oracle Tuxedo service `mysvc1` uses caching entry `svccache1` to improve performance. `svccache1` uses Oracle Tuxedo cache `tc1` to cache the service result. The corresponding key of the response is the value of the request data.

Listing 4-10 Configure UBBCONFIG

```
...
*GROUPS
JGRP1  LMID=L1 GRPNO=10


...
TMJAVASVR     SRVGRP=JGRP1 SRVID=10
               MINDISPATCHTHREADS=4 MAXDISPATCHTHREADS=4 MIN=2 MAX=2
               CLOPT="-- -c /home/scott/tuxedo/dom1/config/tdcsvr_coh.xml"
...
*SERVICES
mysvc1
       ...
       CACHING="svccache1"
...
*CACHING
```

```
svccache1
        CACHENAME="tc1"
        KEY=$request
...
```

Sample: Set on Server Side

Configure on server side. See Listing 4-11, where the request of `mysvc1` is set as `STRING` and the response of `mysvc1` is a `VIEW32 mystruct1`.

Listing 4-11 Configure on Server Side

```
...
struct mystruct1* rsp;
int tpsvrinit(int argc, char *argv[])
{
        rsp = tpalloc("VIEW32", "mystruct1", sizeof(struct mystruct1));
}
...
void mysvc1(TPSVCINFO *rqst)
{
        int ret = 0;
        /*rqst->data is the name, getrsp will get data from the database and
store into rsp*/
        ret = getrsp(rqst->data, rsp);
        if(ret < 0){
                tpreturn(TPFAIL, 0, NULL, 0L, 0);
        }
        tpreturn(TPSUCCESS, 0, rsp, 0L, 0);
}
...
```

Sample: Set on Client Side

Assume a data file is like this:

Listing 4-12 Data File Example

```
...
Scott
Mike
Andy
Scott
Ben
Brian
Scott
Clark
...
```

Set on your client set like Listing 4-13.

At the first time where `Scott` is taken as the request, `mysvc1` is invoked and the response is sent back and the response is cached into Oracle Tuxedo cache `tc1` with a key `Scott`. As long as the data in the cache `tc1` is not expired, all following requests for `Scott` to service `mysvc1` will get response from the cache `tc1` instead of invoking the service itself.

Listing 4-13 Set on Client Set

```
...
```

```
struct mystruct1* rsp;
char* req;
int main(int argc, char *argv[])
{
        int ret;
        long olen = 0;
        rsp = tpalloc("VIEW32", "mystruct1", sizeof(struct mystruct1));
        req = tpalloc("STRING", NULL, 32);
        /*get name from the data file*/
        while(getname(req) == 0){
                tpcall("mysvc1",req, 0, &rsp, &olen,0);
}
}
...
```