ADVANCED TUXEDO FOR PEOPLESOFT

CHAPTER 1

INTRODUCTION

Caveat

This document is still under construction. There is no guarantee that the content is error free, rather that there are certainly errors and inconsistencies!

If you find any errors, please tell author (e-mail address below)!

Acknowledgements

This document makes extensive use of the documentation supplied with the BEA software. A number of the diagrams and some text have been obtained from the BEA documentation.

Variables

Some of the directory references make use of environmental variables, which have been expressed in NT format.

NT Variable	Unix Variable	Description
%TUXDIR%	**$TUXDIR**	**Location under which BEA Tuxedo software installed.**
%PS_HOME%	**$PS_HOME**	**Location under which PeopleSoft software installed.**

David Kurtz, Go-Faster Consultancy Ltd.

(e-mail: david@go-faster.co.uk*, tel: +44-7771-760660)*

Tuxedo Web Manuals

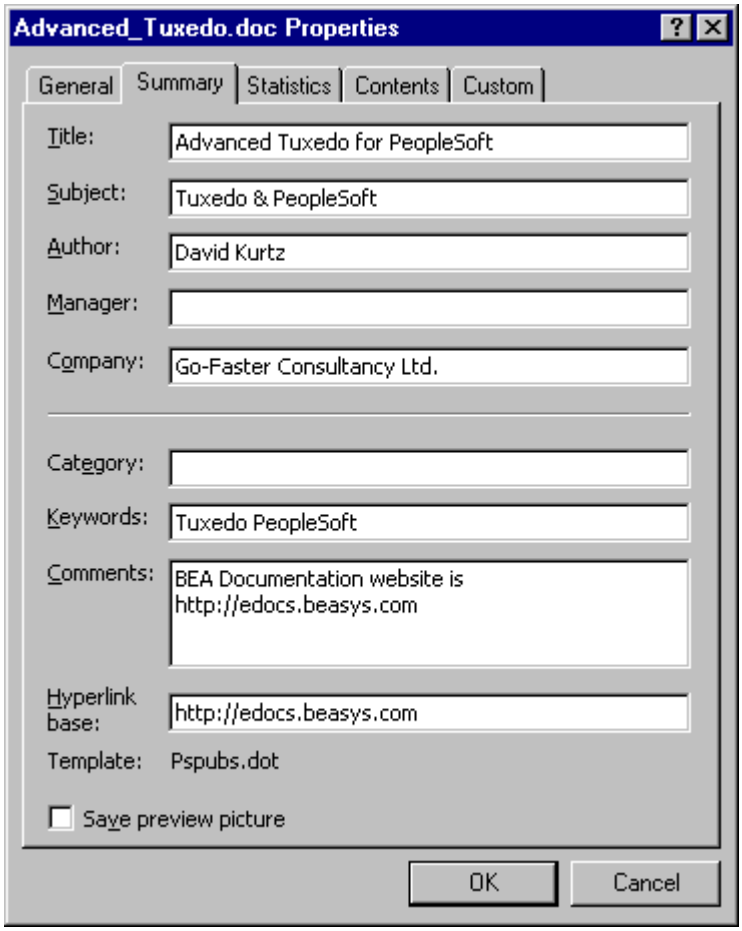
This document is intended as an addendum to any other BEA or PeopleSoft documentation regarding the application server, including PeopleSoft's 'Three-Tier Answer Book'.

BEA delivers Tuxedo and Jolt with very detailed on-line manuals that can be made available via a web server. Manuals for all Tuxedo products are also available on the public Internet on the BEA web site http://e-docs.beasys.com.

Product	File System Directory	Web Alias
Tuxedo 6.4	%TUXDIR%/doc	Tuxedo/tux64/welcome.htm
Tuxedo 6.5	%TUXDIR%/doc/Tuxedo/v6_5	tuxedo/tux65/index.htm
Jolt 1.2	%TUXDIR%/doc/jolt/v1_2	http://e-docs.beasys.com/tuxwle/jolt12/index.htm or tuxwle/jolt12/index.html

Wherever possible, references will be made in this document to them with the URL of the version on the BEA website. Where applicable, references to both the Tuxedo 6.4 and 6.5 manuals will be given.

If this document is opened in Microsoft Word or then the underlined web addresses become hot links, that will open the default browser. Most links are relative to edocs.beasys.com. If you wish to use a local web server then you may change the hyperlink base in File->Properties to match your local web server.

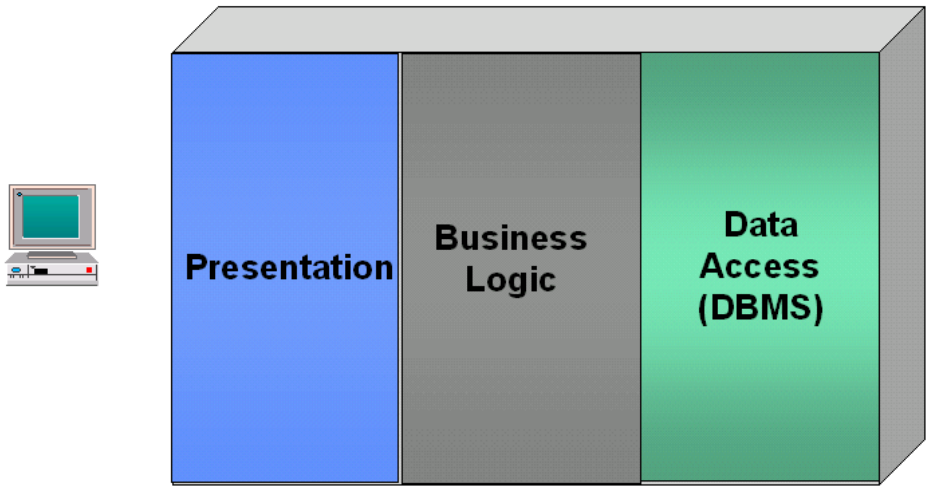


However, the virtual directory structure under the hyperlink base must match that expected by the document.

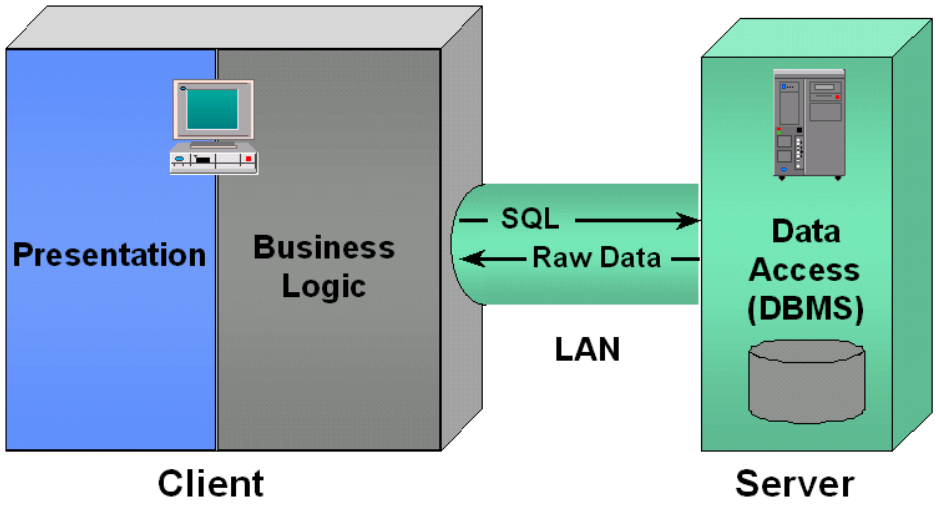
BACKGROUND

Historical Background

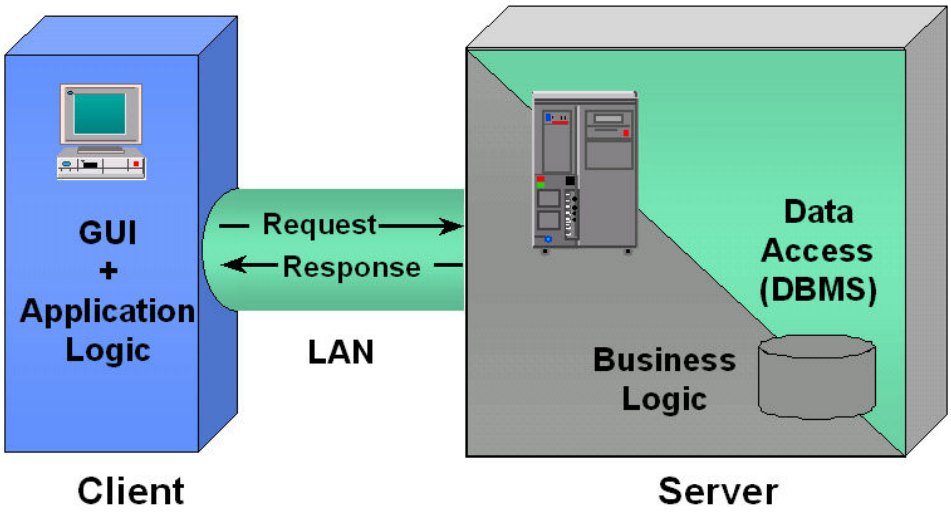
The original computing model is the mainframe. A dumb terminal was connected to a central, monlithic computer.



Everything happened on the central processor. The terminal was nothing more than a device that displayed the characters it was sent, and sent the keystrokes back. The interfaced was enevitably purely textual.



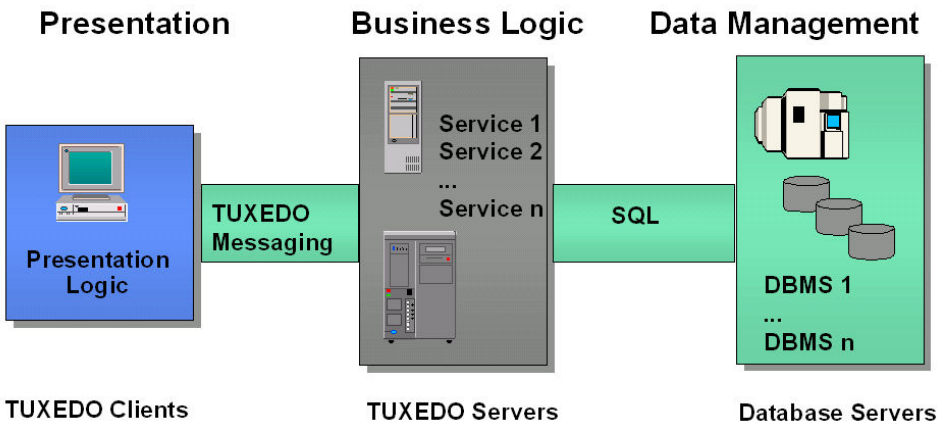
With the PC came client server. Everything moved to the end users desktop computer, except the data. It didn't take all that long for the strain to show in this model. The problems of resource hungry client applications, and software delivery are still with us. The other main issue was the load placed on the LAN by the SQL traffic. This diagram describes the situation up to PeopleTools 6.x (Tuxedo was present in version 6, but only for remote call, which had limited application).



Some of the application logic has migrated back to the server. The demands on client and network have driven this move.

This form of client server was very good and economical for small to medium sized applications, but there were drawbacks:

- the logic in the client was difficult to maintain,
- the shipping of data from client to server created large volumes of network traffic



Then came three-tier, the web is just a variation on the same theme. The database was restricted to doing what is does best, storing and retrieving data. The client does what it does best, the presentation of the application and the user interface. Most of the processing now takes place on the middle tier, the application server.

Where did Tuxedo come from?

Bell Laboratories developed Tuxedo in 1993. It provided an 'application framework for internal Unix based systems enabling construction, execution and administration of high volume business applications'.

They developed 'Transactions for Unix' (TUX). That later became 'Transactions for Unix Extended for Distributed Operations' (TUXEDO). A Tuxedo is what in International English is called a 'dinner jacket', but in America is only worn by a waiter, a person who serves in a restaurant. Tuxedo is a process that serves the client! This corny acronym has spawned a number of equally corny icons (see Icons page 86).

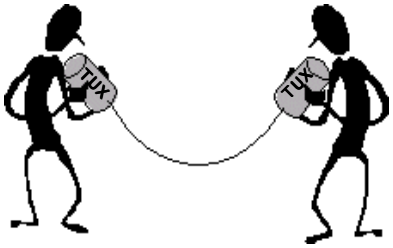
CHAPTER 2

TECHNICAL OVERVIEW

What is Tuxedo?

Tuxedo is often called middleware, it is also referred to as a messaging protocol. In PeopleTools stands between the Client and the database. It passes messages between the client and the database. It also performs some of the processing that is done on the client in two-tier operation. The client connects to Tuxedo and Tuxedo connects to the database.

Conceptually, the PeopleTools client in three-tier mode is the same in the two-tier. However, the client has been cut in half. The front half consists of the presentation layer and some of the application logic processing that is still on the client. The back half has the rest of the application logic layer, which makes the connection to the database. The two halves are reconnected by Tuxedo.

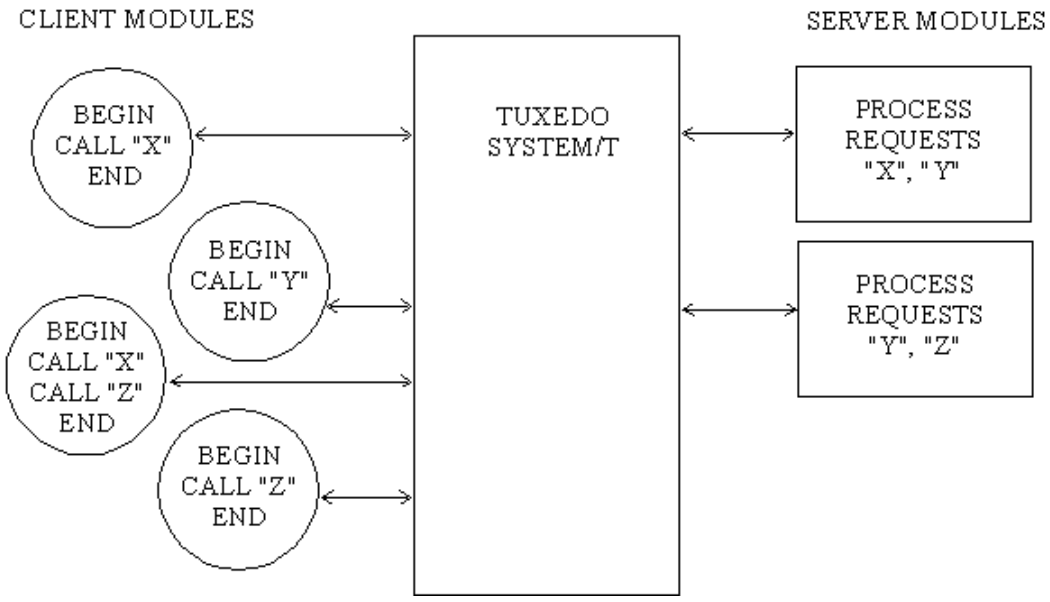


Conceptually it is the cans on each end of the string in the above cartoon.

It is made up of a number of server processes that communicate with the client via shared memory segments and queues.

Servers are software modules responsible for processing requests and sending replies back to clients.

Services are application-specific code within servers that process clients' requests. Often, as in the case with PeopleSoft, they process calls for accessing a database



high-level view of client-server architecture

Tuxedo has the ability to spawn addition copies of server processes in response to demand, to assign priorities to client requests, and allow parallel execution of service requests, thus increasing throughput,

It is important to distinguish between which parts of this picture are BEA Tuxedo vanilla processes and which are PeopleSoft processes which call the Tuxedo library functions.

The Simple Application

A number of sample applications are shipped with Tuxedo. The simple application is a good illustration of how an application can be broken down in three-tier application. All the source files can be found in %TUXDIR%/apps/simpapp.

The application has only one function, to convert a string to upper case. There is a function TOUPPER, which has been placed in an application server. A service called TOUPPER has been defined and is called from a client (see configuration file on page 124).

The Simple Client

The simple Tuxedo client takes the command line parameters and sends them in a Tuxedo message to the application server, and waits for the result.

First the client must connect to the application server, with the tpinit() function.

```
/* Attach to System/T as a Client Process */
if (tpinit((TPINIT *) NULL) == -1) {
 (void) fprintf(stderr, "Tpinit failed\n");
 exit(1);
}
```

Some memory must be allocated to hold the messages sent and received.

```
sendlen = strlen(argv[1]);

/* Allocate STRING buffers for the request and the reply */
if((sendbuf = (char *) tpalloc("STRING", NULL, sendlen+1)) == NULL) {
 (void) fprintf(stderr,"Error allocating send buffer\n");
 tpterm();
 exit(1);
}

if((rcvbuf = (char *) tpalloc("STRING", NULL, sendlen+1)) == NULL) {
 (void) fprintf(stderr,"Error allocating receive buffer\n");
 tpfree(sendbuf);
 tpterm();
 exit(1);
}
```

This is done with the tpalloc() command.

```
/* Request the service TOUPPER, waiting for a reply */
ret = tpcall("TOUPPER", (char *)sendbuf, 0,
                (char **)&rcvbuf, &rcvlen, (long)0);
```

The parameters of the command are the name of the service in the application server, the address and lengths of the send and receive buffers, and an option parameter.

Finally the client disconnects from the application server and relinquishes the memory it allocated.

```
/* Free Buffers & Detach from System/T */
tpfree(sendbuf);
tpfree(rcvbuf);
tpterm();
return(0);
```

When the message is sent to a simple server that converts the string to upper case. The result is sent back to the client (see full code listing of Simple Client on page 121).

The Simple Server

The message from the client passed to the function as a memory structure.

```
…
struct tpsvcinfo {
#define XATMI_SERVICE_NAME_LENGTH  32
     char     name[XATMI_SERVICE_NAME_LENGTH];/* service name invoked */
     long     flags;          /* describes service attributes */
     char     *data;          /* pointer to data */
     long     len;            /* request data length */
     int      cd;             /* reserved for future use */
     long     appkey;         /* application authentication client key */
     CLIENTID cltid;          /* client identifier for originating client */
};
typedef struct tpsvcinfo TPSVCINFO;
…
```

from %TUXDIR%/include/atmi.h

The server end of this application is just a C function, whose name is the same as the Tuxedo Service. The function has a single parameter, which is the message sent from the client (see Full code listing of Simple Server on page 123).

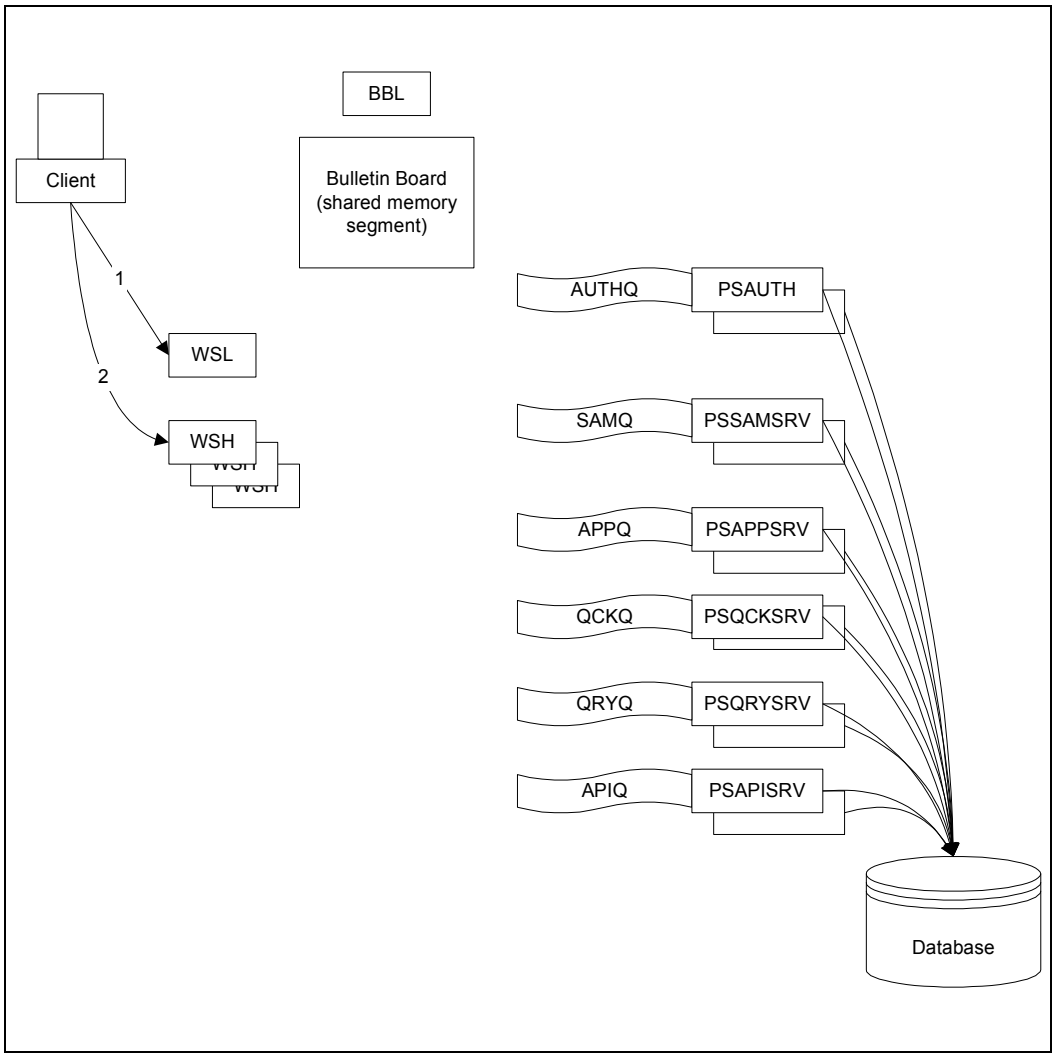
```
TOUPPER(rqst)
TPSVCINFO *rqst;
{

     int i;

     for(i = 0; i < rqst->len-1; i++)
      rqst->data[i] = toupper(rqst->data[i]);

     /* Return the transformed buffer to the requestor. */
     tpreturn(TPSUCCESS, 0, rqst->data, 0L, 0);
}
```

from %TUXDIR%/apps/simpapp/simpserv.c

So a function that could have been placed within the client program has been relocated in the server and is called as a service.

Processes, Memory & Messages

The Tuxedo application server domain is a collection of processes and shared memory segments and queues.



PeopleSoft/Tuxedo Internal Architecture

This diagram shows the key elements of a PeopleSoft application server domain. The picture can be separated into the vanilla BEA components, and the components built by PeopleSoft that make use of the infrastructure provided by Tuxedo.

The BEA jolt components have not been drawn but will also be discussed.

Tuxedo Components

The executable files for all the Tuxedo and Jolt processes described below can be found in %TUXDIR%/bin.

Bulletin Board (BB)

The Bulletin Board (BB) is the heart of the application server. It is a shared memory segment containing a collection of shared data structures designed to keep track of a running BEA TUXEDO system application. It contains information about servers, services, their loads and priorities, clients, and transactions pertaining to a BEA TUXEDO application. The bulletin board is replicated on each machine in the application.

Bulletin Board Liaison (BBL)

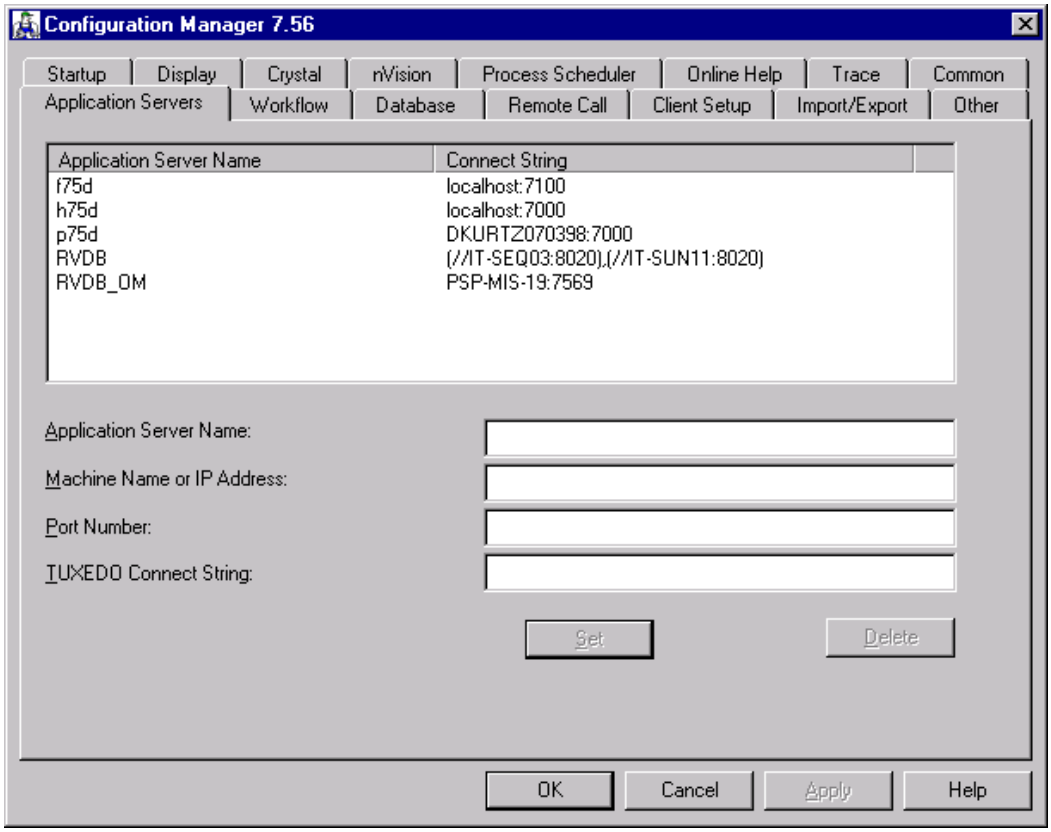
When the domain is booted by the *tmboot* utility, the Bulletin Board Liaison process is started. It reads the Tuxedo configuration file, *Pstuxcfg* and establishes and maintains the Bulletin Board on each machine in the application.

Workstation Listener (WSL) & Workstation Handler (WSH)

The WSL listens on a particular IP address and ports for requests for connection from clients.

The client is then assigned a port on a WSH. The WSL will if necessary, and if the configuration allows spawn an additional WSH process.

There can be multiple WSL as well as WSH processes in a single domain.



The PeopleSoft Client must be told where to find the WSLs, so in the configuration manager there is a Tuxedo connection string.

Note that the RVDB application server has two ports. These could be different ports on completely separate application servers on the same databases, or they could be different partitions, of the same application server, on different nodes.

Workstation Listener	
V6.4	tuxedo/tux64/sect5/wsl.htm
V6.5	tuxedo/tux65/refman/sect5/sect572.htm - 1023064

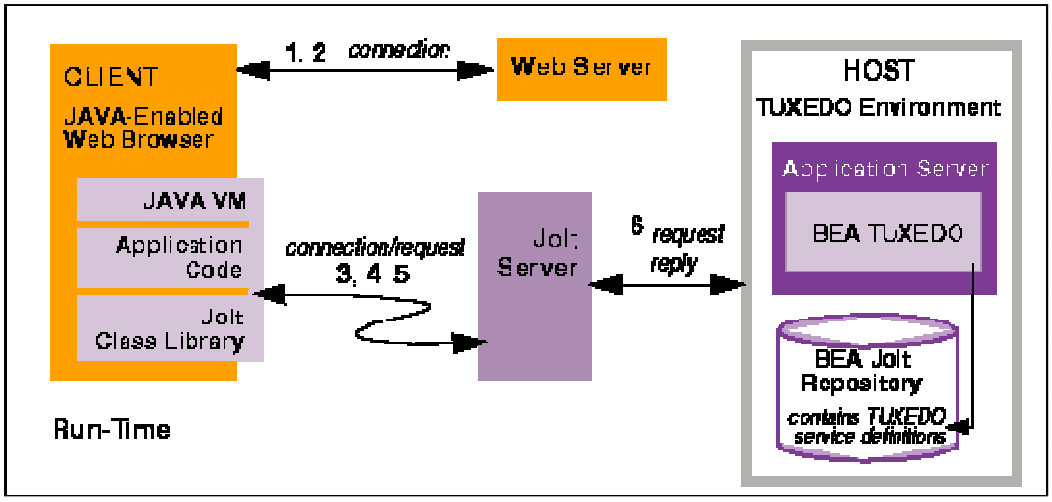
Tuxedo Glossary

The Tuxedo manual provides a glossary of terms.

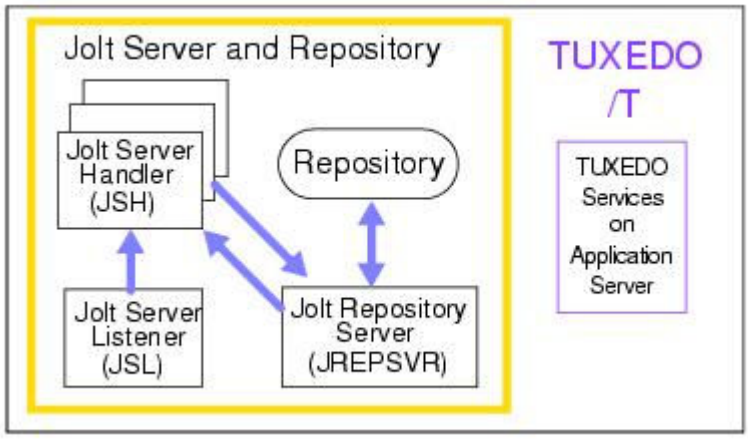
Tuxedo Manual: Glossary	
V6.4	tuxedo/tux64/fgl.htm
V6.5	tuxedo/tux65/glossary/index.htm

Jolt Components

Jolt is used by PeopleSoft to connect the Web client, which is written in Java, to the application server (see tuxwle/jolt12/html/joltdev/dvintro.html - 16377)



The requests from the Java client are passed through the Jolt Server into the Tuxedo domain. Thus, one application server can support both Web and Windows clients.



Jolt Server Listener (JSL) & Jolt Server Handlers

Jolt servers listen for network connections from clients, translate Jolt messages, multiplex multiple clients into a single process, and submit and retrieve requests to and from TUXEDO based applications running on one or more TUXEDO servers.

The JSL handles the initial Jolt client connection, and is responsible for spawning JSH processes, and assigning a Jolt client to the Jolt Server Handler

The JSH manages network connectivity, executes service requests on behalf of the client and translates TUXEDO buffer data into the Jolt buffer and vice versa.

These processes mirror the WSL and WSH processes, and have the same command line parameters.

Jolt Repository Server (JREPSRV)

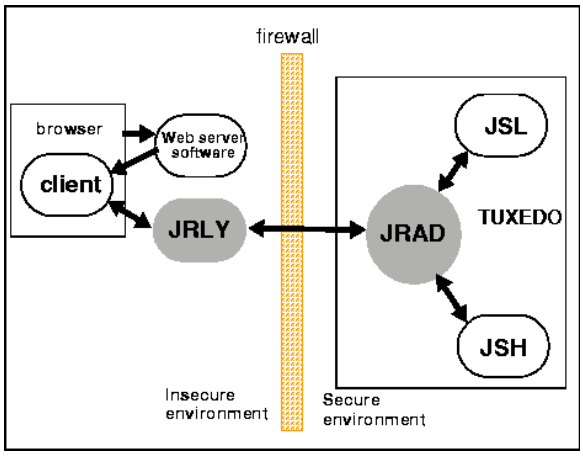
The Jolt Repository is a database where TUXEDO services are defined, such as name, number, type, parameter size, and permissions. The Repository functions as a central database of definitions for TUXEDO services and permits new and existing TUXEDO services to be made available to Jolt client applications.

All or only a few of these definitions may be exported to the Jolt Repository. Within the Jolt Repository, the developer or system administrator can export these services to the Jolt client application.

The JREPSVR retrieves Jolt service definitions from the Jolt Repository and returns the service definitions to the JSH. The JREPSVR also updates or adds Jolt service definitions.

Jolt Relay Adapter (JRAD)

JRAD is a TUXEDO application server, but does not include any TUXEDO services. It requires command line arguments to allow it to work with the JSH and TUXEDO.



Within PeopleTools, it is typically used when the web server for the Java client is on a different IP address to the JSL/JSH processes. It is a security restriction of Java that it can only connect to the server from which it was served up. Jolt Relay is used to redirect the connection from the web server to the application server.

The Jolt relay is transparent to Jolt clients and Jolt servers. A Jolt server can simultaneously connect clients directly to the Jolt Server (intranet clients) and via the Jolt Relay (Internet clients)

The JRLY (front-end relay) process may be started before or after the JRAD is started. If the JRAD is not available when the JRLY is started, the JRLY attempts to connect to the JRAD when it receives a client request. If JRLY is unable to connect to the JRAD, the client is denied access and is disconnected. A warning is written to the JRLY error log file.

JRAD receives client requests from JRLY, and forwards the request to the appropriate server. Replies from the server are forwarded back to JRAD, which sends the response and back to the requesting client. A single Jolt Internet Relay (JRLY/JRAD pair) handles multiple clients concurrently.

PeopleSoft Components

PeopleSoft has written various server processes. They each perform various functions. Each function, which has literally be been coded as a single 'C' function, corresponds to an activity within the client, such as a panel load, a panel save, executing a SQL query statement.

PeopleSoft delivers platform specific versions of the application servers for each platform. This partly explains the size of the PeopleTools patches

All the executables described below can be found in '*$PS_HOME/bin/server/<platform ID>/*'.

Where platform ID is one of *Aix_4_2*, *DIGITALUNIX_4_0*, *DYNIX_4_0*, *HPUX_10_20*, *HPUX_11_00*, *Ra3*, *RELIANT_5_43*, *SOLARIS_2_51*, *Vms_7_1*, *Winx86*.

This explains the size of PeopleTools patches, because compiled executables for 10 different platforms are delivered.

PSAUTH

The authorisation server is responsible for authenticating user logons.

PSAPPSRV

The 'Application Server' has 28 services and does the bulk of the work.

PSQCKSRV

The Quick Server is a copy of PSAPPSRV but used only for the 8 'quick' services that do not take long to execute.

PSQRYSRV

The Query Server has only one service *SQLRequest*. The service is aliases and advertised as *SQLQuery*. This is used to execute queries submitted by PS/Query, Crystal and nVision only.

This server was introduced from PeopleTools 7.05 and 7.54, thus moving the overhead of reporting away from the PSQCKSRV.

PSSAMSRV

This PSSAMSRV application server process is what Tuxedo terms a 'conversational server', which means that it is capable of conversing with other servers by submitting requests to them, or receiving requests from. It provides only one service, *SqlAccess*.

This server is used to administer the process scheduler tables, and to allocate version numbers on PeopleSoft objects during development and upgrade.

When a job is submitted, or when job and process scheduler statuses are updated in the process monitor, the SQL to perform those updates are submitted directly by PSSAMSRV.

PSAPISRV

This server drives the workflow e-mail activity. It serves only one service, *MsgAPI*.

Handling Messages

All communications from clients to the application server domain are in the form of messages requesting a service.

At the initial connection the client contacts the workstation listener (WSL) who then assigns it to a workstation handler (WSH). There after the client communicates with the WSH.

As each service request is received, it is the WSH process that determines where that message goes. For each type of server, a queue is defined. A queue can have many identical server processes taking requests from the front of the queue. It is recommended that not more that 10 processes be permitted on any one queue. If needs be more than one queue can supply the same type of server.

Services that can be serviced by server processes are advertised on the Bulletin Board. Thus by interrogating the Bulletin Board, the WSH can determine upon which queue to enqueue a particular service request.

A return message that will report success or failure, and which may contain data, will be sent back to the client via a return queue and the same WSH process.

Services & Servers

This table sets out the list of services within a PeopleSoft application server (at PeopleTools 7.56).

The application server domain is broken into a number of groups. These groups are a purely logical separation. Each group is assigned a group ID number by which it can also be referred to in *tmadmin* commands.

The servers exist within a group. Each server is given a server ID. The combination of group and server ID is unique. Where a queue can support many servers, the additional servers on that queue are given sequential server IDs following that of the ID specified. These IDs then correspond to slots in the tables in the Bulletin Board.

Only the PeopleSoft server processes advertise any services. The vanilla Tuxedo processes do not advertise any services. The Bulletin Board serves ‘internal’ services, and this can be observed from the ‘printserver’ command in tmadmin

The server processes handle the services. All the server processes on any one queue will be the same and will all handle the same services. Effectively, the services are therefore advertised on the queue.

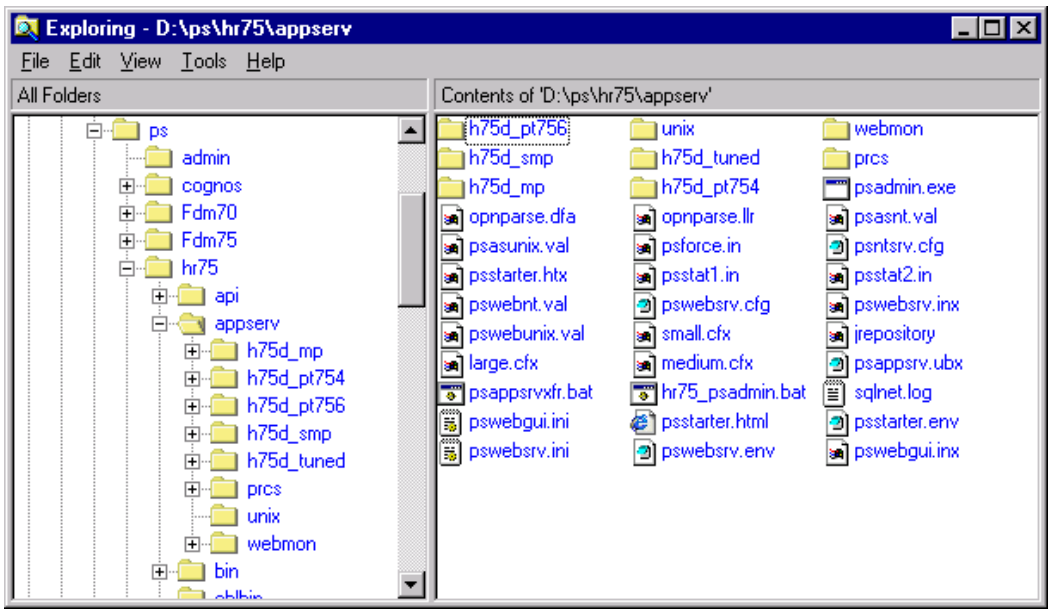
Group (Group ID)	Server (Server ID)	Service Request Name
BASE (1)	TMSYSEVT (200)	
	WSL (20)	
	PSAUTH (1)	GetCertificate
		PSAUTH
APPSRV (99)	PSAPPSRV (1) only	JavaMgrGetObj
		MenuList
		MgrGetObject
		MgrPeek
		MgrUpdObject
		OpnQryDescribe
		OpnQryExecute
		PprChangeValue
		PprDefault
		PprDeleteRow
		PprFieldChange
		PprFullSave
		PprInsertRow
		PprLoad
		PprMItemSelPC
		PprPrePopupPC
		PprSave
		PprSearchSave
		PprSearchStart
		PprSecStart
		PsdGetListSvc
		PsmSchedPrcs
		QdmGelListSvc
		RemoteCall
		StmGetExplain
	PSQCKSRV (50) & PSAPPSRV (1)	MgrClear
		RamList
		SamGetParmsSvc
		SqlRequest
		StmChgPswd
		StmGetTimeOut
		WamChgInstSvc
		WamStartInstSvc
	PSQRYSRV (70)	SqlQuery
	PSSAMSRV (100)	SqlAccess
	PSAPISRV (150)	MsgApi
JREPGRP (94)	JREPSRV (250)	
JSLGRP (95)	JSL (200)	

Configuration Files

The Bulletin Board Liaison (BBL) process reads a configuration file PSTUXCFG (see page 13). This section describes where that file is located and how it is generated.

Directory Structures

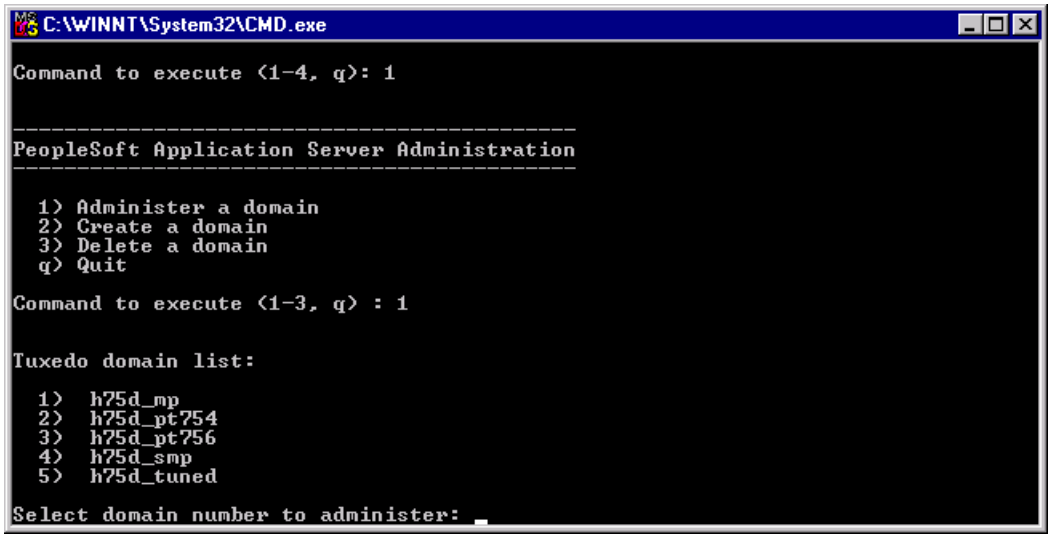
In a PeopleSoft, all the files for the application server are under $PS_HOME/appserv. This directory contains the template files that are then copied when you create a new domain.



The files for the new domain are created in a directory whose name is the same as the domain.

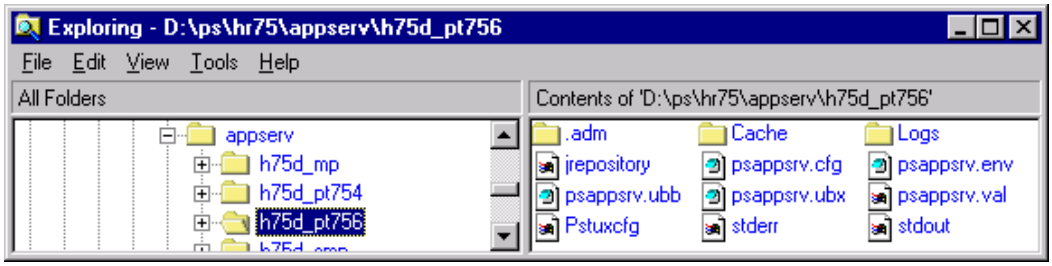
One file to notice here is *psadmin.exe*. This PeopleSoft utility is used to administer the application server domains, the process scheduler and the Tuxedo web server and listener and all relevant configuration files.

It provides a wrapper for a number of Tuxedo utilities and processes. It also sets certain environmental processes. %PS_HOME% must be set before starting *psadmin.exe*.



psadmin.exe

Application Server Domain Directory



Each application server domain directory will typically contain these files

psappsrv.cfg

PeopleSoft configuration file. This file is read by the PeopleSoft Application Server processes to obtain log-on and trace settings (see sample on page 89).

This text file is edited by the interactive configuration dialogue in the 'psadmin' utility, but can be edited manually. It is also possible to add extra variables.

psappsrv.val

This file is used to define what is a valid response during the interactive configuration dialogue. It is a plain text file, and it is possible to add additional validation (see sample on page 98).

psappsrv.ubx

This is a template file supplied by PeopleSoft. The variables embedded within it are found in *psappsrv.cfg* (see sample on page 100).

You may edit the *psappsrv.ubx* in order to change the Tuxedo configuration.

psappsrv.ubb

psappsrv.ubb is generated by the 'ubbgen' process. All the variables in the psappsrv.ubx file are resolved to a literal value in *psappsrv.ubb* (see sample on page 110).

'ubbgen' is a PeopleSoft process whose function is described in the comment section of *psappsrv.ubx*.

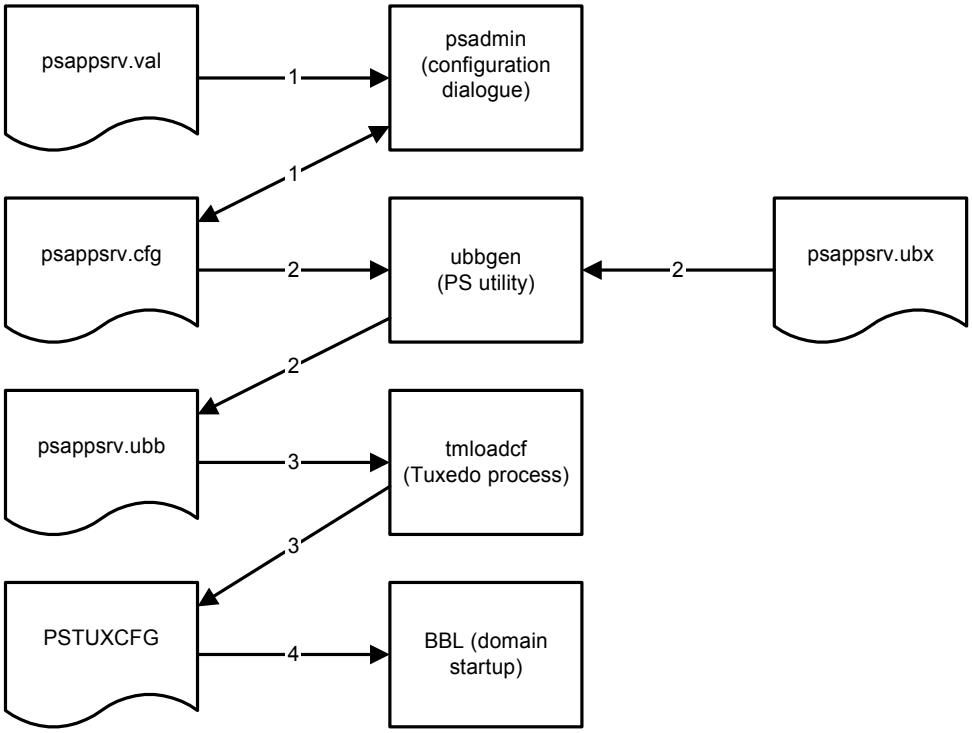
```
# USAGE:
#   ubbgen -t psappsrv.ubx -c psappsrv.cfg -o psappsrv.ubb -q y
#   ubbgen will open the file psappsrv.ubx, process it, and produce the file
#   psappsrv.ubb.
```

You should NEVER change *psappsrv.ubb* because any changes will not be picked up and will be overwritten next time the domain is reconfigured. However, it is often useful to examine this file to see what is currently set. Any changes should be made in *psappsrv.cfg* or *psappsrv.ubx*

Pstuxcfg

'Pstuxcfg' is a binary file generated by the Tuxedo *tmloadcf* utility from *psappsrv.ubb*. The 'BBL' process reads 'Pstuxcfg' when the Tuxedo domain is booted.

Configuration Process



This diagram illustrates the flow of information.

1. The interactive configuration dialogue may be used to change values in *psappsrv.cfg*. The values entered will be validated against *psappsrv.val*.

2. *psappsrv.cfg* and p*sappsrv.ubx* are combined by *ubbgen* to produce *psappsrv.ubb*. The variables in *psappsrv.ubx* and resolved to the literal values supplied in *psappsrv.cfg*.

3. *psappsrv.ubb* is read by *tmloadcf* which validates the configuration file and generates a compiled domain configuration file, *Pstuxcfg*.

4. On booting the domain the BBL process reads *Pstuxcfg* in order to determine the size of the Bulletin Board and configuration of the servers.

CHAPTER 3

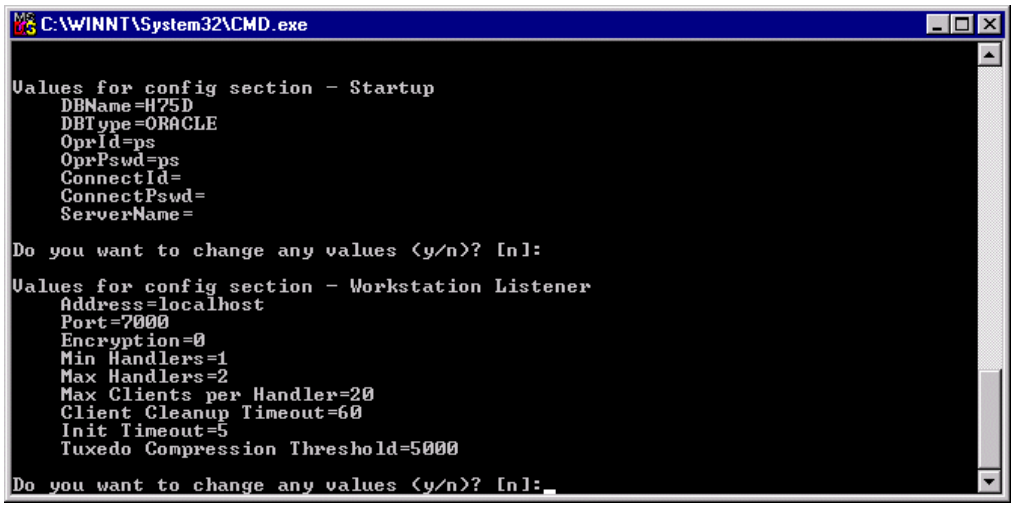
TUXEDO TUNING

In order to explain the Tuxedo configuration files and to discuss possible tuning opportunities this section will go through the *psappsrv.cfg*, *psappsrv.ubx* and *psappsrv.ubb* files.

The files exhibited in this chapter have be modified to illustrate what is possible

psappsrv.cfg

This file is contains settings that are normally user configurable. The file is maintained via the interactive configuration dialogue, but the file can be edited directly.



psadmin.exe

```
 [Startup]
;============================================================================
; Database Signon settings
;============================================================================
DBName=H75D
DBType=ORACLE
OprId=ps
OprPswd=ps
ConnectId=
ConnectPswd=
ServerName=


[Workstation Listener]
;============================================================================
; Settings for Workstation Listener
;============================================================================
;Address Note: Can be either Machine Name or IP address.
;Address Note: %PS_MACH% will be replaced with THIS machine's name
;Address=%PS_MACH%
Address=localhost
Port=7000
```

The Workstation listener listens to incoming connections on a single particular network address.

TCP/IP addresses may be specified in the following forms:

```
"//host.name:port_number"
```

The domain finds an address for *hostname* using the local name resolution facilities (usually DNS). *hostname* must be the local machine, and the local name resolution facilities must unambiguously resolve *hostname* to the address of the local machine.

```
"//#.#.#.#:PORT_NUMBER"
```

The "#.#.#.#" is in dotted decimal format. In dotted decimal format, each # should be a number from 0 to 255. This dotted decimal number represents the IP address of the local machine.

In both of the above formats, *port_number* is the TCP port number at which the domain process will listen for incoming requests. *port_number* can either be a number between 0 and 65535 or a name. If *port_number* is a name, then it must be found in the network services database on your local machine.

The address can also be specified in hexadecimal format when preceded by the characters "0x". Each character after the initial "0x" is a number between 0 and 9 or a letter between A and F (case insensitive). The hexadecimal format is useful for arbitrary binary network addresses.

```
Encryption=0
```

When establishing a network link between a Workstation client and the Workstation Handler, require at least this level of encryption. 0 means no encryption, while 40 and 128 specify the length (in bits) of the encryption key. If this level of encryption cannot be met, then the link will

fail to establish. The default value is 0.

```
Min Handlers=3
Max Handlers=5
Max Clients per Handler=20
```

Max Clients per Handler is passed as the multiplexing factor to the WSL. It controls the degree of multiplexing desired within each workstation handler. The value for this parameter indicates the maximum number of workstation clients that can be supported simultaneously by each workstation handler. The workstation listener ensures that new handlers are started as necessary to handle new workstation clients. This value must be greater than or equal to 1 and less than or equal to 4096.

The value in the PeopleSoft delivered configuration file is 60, but it is recommended that it be reduced to no more than 20, and perhaps lower. This means that the workstation handler is more responsive to client requests. The value must be greater than or equal to 1 and less than or equal to 4096. If this parameter were not specified to the workstation listener, the Tuxedo default would be 10.

The number of handlers should be increased by the same factor. The maximum number of clients than can connect via any one WSL is the product of the maximum number of handlers and the maximum number of clients per handler.

The minimum number of handlers should be set to cater for the average number of clients that will be connected. This will minimise the amount of spawn of WSH processes.

The product of 'Max Handlers' and 'Max Clients per Handler' gives the maximum number of concurrently connected clients that this workstation listener can support. In this example 5 x 20 = 100. Initially 3 handlers have been started so 60 clients can be supported without having to spawn additional WSH processes.

```
Client Cleanup Timeout=60
```

Client-timeout is the time in minutes allowed for a client to stay idle. If a client does not make any requests within this time period, the WSH disconnects the client. The option can be used for client platforms that are unstable (for example, where a user might turn off a personal computer without disconnect properly). Note that the option also affects clients that get unsolicited message notifications and do not follow up on them. If the argument were not supplied to WSL, there would be no timeout.

```
Init Timeout=5
```

This is the time, in seconds, that should be allowed for a workstation client to complete initialisation processing through the WSH before being timed out by the WSL. The Tuxedo default value for this parameter is 60. The legal range is between 1 and 32,767. If the authentication service takes longer than this value, then a time-out may result.

```
Tuxedo Compression Threshold=5000
```

This option determines the compression threshold to be used by workstation clients and handlers. Any buffers sent between workstation clients and handlers will be compressed if they are larger than the given value. The Tuxedo default value for this parameter is 2,147,483,647, which means

no compression is done since the legal range is between 0 and 2,147,483,647.

In a good LAN environment there will be only any improvement in transmission time will be exceeded by the time taken to compress and decompress the message. However, in a WAN or poorly performing LAN environment compression will significantly improve transmission times.

It would be perfectly reasonable to specify more than one workstation listener in a single domain, with different compression thresholds, and attach LAN users to one, and WAN users to the other.

Tuxedo Manual Entry for WSL	
V6.4	tuxedo/tux64/sect5/wsl.htm
V6.5	tuxedo/tux65/refman/sect5/sect572.htm

```
[JOLT Listener]
;===========================================================================
; Settings for JOLT Listener
;===========================================================================
;Address Note: Can be either Machine Name or IP address.
;Address Note: %PS_MACH% will be replaced with THIS machine's name
Address=%PS_MACH%
Port=9000
Encryption=0
Min Handlers=1
Max Handlers=2
Max Clients per Handler=20
```

The Jolt Station Listener behaves in exactly the same way as the Tuxedo WSL. The parameters are the same, and the same recommendations still apply as to reducing Max Clients per Handler to no more than 20.

```
Client Cleanup Timeout=60
Init Timeout=5
Client Connection Mode=ANY
```

The parameters for the Jolt Server Listener are identical to those for the WSL.

```
[JOLT Relay Adapter]
;===========================================================================
; Settings for JOLT Relay Adapter (JRAD)
;===========================================================================
;Listener Address Note: Can be either Machine Name or IP address.
;Listener Address Note: %PS_MACH% will be replaced with THIS machine's name
Listener Address=localhost
Listener Port=9100
```

The Jolt Relay Adapter is only required if the web server that serves up the Java client is on a different node to the JSL and JSHs. It is a

```
[Domain Settings]
;===================================================================
; General settings for this Application Server.
;===================================================================

;-------------------------------------------------------------------
; TUXEDO Domain for this Application Server (8 characters or less).
;
Domain ID=H75D
Local Machine ID=simple
```

The Machine ID is an arbitrary string which is use to name a node with a Tuxedo domain. In a partitioned domain, where a single domain exists across more than one node, different partitions or nodes, within a domain must be given different names. The PeopleSoft configuration is delivered as a single node domain. It is perfectly possible to configure it as a partitioned domain, but the GSC will not support this configuration. A separate support contract directly with BEA would be required.

```
;---------------------------------------------------------------------
; Additional directories for the Application Server's PATH environment
; variable.  This should include the location of the database DLL's.
;
Add to PATH=d:\orant80\bin

;---------------------------------------------------------------------
; Mark servers restartable or not.
;
Restartable=Y

;---------------------------------------------------------------------
; Location of TUXEDO and PeopleSoft Application Server log files for this
; Application Server.
;
;Log Directory=%PS_SERVDIR%\LOGS

;---------------------------------------------------------------------
; Logging detail level
;
; Level     Type of information
; -----     -------------------
;  -100     - Suppress logging
;  -1       - Protocol, memory errors
;   0       - Status information
;   1       - General errors
;   2       - Warnings
;   3       - Tracing Level 1 (default)
;   4       - Tracing Level 2
;   5       - Tracing Level 3
LogFence=3

;---------------------------------------------------------------------
; IMPORTANT NOTE: THIS SETTINGS IS NOT APPLICABLE FOR THE NT PLATFORM!
;                 This setting is not used until the K build!
;
Character Set=latin1


[Trace]
;=====================================================================
; SQL and PeopleCode Trace flags
;=====================================================================

;---------------------------------------------------------------------
; SQL Tracing Bitfield
;
; Bit       Type of tracing
; ---       ---------------
; 1         - SQL statements
; 2         - SQL statement variables
; 4         - SQL connect, disconnect, commit and rollback
```

```
; 8         - Row Fetch (indicates that it occurred, not data)
; 16        - All other API calls except ssb
; 32        - Set Select Buffers (identifies the attributes of columns
;             to be selected).
; 64        - Database API specific calls
; 128       - COBOL statement timings
; 256       - Sybase Bind information
; 512       - Sybase Fetch information
; 4096      - Manager information
; 8192      - Message Agent information
TraceSql=0
TraceSqlMask=12319

;-------------------------------------------------------------------------
; PeopleCode Tracing Bitfield
;
; Bit       Type of tracing
; ---       ---------------
; 1         - Trace entire program
; 2         - List the program
; 4         - Show assignments to variables
; 8         - Show fetched values
; 16        - Show stack
; 64        - Trace start of programs
; 128       - Trace external function calls
; 256       - Trace internal function calls
; 512       - Show parameter values
; 1024      - Show function return value
; 2048      - Trace each statement in program
TracePC=0
TracePCMask=0

;-------------------------------------------------------------------------
; new variable to control whether Tuxedo load balancing trace is enabled
; only valid values are -r to enable trace or blank to disable it
TuxedoLoadTrace=-r
```

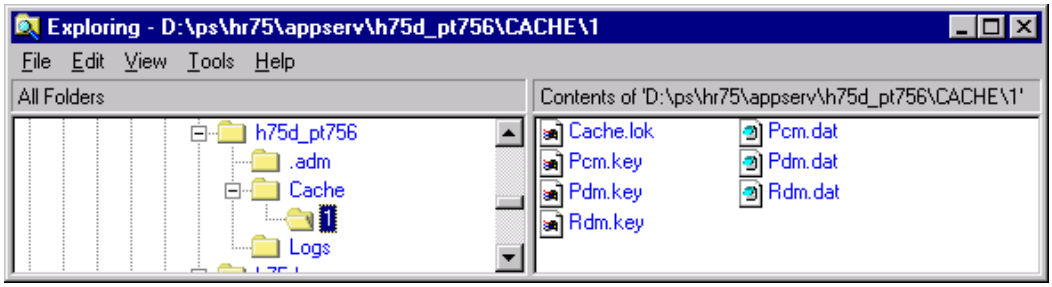
This is an addition variable not delivered by PeopleSoft. By adding the –r parameter to the command line option for a server specifies that it should record, on its standard error file, a log of services performed. This log may be analysed by the txrpt command.

Tuxedo Manual entry for Server Options	
V6.4	tuxedo/tux64/sect5/servopts.htm
V6.5	tuxedo/tux65/refman/sect5/sect535.htm

Tuxedo Manual entry for txrpt	
V6.4	tuxedo/tux64/sect1/txrpt.htm
V6.5	tuxedo/tux65/refman/sect1/sect146.htm

```
[Cache Settings]
;===================================================================
; Settings for managed object caching
;===================================================================

;-------------------------------------------------------------------
; Server caching settings
;
; EnableServerCaching = 1 to enable caching, 0 to disable it
EnableServerCaching=1
```

Just as the Windows client caches panel objects to a file system, so, from PeopleTools 7.5, does that PSAPPSRV process within the application server. Each PSAPPSRV process maintains a separate private cache in memory and on disk, if so configured. Each server in a domain is given a unique server ID, and that is used to name the cache directory in which cache files are placed.



The above example shows a domain with only one application server, whose server ID is 1, and hence there is only one cache directory.

```
; SwapBaseDir = the base cache directory
;SwapBaseDir=%PS_SERVDIR%\CACHE
```

The location of the cache directories is also configurable.

```
[Database Options]
;===================================================================
; Database-specific configuration options
;===================================================================

DB2InputMessageSize=
DB2OutputMessageSize=
SybasePacketSize=
; Please see Chapter "Tuning and Administration", in
; Oracle Installation and Administration Guide for details
UseLocalOracleDB=0
```

This is an Oracle specific parameter. If *UseLocalOracleDB* is set to 1, the connect string used by the application servers will not include the TNS Service name. This will cause them to make a 'Bequeath' or direct shared memory connection to the database. This will only work if the application server and database are on the same node, and the Oracle Server ID environment variable, ORACLE_SID, is set to the name of the database.

The same effect can be obtained by leaving this parameter set to the default value of 0, and

specifying the service in the TNSNAMES.ORA as a 'bequeath' service on the database server only.

Or Alternatively, by setting AUTOMATIC_IPC=ON in the SQLNET.ORA on the database server only.

```
EnableDBMonitoring=1
```

This parameter was introduced from PT7.53 and is currently only effective on Oracle. When set to 1 it causes the application server to register a string of information with the database. That string can be seen in V$SESSION.CLIENT_INFO. The string consists of the PeopleSoft Operator ID, the OS user name under which the application server process is running, and the name of the machine on which the application server process is running

For more information see PeopleBooks for PeopleTools 7.53 (*pt753eng.nfo*), or *PeopleSoft for the Oracle DBA*.

```
[RemoteCall]
;=====================================================================
; Settings for RemoteCall
;=====================================================================

;---------------------------------------------------------------------
; RemoteCall child process output redirection
;
; If this parameter is non-zero, the child process output is saved to
; <Domain Settings\Log Directory>\<program>_<oprid>.out, and any error
; output is saved to <program>_<oprid>.err.
; By default, the output is not saved
;
RCCBL Redirect=0

;---------------------------------------------------------------------
; Location of COBOL programs
; By default, RemoteCall looks for the COBOL programs in %PS_HOME%\cblbin
;
;RCCBL PRDBIN=%PS_HOME%\cblbin


[PSAUTH]
;=====================================================================
; Settings for PSAUTH
;=====================================================================

;---------------------------------------------------------------------
; UBBGEN settings
Min Instances=1
Max Instances=2
; Note: Spawn is not valid for this (MSSQ) server!
```

The spawn parameter is used by psadmin to determine whether to include the *spawn server* parameter in the server command line. Spawning is valid for the PSAUTH server, if a MSSQ

Queue is specified in *psappsrv.ubx*.

```
Spawn=0
```

If this variable is set to 1, and Max instances is greater than Min instances, PSADMIN sets a variable, Spawn Server to –p 1,600:1,1.

```
-p[L][low_water][,[terminate_time]][:[high_water][,create_time]]
```

This option can be used to support automatic spawning/decay of servers. It may be used for servers on an MSSQ with MAX greater than 1; it is not allowed (and not necessary) for conversational servers. Arguments to the option have the following meanings: L The decision to spawn more servers is based on load rather than number of servers or messages. -- The remaining arguments, 'low_water', 'terminate_time', 'high_water', and 'create_time' are used to control when servers are spawned or deactivated.

The algorithm is: if the load meets or exceeds 'high_water' for at least 'create_time' seconds, a new server is spawned. If the load drops below 'low_water' for at least 'terminate_time' seconds, a server is deactivated.

If not specified, 'low_water' defaults to an average of 1 server or message on the MSSQ or a workload of 50. 'high-water' defaults to an average of 2 servers or messages, or a workload of 100. 'create_time' defaults to 50: 'terminate_time' defaults to 60.

It makes no sense that the start-up and shutdown thresholds are the same. However, the Spawn parameter cannot be configured in PeopleTools 7.5x, its value is hard coded within 'psadmin'.

```
User Spawn=-p L50,600:100,60
```

Hence, a new variable 'User Spawn' has been specified which can be set to any value within the configuration dialogue.

```
Service Timeout=300

;------------------------------------------------------------------------
; Number of services after which PSAUTH will automatically restart.
; If the recycle count is set to zero, PSAUTH will never be recycled.
; The default value is zero.
Recycle Count=100000
```

The PeopleSoft servers will terminate are handling the number of request specified in the recycle count. They will then be respawned by Tuxedo. The uses for this are to relinquish memory claimed by the server to cache objects. The memory cache can be quickly rebuilt from the file system cache.

```
;------------------------------------------------------------------------
; Number of consecutive service failures after which PSAUTH will
; automatically restart.
; If this is set to zero, PSAUTH will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=0

;------------------------------------------------------------------------
; Use the database for additional signon authentication.
; The default value is zero.
Validate Signon with Database=0


[PSAPPSRV]
;========================================================================
; Settings for PSAPPSRV
;========================================================================

;------------------------------------------------------------------------
; UBBGEN settings
Min Instances=1
Max Instances=2
; Note: Spawn may only be 1 if Max Instances is greater than Min Instances!
Spawn=0
User Spawn=-p L50,600:100,60
Service Timeout=300

;------------------------------------------------------------------------
; Number of services after which PSAPPSRV will automatically restart.
; If the recycle count is set to zero, PSAPPSRV will never be recycled.
; The default value is zero.
Recycle Count=100000

;------------------------------------------------------------------------
; Number of consecutive service failures after which PSAPPSRV will
; automatically restart.
; If this is set to zero, PSAPPSRV will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=2
```

```
; Max Fetch Size -- max result set size in KB for a SELECT query
; Default is 5000KB. Use 0 for no limit.
Max Fetch Size=5000


[PSSAMSRV]
;===================================================================
; Settings for PSSAMSRV
;===================================================================

;-------------------------------------------------------------------
; UBBGEN settings
Min Instances=1
Max Instances=2
; Note: Spawn is not valid for this (conversational) server!
```

Conversational servers are automatically spawned as needed. When a TPCONNECT call is made to a service offered by that server, the system starts up a second copy. As each copy is called a new one is spawned, up to the limit imposed by *Max Instances*.

```
Service Timeout=300

;---------------------------------------------------------------------
; Number of services after which PSSAMSRV will automatically restart.
; If the recycle count is set to zero, PSSAMSRV will never be recycled.
; The default value is zero.
Recycle Count=100000

;---------------------------------------------------------------------
; Number of consecutive service failures after which PSSAMSRV will
; automatically restart.
; If this is set to zero, PSSAMSRV will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=2

; Max Fetch Size -- max result set size in KB for a SELECT query
; Default is 32KB. Use 0 for no limit
Max Fetch Size=32


[PSQCKSRV]
;=====================================================================
; Settings for PSQCKSRV
;=====================================================================

;---------------------------------------------------------------------
; UBBGEN settings
Min Instances=1
Max Instances=2
; Note: Spawn may only be 1 if Max Instances is greater than Min Instances!
Spawn=0
User Spawn=-p L50,600:100,60
Service Timeout=300

;---------------------------------------------------------------------
; Number of services after which PSQCKSRV will automatically restart.
; If the recycle count is set to zero, PSQCKSRV will never be recycled.
; The default value is zero.
Recycle Count=100000

;---------------------------------------------------------------------
; Number of consecutive service failures after which PSQCKSRV will
; automatically restart.
; If this is set to zero, PSQCKSRV will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=2

; Max Fetch Size -- max result set size in KB for a SELECT query
; Default is 5000KB. Use 0 for no limit.
Max Fetch Size=5000
```

```
[PSQRYSRV]
;===========================================================================
; Settings for PSQRYSRV
;===========================================================================

;---------------------------------------------------------------------------
; UBBGEN settings
Min Instances=1
Max Instances=2
; Note: Spawn may only be 1 if Max Instances is greater than Min Instances!
Spawn=0
User Spawn=-p L50,600:200,60
Service Timeout=300

;---------------------------------------------------------------------------
; Number of services after which PSQRYSRV will automatically restart.
; If the recycle count is set to zero, PSQRYSRV will never be recycled.
; The default value is zero.
Recycle Count=100000

;---------------------------------------------------------------------------
; Number of consecutive service failures after which PSQRYSRV will
; automatically restart.
; If this is set to zero, PSQRYSRV will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=2

; Max Fetch Size -- max result set size in KB for a SELECT query
; Default is 10000KB. Use 0 for no limit.
Max Fetch Size=10000


[PSAPISRV]
;===========================================================================
; Settings for PSAPISRV
;===========================================================================

;---------------------------------------------------------------------------
; UBBGEN settings
Min Instances=1
Max Instances=2
; Note: Spawn may only be 1 if Max Instances is greater than Min Instances!
Spawn=0
User Spawn=-p L50,600:100,60
Service Timeout=300

;---------------------------------------------------------------------------
; Number of services after which PSAPISRV will automatically restart.
; If the recycle count is set to zero, PSAPISRV will never be recycled.
; The default value is zero.
Recycle Count=100000
```

```
;-------------------------------------------------------------------------
; Number of consecutive service failures after which PSAPISRV will
; automatically restart.
; If this is set to zero, PSAPISRV will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=0


[SMTP Settings]
;=========================================================================
; Settings for SMTP mail
;=========================================================================

SMTPServer=
SMTPPort=25
SMTPServer1=
SMTPPort1=0
SMTPSender=PeopleSoft@peoplesoft.com
SMTPSourceMachine=
```

psappsrv.val

This file is used to validate the responses given during the interactive configuration dialogue to configure an application server. It was introduced from PeopleTools 7.5.

If psappsrv.cfg is edited directly then there will be no validation of the values.

```
#
# Default PeopleSoft Application Server validation file for the Windows NT
platform.
#
# Note: This is the REQUIRED format for this file!
#
#       Element={type}(criteria)
#       Three types are allowed: int, path and string.
#
#       Format for each is as follows:
#              Element={int}(range)
#              Element={path}(filename) Note: filename is relative to the path
entered or listed in the cfg file
#              Element={string, MAXIMUM length}(optional criteria list)
#
#       If type is string, there MUST be a MAXIMUM length associated as in:
{string,8}
#       This means that a string entered MUST be less than or equal to 8
characters!
#       There can also be a (criteria) list associated with a string type.
#       If type is other than string, there MUST be criteria.
#       There should be no extraneous characters or spaces!
#
```

```
#
[Startup]
DBName={string,8}
```

In any system there will be a limited number of databases to connect to, a list of valid database names could be encoded.

```
DBType={string,8}(DB2,DB2400,DB2ODBC,DB2UNIX,INFORMIX,MICROSFT,ORACLE,SQLBASE,SY
BASE)
```

Most customers only run one platform, so the list of valid platforms could be cut down.

```
OprId={string,8}
OprPswd={string,8}
#
[Workstation Listener]
Port={int}(1025-65536)
```

The range of valid WSL port values could be restricted to a tighter range if necessary.

```
Encryption={string,3}(0,40,128)
#
[JOLT Listener]
Client Connection Mode={string,9}(RETAINED,RECONNECT,ANY)
#
[Domain Settings]
Character Set={nodisplay}
#
```

All the settings above this point are vanilla validation rules supplied by PeopleSoft. All the settings below have been added.

```
#dmk - added trace section
[Trace]
TuxedoLoadTrace={string,2}(-r, )
```

The only valid values for the Tuxedo trace parameter are –r or a space if trace is not required.

```
#
# dmk - maximum of 10 servers on a queue
[PSAUTH]
Min Instances={int}(1-10)
Max Instances={int}(1-10)
```

BEA recommends that no more than 10 servers are configured on any one queue. Therefore, additional validation has been added for Min and Max instances of each server type.

```
#
[PSAPPSRV]
Min Instances={int}(1-10)
Max Instances={int}(1-10)
#
[PSSAMSRV]
Min Instances={int}(1-10)
Max Instances={int}(1-10)
#
[PSQRYSRV]
Min Instances={int}(1-10)
Max Instances={int}(1-10)
#
[PSAPI]
Min Instances={int}(1-10)
Max Instances={int}(1-10)
```

psappsrv.ubx

This file is the template for the Tuxedo configuration file *psappsrv.ubb*. The variables contained within are resolved from *psappsrv.cfg* by the *ubbgen* process.

```
################################################################################
# SourceSafe Information:
#
# $Logfile:: /PT75/APPSERV/psappsrv.ubx                                      $
# $Revision:: 80                                                             $
# $Author:: Lzhuang                                                          $
# $Date:: 2/15/99 10:37p                                                     $
################################################################################

*PS_DEFINES
# This section defines the variables used in the ubb file.  The ubb config
# file generation utility, ubbgen, processes this section and replaces them
# with their values in the rest of the file.  The resulting ubb file can
# be used to generate the tuxconfig file directly, or can be modified
# further with any text editor.
#
# USAGE:
#   ubbgen -t psappsrv.ubx -c appsrv.cfg -o psappsrv.ubb -q y
#   ubbgen will open the file psappsrv.ubx, process it, and produce the file
#   psappsrv.ubb.
#   The -q y option allows for command line processing when ubbgen is called
from psadmin. (Quiet mode)
#
# FILE FORMAT:
#   Substitution variables are enclosed in braces ('{', '}'), and
#   must be defined in the PS_DEFINE block (delimited with the *PS_DEFINES
#   and *END tokens).  There are four types of substitution variables:
#       - Environment variables.  The name of the variable starts with '$'.
#         ubbgen gets the value of these variables with getenv.
#         ($TUXCONFIG)
#       - Config file variables.  These variables are read from the
#         configuration file passed in to ubbgen (appsrv.cfg, by default).
#         The name of the variable starts with '$', followed by the name of
#         the sub-section in the registry, then a slash ('\'), and then the
#         name of the key.
#         ($Domain Settings\Application Server Home)
#       - Special variables.  ubbgen recognizes these variables by name, and
#         performs special handling to get their values.
#         (DOMAINID, IPCKEY, MACH, WSNADDR, UID GID)
#       - Prompted variables.  Any variable which does not fall into the
#         catergories above is assumed to be of this type.  ubbgen prompts
#         the user to specify the value of the variable.  The line of text
#         immediately following the variable definition is used as the prompt.
#         (APPDIR, TUXDIR)
#
```

```
#   Any text following the pound symbol ('#') is interpreted as a comment,
#   and is not searched for substitution variables.
#
#   The PS_DEFINES block is not copied into the output file.
```

The above comment section explains how *ubbgen* works.

```
{QUICKSRV} Move quick PSAPPSRV services into a second server (PSQCKSRV) (y/n)?
[y]:
{QUERYSRV} Move long-running SqlQuery service into a second server (PSQRYSRV)
(y/n)? [y]:
{JOLT} Do you want JOLT configured (y/n)? [n]:
{JRAD} Do you want JRAD configured (y/n)? [n]:

*END
```

The questions, and their default answers, asked during the configuration process are also defined in the *PSDEFINE section. The responses set the values of the logical variables defined in the curly brackets. The variables are then used like #DEFINE pre-processor directives in C to control which parts of *psappsrv.ubx* are parsed by *ubbgen*.

The text of the first question is highly misleading. If the answer is yes, the 8 quick PSAPPSRV services are not moved off PSAPPSRV, they are additionally advertised on PSQCKSRV and well as PSAPPSRV. Thus these 8 services are available on two different queues. This has an implication for how the load is distributed across the APPQ and QCKQ queues.

In PeopleTools 8 the configuration is different. If PSQCKSRV is configured, then the 8 quick services on PSQCKSRV are no longer advertised on PSAPPSRV.

The second question does mean what it says. If the response is positive, the SqlQuery service will no longer be advertised on PSAPPSRV or PSQCKSRV, but will advertised on PSQRYSRV instead.

From this point on, the psappsrv.ubx contains Tuxedo configuration settings.

Tuxedo Manual entry for ubb configuration specification	
V6.4	tuxedo/tux64/sect5/ubbcfg.htm
V6.5	tuxedo/tux65/refman/sect5/sect566.htm

```
##########################################################################
#
# This is a skeletal TUXEDO configuration file - "psappsrv.ubb" designed
# to be used for PeopleTools 7.5 app server and the Remote Call mechanism.
# To configure additional resources, machines, servers, services, etc.
# please refer to "ubbconfig" in section 5 of the TUXEDO System Reference
# Manual.
#
##########################################################################
```

The file is made up of up to none specification sections. Lines beginning with an asterisk (*) indicate the beginning of a specification section. Each such line contains the name of the section

immediately following the *.

```
*RESOURCES
IPCKEY          {IPCKEY}        # ( 32768 < IPCKEY < 262143 )
```

IPCKEY specifies the numeric key for the well-known address in a BEA TUXEDO system bulletin board. In a single processor environment, this key "names" the bulletin board. In a multiple processor environment, this key names the message queue of the DBBL. In addition, this key is used as a basis for deriving the names of resources other than the well-known address, such as the names for bulletin boards throughout a multiprocessor. IPCKEY must be greater than 32,768 and less than 262,143. This parameter is required, and it is generated by *psadmin.exe*.

```
MASTER          simple
DOMAINID        {$Domain Settings\Domain ID}
MODEL           SHM
```

Model specifies the configuration type. This parameter is required and only one of the two settings can be specified. SHM specifies a single machine configuration; only one machine may be specified in the MACHINES section. MP specifies a multi-machine configuration; MP must be specified if a networked application is being defined.

```
LDBAL           Y
```

LDBAL specifies whether or not load balancing should be performed. If *LDBAL* is not specified, the default is Y. It is recommended that if each service maps to one and only one queue, then *LDBAL* should be set to N, since load balancing is automatic.

By default, if PSQCKSRV is configured, in PeopleTools 7.x there is a choice of queue for the quick services. In PeopleTools 8 there is not.

```
#
MAXMACHINES     256             # min, default=256
```

MAXMACHINES specifies the maximum number of configured machines to be accommodated in the machine tables of the bulletin board. This value must greater than or equal to 256 and less than 8,191. If not specified, the default is 256.

```
MAXGROUPS       100             # min, default=100
```

MAXGROUPS specifies the maximum number of configured server groups to be accommodated in the group table of the bulletin board. This value must greater than or equal to 100 and less than 32,768. If not specified, the default is 100. A PeopleSoft application server only uses 4 groups.

```
#{MAXSERVERS}
MAXSERVERS    40
```

MAXSERVERS specifies the maximum number of servers to be accommodated in the server table of the bulletin board. This value must be greater than 0 and less than 8192. If not specified, the default is 50.

This must also be sufficient to cater for all the application server processes, the *BBL*, *DBBL*, *WSL*, *WSH*, *JSL*, and *JSH* processes that may be booted.

This value is calculated by *psadmin* but the value will not be calculated properly if multiple queues are configured for the same server process (see Calculation of other Tuxedo Settings, MAXSERVERS, page 69).

```
#{MAXSERVICES}
MAXSERVICES   200
```

MAXSERVICES specifies the maximum total number of services to be accommodated in the service table of the bulletin board. This value must be greater than 0 and less than 32,768. If not specified, the default is 100.

It must be large enough to accommodate all the services advertised by all the server processes. 28 services/*PSAPPSRV*, 8 services/*PSQCKSRV*, 2 services/*PSAUTH*, and 1 server per *PSSAMSRV*, *PSAPISRV* and *PSQRYSRV*.

Like MAXSERVERS is calculated by *psadmin* but again the value will not be calculated properly if multiple queues are configured for the same server process (see Calculation of other Tuxedo Settings, MAXSERVICES, page 70).

```
#
MAXACLGROUPS    1               # def=16K (incl only to save BB space)
MAXGTT          0               # def=100 (----------   " ----------)
MAXCONV         50              # def=10  (----------   " ----------)
#
SECURITY        USER_AUTH
```

SECURITY specifies the type of application security to be enforced. The possible string values are NONE, APP_PW, USER_AUTH, ACL, MANDATORY_ACL. This parameter defaults to NONE. The value USER_AUTH indicates that access control checks will be done on service names, queue names, and event names.

```
AUTHSVC         PSAUTH
```

The value of AUTHSVC, which is always PSAUTH in a PeopleSoft application server, specifies the name of an application authentication service that is invoked by the system for each client joining the system. This parameter requires that the SECURITY identifier be set to USER_AUTH, ACL, or MADATOR_ACL. The parameter value must be 15 characters or less in length. For SECURITY level USER_AUTH, the default service name, if not specified, is AUTHSVC.

```
SYSTEM_ACCESS   FASTPATH
#
PERM            0660            # can override on a per m/c basis
#
SCANUNIT        5               # Time in seconds between scans by the BBL
                                # for old transactions and timed-out blocking
                                # calls.

SANITYSCAN      2               # The BBL indicates to the DBBL that it
                                # is alive every (SCANUNIT * SANITYSCAN
                                # seconds.

DBBLWAIT        2               # (SCANUNIT * DBBLWAIT) is the time in
                                # seconds after which the DBBL will time
                                # out an unresponsive BBL.

BBLQUERY        30              # (SCANUNIT * BBLQUERY) is the frequency
                                # of status verification by the DBBL.
                                # If the DBBL has not received an "I'm OK"
                                # message from a BBL during this period,
                                # the DBBL will send a status query message
                                # to the BBL.  If no response is received,
                                # the BBL's node is partitioned.

# DEBUG
#dmk reduced from 6000 to 60
BLOCKTIME       60              # (SCANUNIT * BLOCKTIME) is the time in
                                # seconds after which a blocking call
                                # will time out from the client.
```

BLOCKTIME sets a multiplier of the basic SCANUNIT after which a blocking call (for example, receiving a reply) times out. The value of BLOCKTIME must be greater than 0. If this parameter is not specified, the default is set so that (SCANUNIT * BLOCKTIME) is approximately 60 seconds.

In the PeopleSoft delivered file this value is set to 6000, thus the blocking time is 30000 seconds, 8 hours 20 seconds.

```
#
CMTRET          COMPLETE
NOTIFY          DIPIN
USIGNAL         SIGUSR2

# ------------------------------------------------------------------------

*MACHINES
```

The MACHINES section specifies the logical names for physical machines and processing elements within multiprocessor computers for the configuration. It also specifies parameters specific to a given machine. The MACHINES section must contain an entry for each physical processor used by the application.

```
"{MACH}" LMID="simple"                          # Machine name must be upper
case
        TUXDIR="{$TUXDIR}"                      # Paths cannot end in '\'
        APPDIR="{$PS_SERVDIR}"                  # include the database path
        TUXCONFIG="{$TUXCONFIG}"
        ULOGPFX="{LOGDIR}{FS}TUXLOG"
        ENVFILE="{$PS_SERVDIR}{FS}{ENVFILE}"
```

The presence of the ENVFILE keyword specifies that all the servers in the machine are to be executed with the environment specified in the named file, which contains a list of environmental variables and the values to which they should be set.

```
        UID={UID}                               # Has to be 0 at this time.
        GID={GID}                               # Has to be 0 at this time.
{WINDOWS}
        TYPE="i386NT"
{WINDOWS}
        NETLOAD=0                               # We are not using multiple
machines.
        {MAXWSCLIENTS}
```

MAXWSCLIENTS specifies the number of accesser entries on this processor to be reserved for workstation clients only. The number specified here takes a portion of the total accesser slots specified with MAXACCESSERS. The appropriate setting of this parameter helps to conserve IPC resources since workstation client access to the system is multiplexed through a System/T supplied surrogate, the workstation handler. This value must be greater than or equal to 0 and less than 32,768. The default value is 0. It is an error to set this number to a value greater than MAXACCESSERS.

The value is calculated by *psadmin* but again the value will not be calculated properly if multiple queues are configured for the same server process because *psadmin* assumes that each occurrence of *Max Instances* is used for only one queue.

```
#       {MAXACCESSERS}
        MAXACCESSERS=140
```

MAXACCESSERS specifies the maximum number of processes that can have access to the bulletin board on this processor at any one time. System administration processes, such as the BBL and tmadmin, need not be accounted for in this figure, but all application servers and clients are counted. This value must be greater than 0 and less than 32,768. If not specified, the default value is the value specified in the *RESOURCES section.

The psadmin utility appears to calculate MAXACCESSERS as follows

MAXACCESSERS = (<maximum number of WSHs> * <WSH multiplexing factor>) + <maximum number of instances of all server processes> + 4

If you have multiple queues configured by the same *Max Instances* parameter, you must allow for these additional accessing servers in the calculation.

```
# ------------------------------------------------------------------------

*GROUPS

#
# Tuxedo Groups
# For application group numbers for new machines (LMIDs)
# use group numbers 101-199;  201-299;  etc.
#
```

The PeopleSoft application server is split into 4 logical groups. Each server group can only exist within a single machine.

BASE	Contains the WSL and WSH processes and the authorisation server PSAUTH
APPSRV	All other PeopleSoft application server processes
JREPGRP	JREPSRV only
JSLGRP	Contains the JSL and JSH processes

```
DEFAULT:
        LMID=simple

BASE    GRPNO=1

APPSRV  GRPNO=99

{JOLT}
#
# JOLT Groups
#
JREPGRP LMID=simple     GRPNO=94
JSLGRP  LMID=simple     GRPNO=95
```

```
{JOLT}

# ---------------------------------------------------------------------

*SERVERS

DEFAULT:
        CLOPT="{$Trace\TuxedoLoadTrace} -A"       # Advertise all services.
        REPLYQ=N          # Reply queue not needed for our simple setup.
        MAXGEN=3          # Max number of restarts in the grace period.
        GRACE=60          # Ten minutes grace period.
        RESTART={$Domain Settings\Restartable}
        SYSTEM_ACCESS=FASTPATH

#
# PeopleSoft Tuxedo Authentication Server
#
PSAUTH          SRVGRP=BASE
                MIN={$PSAUTH\Min Instances}
                MAX={$PSAUTH\Max Instances}
                SRVID=1
                SEQUENCE=1
                RQADDR="AUTHQ"
                REPLYQ=Y
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/AUTHQ.stderr -A -- -C
{CFGFILE} -D {$Domain Settings\Domain ID} -S PSAUTH"
```

Note that all servers have a server ID, *SRVID*. This must be unique within the server group, *SRVGRP*. However, it is advantageous if the servers IDs are unique within an entire domain. It means that if individual servers are to be manually booted or shutdown, the commands to be issued within *tmadmin* are less complicated.

The servers IDs shown in the examples in this chapter have been altered so that they are unique.

If there is more than one server on a queue, the servers are given sequential server IDs starting with the server ID specified. So the first PSATH process will be server ID 1, the next will be server ID 2, and so on.

A sequence number has been specified for this and a number of other servers. This specifies the order in which the queues and their servers are started.

If only some of the servers have a sequence number then the ones with sequence numbers are started first, and in sequence number order, then the rest of the servers are started in the order in which they appear in *psappsrv.ubb*.

In a large domain, the sequence number can be used to starts enough services to get the domain functioning as quickly as possible.

In this example a number of servers have been given sequences.

Sequence Number	Queue Name	Server
1	AUTHQ	PSAUTH
2	APPQ	PSAPPSRV
3	SAMQ	PSSAMSRV
4	QRYQ	PSQRYSRV
5	APIQ	PSAPISRV
6		WSL
7		JREPSRV
8		JSL

The Workstation listener is not started until all the services can be services by at least one server. Thus a user will not be able to log in and receive a service unavailable error.

PSQCKSRV has not been given a sequence because its services are available from PSAPPSRV. Thus it will be booted after all the numbered servers.

```
#
# Workstation Listener
#  -I xx   Max time (seconds) for a client connect
#  -T xx   Max time (minutes) for a client to stay idle.
#  -m xx   Min number of workstation handlers
#  -M xx   Max number of workstation handlers
#  -x xxx  Multiplexing, the max number of clients per handler
#
#
WSL             SRVGRP=BASE
                SRVID=120
                SEQUENCE=6
{WINDOWS}
                CLOPT="-A -- -n {$Workstation Listener\Address}:{$Workstation
Listener\Port} -z {$Workstation Listener\Encryption} -Z {$Workstation
Listener\Encryption} -I {$Workstation Listener\Init Timeout} {WSL Client Cleanup
Timeout} -m {$Workstation Listener\Min Handlers} -M {$Workstation Listener\Max
Handlers} -x {$Workstation Listener\Max Clients per Handler} -c {$Workstation
Listener\Tuxedo Compression Threshold}"
{WINDOWS}
{UNIX}
                CLOPT="-A -- -n {$Workstation Listener\Address}:{$Workstation
Listener\Port} -z {$Workstation Listener\Encryption} -Z {$Workstation
Listener\Encryption} -d {$PS_TUXDEV} -I {$Workstation Listener\Init Timeout}
{WSL Client Cleanup Timeout} -m {$Workstation Listener\Min Handlers} -M
{$Workstation Listener\Max Handlers} -x {$Workstation Listener\Max Clients per
Handler} -c {$Workstation Listener\Tuxedo Compression Threshold}"
{UNIX}
```

The server ID of the workstation listener has been changed so that it is unique throughout the domain.

```
WSL             SRVGRP=BASE
                SRVID=130
{WINDOWS}
                CLOPT="-A -- -n {$Workstation Listener\Address}:7500 -z
{$Workstation Listener\Encryption} -Z {$Workstation Listener\Encryption} -I
{$Workstation Listener\Init Timeout} {WSL Client Cleanup Timeout} -m
{$Workstation Listener\Min Handlers} -M {$Workstation Listener\Max Handlers} -x
{$Workstation Listener\Max Clients per Handler} -c {$Workstation Listener\Tuxedo
Compression Threshold}"
{WINDOWS}
{UNIX}
                CLOPT="-A -- -n {$Workstation Listener\Address}:7500 -z
{$Workstation Listener\Encryption} -Z {$Workstation Listener\Encryption} -d
{$PS_TUXDEV} -I {$Workstation Listener\Init Timeout} {WSL Client Cleanup
Timeout} -m {$Workstation Listener\Min Handlers} -M {$Workstation Listener\Max
Handlers} -x {$Workstation Listener\Max Clients per Handler} -c 100"
{UNIX}
```

A

second Workstation listener has been configured to listen on a different port on the same address. It has a compression threshold of only 100. So any message larger that 100 bytes will be compressed. It would have been possible to specify additional variables in *psappsrv.ubx* and *psappsrv.cfg* so that the values can be maintained via the interactive dialogue, but it is quite legitimate and easier for this example to simply hard code the values in *psappsrv.ubx*.

```
#
# Tuxedo System Event Server
#
TMSYSEVT        SRVGRP=BASE
                SRVID=200
```

The System Event Server or Broker is a tool that enhances the tracking of events in a server.

Publish and subscribe applications can create application events which are then detected by the broker. PeopleSoft has not coded any events into the application.

There are also system events that are monitored, these are mainly various forms of errors.

The principal use of the Event Broker in a PeopleSoft domain is to enhance the logging information that can be seen in the Tuxedo GUI Administrative Applet. This allows the Tuxedo domain to be administered, including shutdown, from a web browser.

Tuxedo Manual entry for System Events	
V6.4	tuxedo/tux64/sect5/events.htm
V6.5	tuxedo/tux65/refman/sect5/sect517.htm - 1000754

```
#
# PeopleSoft Manager Application Server
#
PSAPPSRV        SRVGRP=APPSRV
                SRVID=10
                SEQUENCE=2
                MIN={$PSAPPSRV\Min Instances}
                MAX={$PSAPPSRV\Max Instances}
                RQADDR="APPQ"
                REPLYQ=Y
{QUERYSRV}
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/APPQ.stderr
{$PSAPPSRV\User Spawn} -A -- -C {CFGFILE} -D {$Domain Settings\Domain ID} -S
PSAPPSRV"
{QUERYSRV}

{!QUERYSRV}
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/APPQ.stderr
{$PSAPPSRV\User Spawn} -A -sSqlQuery:SqlRequest -- -C {CFGFILE} -D {$Domain
Settings\Domain ID} -S PSAPPSRV"
{!QUERYSRV}
```

The server ID has been changed from 1 to 10 so that it is unique within the whole domain, not just the server group.

The CLOPT parameter specifies the command line which server is started. Anything to the left of the double hyphen (--) is interpreted as a Tuxedo parameter and can be looked up in the Tuxedo documentation. Anything to the right of the double hyphen is a PeopleSoft parameter that is passed to the tpsvrinit() function which is executed when a server is booted, before it begins processing requests (see Simple Server on page 123).

The –A parameter in CLOPT indicates the PSAPPSRV advertises all services defined within it.

If neither PSQRYSRV nor PSQCKSRV servers are configured then the SqlQuery Service alias is applied to the PSAPPSRV. If there are no PSQCKSRV processes either then it is added to the PSAPPSRV. Even though all services are advertised on PSAPPSRV, the service SqlQuery aliases must be defined with a separate –s parameter.

As mentioned elsewhere, BEA recommends that no more than 10 server processes are run on the same queue. Above this number the will contend with each on the shared memory message queue and Bulletin Board shared memory segment.

If a system requires more than 10 of any one server, the solution is to configure a second queue with more instances of the same server.

```
PSAPPSRV        SRVGRP=APPSRV
                SRVID=20
                MIN={$PSAPPSRV\Min Instances}
                MAX={$PSAPPSRV\Max Instances}
                RQADDR="APPQ2"
                REPLYQ=Y
{QUERYSRV}
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/APPQ2.stderr
{$PSAPPSRV\User Spawn} -A -- -C {CFGFILE} -D {$Domain Settings\Domain ID} -S
PSAPPSRV"
{QUERYSRV}

{!QUERYSRV}
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/APPQ2.stderr
{$PSAPPSRV\User Spawn} -A -sSqlQuery:SqlRequest -- -C {CFGFILE} -D {$Domain
Settings\Domain ID} -S PSAPPSRV"
{!QUERYSRV}
```

In the example a second application server queue has be specified. The server IDs for this queue will start at 20. The queue name must also be unique within the domain. The same variables from psappsrv.cfg are used to specify the minimum and maximum number of servers on queue APPQ2 as were used for APPQ. This ensures that both queues have the same number of servers, at least initially. They can handle the same amount of work.

Important: It is essential that load balancing is enabled if this approach is used.

```
{QUICKSRV}
#
# PeopleSoft Manager Application Server
#
PSQCKSRV        SRVGRP=APPSRV
                SRVID=50
                MIN={$PSQCKSRV\Min Instances}
                MAX={$PSQCKSRV\Max Instances}
                RQADDR="QCKQ"
                REPLYQ=Y
{!QUERYSRV}
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/QCKQ.stderr
{$PSQCKSRV\User Spawn} -s
MgrClear,RamList,SqlRequest,StmChgPswd,StmGetTimeOut,SamGetParmsSvc,WamChgInstSv
c,WamStartInstSvc -sSqlQuery:SqlRequest -- -C {CFGFILE} -D {$Domain
Settings\Domain ID} -S PSQCKSRV"
{!QUERYSRV}

{QUERYSRV}
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/QCKQ.stderr
{$PSQCKSRV\User Spawn} -s
MgrClear,RamList,SqlRequest,StmChgPswd,StmGetTimeOut,SamGetParmsSvc,WamChgInstSv
c,WamStartInstSvc -- -C {CFGFILE} -D {$Domain Settings\Domain ID} -S PSQCKSRV"
{QUERYSRV}
```

The PSQCKSRV program is simply a copy of PSAPPSRV. While the command line option for PSAPPSRV has the –A parameter in CLOPT, indicating that the PSAPPSRV advertises all services defined within it, PSQCKSRV has an explicit list of the 8 quick services that it advertises.

If no PSQRYSRV servers are configured then the SqlQuery Service alias is applied to the PSQCKSRV. If there are no PSQCKSRV processes either then it is added to the PSAPPSRV. Service aliases must be defined with a separate –s parameter.

```
PSQCKSRV        SRVGRP=APPSRV
                SRVID=60
                MIN={$PSQCKSRV\Min Instances}
                MAX={$PSQCKSRV\Max Instances}
                RQADDR="QCKQ2"
                REPLYQ=Y
{!QUERYSRV}
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/QCKQ2.stderr
{$PSQCKSRV\User Spawn} -s
MgrClear,RamList,SqlRequest,StmChgPswd,StmGetTimeOut,SamGetParmsSvc,WamChgInstSv
c,WamStartInstSvc -sSqlQuery:SqlRequest -- -C {CFGFILE} -D {$Domain
Settings\Domain ID} -S PSQCKSRV"
{!QUERYSRV}
```

As with PSAPPSRV, a second queue for PSQCKSRV has been configured.

```
{QUERYSRV}
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/QCKQ2.stderr
{$PSQCKSRV\User Spawn} -s
MgrClear,RamList,SqlRequest,StmChgPswd,StmGetTimeOut,SamGetParmsSvc,WamChgInstSv
c,WamStartInstSvc -- -C {CFGFILE} -D {$Domain Settings\Domain ID} -S PSQCKSRV"
{QUERYSRV}

{QUICKSRV}

{QUERYSRV}
#
# PeopleSoft Query Application Server
#
PSQRYSRV        SRVGRP=APPSRV
                SRVID=70
                SEQUENCE=4
                MIN={$PSQRYSRV\Min Instances}
                MAX={$PSQRYSRV\Max Instances}
                RQADDR="QRYQ"
                REPLYQ=Y
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/QRYQ.stderr
{$PSQRYSRV\User Spawn} -sSqlQuery:SqlRequest -- -C {CFGFILE} -D {$Domain
Settings\Domain ID} -S PSQRYSRV"
{QUERYSRV}

#
# PeopleSoft SQL Access Application Server
#
PSSAMSRV        SRVGRP=APPSRV
                SRVID=100
                SEQUENCE=3
                MIN={$PSSAMSRV\Min Instances}
                MAX={$PSSAMSRV\Max Instances}
                RQADDR="SAMQ"
                REPLYQ=Y
                CONV=Y
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/SAMQ.stderr -A -- -C
{CFGFILE} -D {$Domain Settings\Domain ID} -S PSSAMSRV"
```

Note: PSSAMSRV has been declared to Tuxedo as conversational with the CONV keyword.

```
#
# PeopleSoft API Application Server
#
PSAPISRV        SRVGRP=APPSRV
                SRVID=150
                SEQUENCE=5
                MIN={$PSAPISRV\Min Instances}
                MAX={$PSAPISRV\Max Instances}
                RQADDR="APIQ"
                REPLYQ=Y
                CLOPT="{$Trace\TuxedoLoadTrace} -e LOGS/APIQ.stderr
{$PSAPISRV\User Spawn} -A -- -C {CFGFILE} -D {$Domain Settings\Domain ID} -S
PSAPISRV"

{JOLT}
#
# JOLT Listener and Rep Server
#
JSL
                SRVGRP=JSLGRP
                SRVID=200
                SEQUENCE=8
                CLOPT="-A -- {TUXDEV} -n {$JOLT Listener\Address}:{$JOLT
Listener\Port} -m {$JOLT Listener\Min Handlers} -M {$JOLT Listener\Max Handlers}
-I {$JOLT Listener\Init Timeout} -c {$JOLT Listener\Client Connection Mode} -x
{$JOLT Listener\Max Clients per Handler} {Jolt Encryption} {Jolt Client Cleanup
Timeout} -w JSH"

JREPSVR
                SRVGRP=JREPGRP
                SRVID=250
                SEQUENCE=7
#dmk - removed -W, its a security risk
#                CLOPT="-A -- -W -P {$PS_SERVDIR}{FS}jrepository"
                CLOPT="-A -- -P {$PS_SERVDIR}{FS}jrepository"
```

The –W parameter has been removed from the Java Repository Server because otherwise the repository database can be updated and it would be technically possible for someone to add or remove an entries, thus adding or remove a tuxedo service to or from a Java client.

```
{JOLT}

{JRAD}
#
# JOLT Internet Relay (Back End)
#
JRAD
                SRVGRP=JSLGRP
                SRVID=2501
                CLOPT="-A -- -l {$JOLT Relay Adapter\Listener Address}:{$JOLT
Relay Adapter\Listener Port} -c {$JOLT Listener\Address}:{$JOLT Listener\Port}"
{JRAD}

*SERVICES
```

In PeopleTools 7.54.10 there was no specification in psappsrv.ubx for the SqlQuery services. This bug was fixed in PT7.55. However, it meant that there was no timeout for the SqlQuery services. So if a user wrote and submitted a poor query in PS/Query, got bored, and killed the client process and tried again, the submitted query would run until the query completed, sometimes effectively never. This would tie up a PSQRYSRV process as well as continuing to consume large amounts of database resources. The solution was either to code a service entry for the missing server, or to specify a default timeout for all services, or both.

```
# dmk - default timeouts (if not specified) of 15 min
DEFAULT: LOAD=50 PRIO=50 SVCTIMEOUT=900
```

The loads for a number of the services have been changed from the default of 50 specified in the vanilla configuration files. These values were calculated at a customer site, and are fully explained in the section Load Balancing below (see page 62).

```
GetCertificate
                LOAD=40 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

MenuList
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

MgrGetObject
                LOAD=13 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

MgrUpdObject
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

MgrClear
                LOAD=30 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

MgrPeek
                LOAD=3 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

JavaMgrGetObj
                LOAD=7 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprLoad
                LOAD=78 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprSave
                LOAD=386 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprFullSave
                LOAD=135 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprChangeValue
                LOAD=6 PRIO=50 #dmk - calculated load
```

```
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprFieldChange
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprDefault
                LOAD=5 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprInsertRow
                LOAD=7 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprDeleteRow
                LOAD=6 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprSearchStart
                LOAD=3 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprSearchSave
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprSecStart
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprMItemSelPC
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprPrePopupPC
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PsmSchedPrcs
                LOAD=12 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"
```

```
PsdGetListSvc
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

QdmGetListSvc
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

RamList
                LOAD=63 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

RemoteCall
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

SamGetParmsSvc
                LOAD=7 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

SqlAccess
                LOAD=22 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSSAMSRV\Service Timeout}
                BUFTYPE="ALL"

# dmk - 20/7/99 - new service defn to specify a time out
SqlQuery
                LOAD=1630 PRIO=50
                SVCTIMEOUT={$PSQRYSRV\Service Timeout}
                BUFTYPE="ALL"


SqlRequest
                LOAD=3 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

StmChgPswd
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

StmGetTimeOut
                LOAD=6 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"
```

```
StmGetExplain
                LOAD=3 PRIO=50 #dmk - calculated load
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

OpnQryDescribe
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

OpnQryExecute
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

WamChgInstSvc
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

WamStartInstSvc
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

MsgAPI
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPISRV\Service Timeout}
                BUFTYPE="ALL"
# ------------------------------------------------------------------------
```

The following section is a PeopleSoft specific section. It is used by *ubbgen* to generate the *psappsrv.env* file. The file is referenced by the ENVFILE variable in the Machine section of *psappsrv.ubx* and *psappsrv.ubb*.

```
*PS_ENVFILE
{WINDOWS}
PATH={$PS_HOME}\bin\server\winx86;{$Domain Settings\Add to PATH}
{WINDOWS}
{UNIX}
LD_LIBRARY_PATH={$LD_LIBRARY_PATH}
LIBPATH={$LIBPATH}
SHLIB_PATH={$SHLIB_PATH}
COBPATH={$PS_HOME}/bin
PATH={$PS_HOME}/bin:{$Domain Settings\Add to PATH}
{UNIX}
INFORMIXSERVER={$Startup\ServerName}
{UNIX}
```

psappsrv.env

This is an example *psappsrv.env* generated on an NT machine.

```
PATH=d:\ps\hr75\bin\server\winx86;d:\orant80\bin
INFORMIXSERVER=
```

Load Balancing

What happens without Load Balancing?

In order to understand the importance of enabling Load Balancing for a PeopleSoft application server domain it is first necessary to explain what happens if load balancing is not enabled.

The Bulletin Board maintains a table of servers in the form of a stack. Whenever an server is booted it is placed on the top of the stack. So the stack is ordered in the order in which the servers were started, rather than server ID order.

Where there is a choice as to which queue a request maybe placed upon, the WSH starts at the top of the stack, and works its way down looking for an idle server. If it finds one it places the request on the queue for that server. If no suitable server is idle, then it round-robins between queues.

In a default domain the only time this is an issue is with the services that are advertised on PSQCKSRV.

If a PSAPPSRV process is started either manually, or by Tuxedo spawning, or because it reaches its recycle count, the entry for the new server is placed at the top of the stack. In this situation, requests for services advertised on PSQCKSRV are likely to be sent to the new PSAPPSRV. This can cause short requests to be queued up for PSAPPSRVs behind longer requests. The resultant queuing causes response times to degrade.

If there is no choice as to which queue a request maybe placed upon, then this issue does not arise and load balancing should not be enabled.

What happens without Balancing?

If load balancing is enabled then a completely different algorithm is used to choose the queue.

If any queue is idle, then the request is placed on that queue. Otherwise, the sum of the loads of service requests queued on each the queue upon which it is possible to place the new request is calculated. The request is placed on the queue with the lowest load queued up on it.

An Analogy

A useful analogy is a row of supermarket checkouts. The checkouts are the server processes, the queues are still first in first out queues. The shoppers with their baskets represent the messages requesting a service.

When you have finished shopping how do you choose which queue for which checkout to join?

There are standard checkouts, and there are usually either '10 items or less' or 'basket not trolley' checkouts. The standard checkouts are like PSAPPSRVs, the 'basket' checkouts are PSQCKSRVs. A shopper with just a basket is allowed to use a standard checkout but not vice versa. Similarly the quick services are advertised on PSAPPSRV, but the slow services are not advertised on PSQCKSRV.

A person with just a basket might choose to use a standard checkout if there was no queue, but they are unlikely to choose to wait behind even a single big basket unless there is a huge queue for the basket checkout..

What about the shopper with a trolley, who is faced with a choice of queue?

What most people do is look at the queues and make a judgement as which queue will take the shortest amount of time. To put it a different way, which queue has the least load queued on it. That judgement is more sophisticated than simply counting the number of baskets, it will involve an estimate as to how many items are in the basket.

This is exactly what Tuxedo does. It uses the definition of the service in the Services section in the configuration to obtain the load value for the service and so sums the loads of all the service requests in the queue.

Tuxedo Load Balancing in a PeopleSoft Domain

As delivered, all the services have a load of 50. This is like saying that all trolleys in the supermarket take the same amount of time to get through the checkout. This is demonstrably not the case.

If you choose your queue simply on the basis of the number of trolleys in the queue, then you would get in a queue behind two trolleys filled to brimming over, in preference to three almost empty trolleys.

If load balancing is enabled in a PeopleSoft domain where all the services have the default load of 50, services that could be handled by PSQCKSRV will be queued for PSAPPSRV. The resultant performance will be even worse than if load balancing was disabled.

If load balancing is to be used, then realistic loads must be applied to the services.

Calculating Service Loads

If the –r parameter is placed in the command line option for a server then that server will record a log of all services performed in its standard errors file.

By default the standard error file is written in the application server directory and is called *stderr*. The name can be modified with the –e parameter, so you can have a standard error file per queue, thus enabling analysis of an individual queue.

Standard Error Files

The following standard error files were obtained by starting the application server with the configuration file exhibited in this chapter and going into PeopleTools-Utilities->DDL Model Defaults.

Each service request wrote a line to the file. There are values. The name of the service, the operating system process ID of the server process that wrote the line. There are a pair of start times and a part of end times.

The first number of each pair, SDATE and EDATE, is the system time in seconds since 0:00hrs on January 1st, 1970, this is a standard way to define a time in a 'C' or 'Unix' environment.

The second number of the pair, STIME and ETIME, is a time string in 1/100th of a second. There is no particular origin, the number wraps round to zero occasionally. Analysis has shown that this value looses a few 1/100th of a second per day, so it is not directly coupled to the system time.

SERVICE	PID	SDATE	STIME	EDATE	ETIME
@MgrGetObject	284	941649117	9129637	941649126	9138860
@MgrPeek	284	941649130	9142726	941649130	9142746
@MgrGetObject	284	941649130	9142766	941649133	9146141
@MgrGetObject	284	941649133	9146171	941649133	9146191
@PprLoad	284	941649134	9146812	941649137	9149316

APPQ2.stderr

SERVICE	PID	SDATE	STIME	EDATE	ETIME
@SamGetParmsSvc	284	941649086	9099143	941649087	9099644
@SqlRequest	284	941649087	9100215	941649088	9100285
@SqlRequest	284	941649088	9100315	941649088	9100505
@SqlRequest	284	941649088	9100545	941649088	9100605
@SqlRequest	284	941649088	9100615	941649088	9100636
@SqlRequest	284	941649088	9100646	941649088	9100666
@SqlRequest	284	941649088	9100706	941649088	9100726
@SqlRequest	284	941649088	9100746	941649088	9100766
@MgrClear	284	941649088	9100996	941649088	9101076
@SqlRequest	284	941649089	9101707	941649089	9101737
@MgrClear	284	941649097	9109518	941649097	9109538
@MgrClear	284	941649097	9109769	941649097	9109799
@SqlRequest	284	941649097	9109979	941649098	9110339
@SqlRequest	284	941649099	9111291	941649099	9111511
@SqlRequest	284	941649099	9111531	941649099	9111731
@SqlRequest	284	941649099	9111752	941649099	9112082
@SqlRequest	284	941649101	9114115	941649101	9114145
@MgrClear	284	941649103	9115998	941649103	9116058
@SqlRequest	284	941649104	9117159	941649104	9117179
@MgrClear	284	941649116	9128706	941649116	9128726
@SqlRequest	284	941649127	9139311	941649127	9139331
@RamList	284	941649128	9140603	941649128	9141034
@SqlRequest	284	941649134	9146421	941649134	9146441

QCKQ2.stderr

The standard error files can be processed with the *txrpt* utility with a simple script.

```
%TUXDIR%\bin\txrpt <APPQ2.stderr >txrpt.txt
%TUXDIR%\bin\txrpt <QCKQ2.stderr >>txrpt.txt
```

txrpt.bat

Tuxedo Manual entry for txrpt	
V6.4	tuxedo/tux64/sect1/txrpt.htm
V6.5	tuxedo/tux65/refman/sect1/sect146.htm

The result is this report.

```
SERVICE SUMMARY REPORT


SVCNAME         18p-19p      TOTALS
                Num/Avg      Num/Avg
--------------- --------     -------
MgrGetObject    3/4.21       3/4.21
PprLoad         1/2.50       1/2.50
MgrPeek         1/0.02       1/0.02
--------------- -------      -------
TOTALS          5/3.03       5/3.03
      SERVICE SUMMARY REPORT


SVCNAME         18p-19p      TOTALS
                Num/Avg      Num/Avg
--------------- --------     -------
SqlRequest      16/0.10      16/0.10
MgrClear        5/0.04       5/0.04
RamList         1/0.43       1/0.43
SamGetParmsSvc  1/0.50       1/0.50
--------------- -------      -------
TOTALS          23/0.12      23/0.12
```

Txrpt.txt

The report contains the number of services and the average execution time in seconds for each service on each queue, for each hour of the day, and a summary of the whole day.

Even from this very brief test, it is clear that most of the quick services were naturally queue on PSQCKSRV by the load-balancing algorithm. At quiet times it doesn't matter where the services are handled.

Obviously, to get a realistic picture the trace needs to be collected over a longer period of time. When this has been done on live production systems, it is clear that 99% of the requests for the quick services go to the quick servers, queuing is reduced and performance may improve. It is more likely that any database problem is highlighted either by removing the mask of the application server queuing, or by increasing the load on the database.

Having obtained an average execution time for a service, the load for that service is proportional to the execution time. The load on a queue is a measure of how long you would expect for the requests on a queue to take to process.

Specifying the Load for a Service

It is recommended that the load on a service should be defined in the psappsrv.ubx as the execution time in $1/100^{th}$ of a second.

Priority

The priority of a service controls how it is dequeued. That is to say which enqueued request is taken off the queue to be serviced by a server.

The value must be greater than 0 and less than or equal to 100, with 100 being the highest priority. The default is 50. All PeopleSoft services are defined as having a priority of 50.

The highest assigned priority gets first preference. It is recommended that this should occur less frequently. A lower priority message does not remain forever enqueued, because every tenth message is retrieved on a FIFO basis. Response time should not be a concern of the lower priority interface or service.

There is no particular advantage to be gained by giving particular services a lower priority.

Splitting Up Services

It is quite legitimate to change the configuration as shipped by PeopleSoft, in order to remove services from particular queues and advertise them on other queues.

Separate Queue for *RemoteCall*

A particular customer (Financials 7, PeopleTools 7.01.30) was experiencing queuing problems inside the application server, and this was aggravated during month end processing by the excessive time taken to perform on-line voucher edit and post.

This functionality makes use of the *remote call* facility. In two-tier mode the client would initiate a Cobol process on the client which would make another two-tier connection across the network to the database. In three-tier mode the server that advertises the *RemoteCall* service initiates the same Cobol process.

Just as requests for one of the 8 quick services that end up on PSAPPSRV can end up queuing for a long time, other requests were found to be queuing behind *RemoteCall* requests. So, the configuration was changed such that *RemoteCall* was deadvertised from PSAPPSRV and placed, on its own, on a separate queue.

This is the new entry for the new queue in *psappsrv.ubx*.

```
#dmk 1/6/99 remote call on separate queue
PSAPPSRV        SRVGRP=APPSRV
                SRVID=110
                MIN=5
                MAX=10
                RQADDR="REMOTEQ"
                REPLYQ=Y
#dmk 1/6/99 remote call on separate queue
                CLOPT="-s RemoteCall -- -C {CFGFILE} -D {$Domain Settings\Domain
ID}"
```

The PSAPPSRV server executable was still used for this new queue, because it provides the service. The queue was given a distinct name. The –s parameter was used in the command line option string to specify that this queue only advertises one server, *RemoteCall*.

It was expected that, usually, no more than 5 users would, simultaneously, be running on-line voucher edit/post, so five servers were configured on the new queue. Allowances for up to 10 server processes were made. Automatic spawning was not configured (because it didn't work properly in that port of Tuxedo 6.3), and so the additional servers would have been started manually.

Deadvertising other Services

Simultaneously a decision was taken to deadvertise the 8 quick services from PSAPPSRV so that they could no longer go to PSAPPSRV.

The PSAPPSRVs were specified as

```
PSAPPSRV        SRVGRP=APPSRV
                SRVID=10
                MIN={$PSAPPSRV\Min Instances}
                MAX={$PSAPPSRV\Max Instances}
                RQADDR="APPQ"
                REPLYQ=Y
CLOPT="-s
PprSecStart,PprSearchStart,PprSearchSave,PprSave,PprLoad,PprInsertRow,PprFullSav
e,PprFieldChange,PprDeleteRow,PprDefault,PprChangeValue,OpnQryExecute,OpnQryDesc
ribe,MgrUpdObject,MgrPeek,MgrGetObject,JavaMgrGetObj -- -C {CFGFILE} -D {$Domain
Settings\Domain ID}"
```

Only those services explicitly referenced in the –s option are advertised on PSAPPSRV.

Note: NT imposes a 256-character limit on the length of the command line. The services would have to be listed in a file referenced on in the command line. This is approach is now used in PeopleTools 8.

```
#
# PeopleSoft Application Server
#
PSAPPSRV        SRVGRP=APPSRV
                SRVID=1
                MIN=1
                MAX=3
                RQADDR="APPQ"
                REPLYQ=Y
                CLOPT="-p 1,600:2,30 -s@..\psappsrv.lst -- -C psappsrv.cfg -D
TESTSERV -S PSAPPSRV"
```

extract of psappsrv.ubx

```
# Services for PSAPPSRV
JavaMgrGetObj
MenuList
MgrGetObject
MgrUpdObject
MgrPeek
ObjectAccess
PprLoad
PprSave
PprFullSave
PprChangeValue
PprFieldChange
PprDefault
PprInsertRow
PprDeleteRow
PprSearchStart
PprSearchSave
PprSecStart
PprMItemSelPC
PprPrePopupPC
PsmSchedPrcs
PsdGetListSvc
QdmGetListSvc
RemoteCall
StmGetExplain
TdmUtilSvc
OpnQryDescribe
OpnQryExecute
Publish
UpdatePubData
ResubmitPub
CancelPub
GetPub
SubContract
PubContract
# Services for PSAUTH
GetCertificate
PSAUTH
# Services for PSAPISRV
MsgAPI
PSSession
PSBusComp
XmlDriver
PSBusInterlink
# Services for PSICSRV
ICPanel
ICScript
ICNav
```

PeopleTools 8 %PS_HOME%/appserv/psappsrv.lst

Calculation of other Tuxedo Settings

These variables are normally calculated by psadmin, but if more additional queues have been established *psadmin* does not cope with this, and will not calculate adequate values.

The parameters must be manually calculated to allow the maximum number of servers within the domain to be started on all queues.

MAXSERVERS

MAXSERVERS is the total number of server processes that can be run, not including the BBL process.

Using the example configuration file used throughout this chapter, it should have been calculated as:

```
(2 WSLs * 2 WSH/WSL) + system event broker + (1 AUTHQ * 2 PSAUTHs) + (2 QCKQ
queues * 2 PSQCKSRVs) + (2 APPQ queues * 2 PSAPPSRVs) + 2 PSQRYSRVs + 2
PSSAMSRVs + 2 PSAPISRVs + (1 JSL * 2 JSH/JSL) + 1 JREPSRV
= 4 + 1 + 2 + 4 + 4 + 2 + 2 + 2 + 2 + 1
= 24
```

MAXSERVICES

MAXSERVICES is sum of all the services available on all of the servers.

Using the same example it should be calculated as

```
(1 AUTHQ * 2 PSAUTHs + 2 services/PSAUTH) + (2 QCKQ queues * 2 PSQCKSRVs * 8
services/PSQCKSRV) + (2 APPQ queues * 2 PSAPPSRVs * 33 services/PSAPPSRV) + (2
PSQRYSRVs * 1 service/PSQRYSRV) + (2 PSSAMSRVs * 1 service/PSSAMSRV) + (2
PSAPISRVs * 1 services/PSAPISRV)
= 4 + 32 + 132 + 2 + 2 + 2
= 174
```

MAXWSCLIENTS

MAXWSCLIENTS specifies the number of accesser entries on this processor to be reserved for workstation clients only. The parameter is only used when the BEA TUXEDO system Workstation feature is used. The number specified here takes a portion of the total accesser slots specified with MAXACCESSERS. The appropriate setting of this parameter helps to conserve IPC resources since workstation client access to the system is multiplexed through a BEA TUXEDO system-supplied surrogate, the workstation handler. This value must be greater than or equal to 0 and less than 32,768. The default is 0. It is an error to set this number greater than MAXACCESSERS.

MAXACCESSERS

MAXACCESSERS specifies the maximum number of processes that can have access to the bulletin board on this processor at any one time. It is specified in the machine section of *psappsrv.ubx/psappsrv.ubb*. The Tuxedo default for this parameter is 50. The value of this variable must not be less that MAXSERVERS, otherwise server or handler processes will not start. *Psadmin* calculates this variable as *MAXSERVERS + MAXWSCLIENTS.*

CHAPTER 4

TUXEDO UTILITIES

There are a variety of Tuxedo commands and utilities that can be issued at the operating system command line. PeopleSoft's *psadmin* utility is a wrapper for a number of these commands.

This chapter examines just a few commands that might be of particular use.

Tuxedo Manual entry for Tuxedo OS Commands	
V6.4	tuxedo/tux64/fman1.htm
V6.5	tuxedo/tux65/refman/sect1/index.htm

tmadmin

tmadmin, provides a command line interface through which the application server domain can be monitored and controlled.

It can be invoked from within *psadmin* or from the operating system command line.

It can take various parameters. The –r option instructs *tmadmin* to enter the bulletin board as a client instead of the administrator and provides read-only access. This is useful if it is desired to leave the administrator slot unoccupied. Only one *tmadmin* process can be the administrator at a time. When the -r option is specified by a user other than the BEA TUXEDO administrator and security is turned on, the user will be prompted for a password.

Tuxedo Manual entry for tmadmin	
V6.4	tuxedo/tux64/sect1/tmadmin.htm
V6.5	tuxedo/tux65/refman/sect1/sect124.htm

help (h) [{command|all}]

Print help messages. If command is specified, the abbreviation, arguments, and description for that command are printed. All causes a description of all commands to be displayed. Omitting all arguments causes the syntax of all commands to be displayed.

Configuration

boot(b) and shutdown(stop)

Individual server processes can be manually started and stopped. Each server process requires a ‘slot’ on the bulleting board. The slot is given a server ID that is unique within the group.

In order to start or stop a particular server process you need to know the server ID number. If the server ID is not unique in itself, the group must also be specified.

```
>boot –I 54 –g PSAPPSRV
INFO: TUXEDO(r) System Release 6.5
INFO: Serial #: 1000000044, Expiration NONE, Maxusers 1000000
INFO: Licensed to: PeopleSoft

Booting server processes ...

exec PSQCKSRV -r -e LOGS/QCKQ.stderr -p L50,600:100,60 -s
MgrClear,RamList,SqlRequest,StmChgPswd,StmGetTimeOut,SamGetParmsSvc,WamChgInstSv
c,WamStartInstSvc -- -C psappsrv.cfg -D H75D -S PSQCKSRV :
        process id=319 ... Started.
1 process started.
```

The above command will start a new PSQCKSRV process, server ID on the QCKQ queue. If the command parameters are ambiguous more than one server will be affected. For instance *boot*, without any parameters will boot all server processes. The following command will boot all the servers in the PSAPPSRV group

```
boot -g PSAPPSRV
```

The stop command will stop the specified process or processes. It takes the same arguments.

```
> stop -i 50
Shutting down server processes ...

        Server Id = 50 Group Id = APPSRV Machine = simple:      shutdown
succeeded
1 process stopped.
```

advertise (adv) {-q qaddress [-g groupname] [-i srvid] | -g groupname -i srvid}service[:func], unadvertise (unadv) {-q qaddress [-g groupname] [-i srvid] | -g groupname -i srvid}service[:func]

It is possible to add or remove an entry in the service table in the Bulletin Board for an indicated service.

The following sequence of commands show the SqlRequest service being removed from a particular queue.

```
> psc -g APPSRV -i 50
Service Name Routine Name Prog Name  Grp Name  ID    Machine  # Done Status
------------ ------------ ---------  --------  --    -------  ------ ------
WamStartIns+ WamStartIns+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
WamChgInstS+ WamChgInstS+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
SamGetParms+ SamGetParms+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
StmGetTimeO+ StmGetTimeO+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
StmChgPswd   StmChgPswd   PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
SqlRequest   SqlRequest   PSQCKSRV.+ APPSRV    50     simple       1 AVAIL
RamList      RamList      PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
MgrClear     MgrClear     PSQCKSRV.+ APPSRV    50     simple       0 AVAIL

> unadv -g APPSRV -i 50 SqlRequest
SqlRequest removed from 1 server.
```

Note: The message is misleading, the service is removed from all servers on the same queue

```
> psc -g APPSRV -i 50
Service Name Routine Name Prog Name  Grp Name  ID    Machine  # Done Status
------------ ------------ ---------  --------  --    -------  ------ ------
WamStartIns+ WamStartIns+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
WamChgInstS+ WamChgInstS+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
SamGetParms+ SamGetParms+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
StmGetTimeO+ StmGetTimeO+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
StmChgPswd   StmChgPswd   PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
RamList      RamList      PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
MgrClear     MgrClear     PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
```

SqlRequest has indeed been removed from server 50.

```
> psc -g APPSRV -i 51
Service Name Routine Name Prog Name  Grp Name  ID    Machine  # Done Status
------------ ------------ ---------  --------  --    -------  ------ ------
WamStartIns+ WamStartIns+ PSQCKSRV.+ APPSRV    51     simple       0 AVAIL
WamChgInstS+ WamChgInstS+ PSQCKSRV.+ APPSRV    51     simple       0 AVAIL
StmGetTimeO+ StmGetTimeO+ PSQCKSRV.+ APPSRV    51     simple       0 AVAIL
StmChgPswd   StmChgPswd   PSQCKSRV.+ APPSRV    51     simple       0 AVAIL
SamGetParms+ SamGetParms+ PSQCKSRV.+ APPSRV    51     simple       0 AVAIL
RamList      RamList      PSQCKSRV.+ APPSRV    51     simple       0 AVAIL
MgrClear     MgrClear     PSQCKSRV.+ APPSRV    51     simple       0 AVAIL
```

But it has also been removed from all the other servers on the queue.

```
> adv -g APPSRV -i 50 SqlRequest
SqlRequest advertised on 1 server on queue QCKQ.

> psc -g APPSRV -i 50
Service Name Routine Name Prog Name  Grp Name  ID    Machine  # Done Status
------------ ------------ ---------  --------  --    -------  ------ ------
SqlRequest   SqlRequest   PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
WamStartIns+ WamStartIns+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
WamChgInstS+ WamChgInstS+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
SamGetParms+ SamGetParms+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
StmGetTimeO+ StmGetTimeO+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
StmChgPswd   StmChgPswd   PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
RamList      RamList      PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
MgrClear     MgrClear     PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
```

The server can also be added to a server. Again, the message is misleading. The service is added to all servers on the queue.

changeload (chl)

It is possible to change the service load for a particular service within a particular queue.

```
> scp -g APPSRV -i 50 -s SqlRequest
    Service Name: SqlRequest
   Function Name: SqlRequest
            Load: 3
        Priority: 50
         Address: 0x1b

> chl -g APPSRV -i 50 -s SqlRequest 5
2 entries changed.
```

Note that both the servers (50 and 51) on this queue have been changed.

```
> scp -g APPSRV -i 50 -s SqlRequest
    Service Name: SqlRequest
   Function Name: SqlRequest
            Load: 5
        Priority: 50
         Address: 0x1b

> scp -g APPSRV -i 51 -s SqlRequest
    Service Name: SqlRequest
   Function Name: SqlRequest
            Load: 5
        Priority: 50
         Address: 0x1b
```

However, the same service on a different queue has not been changed.

```
> scp -g APPSRV -i 10 -s SqlRequest
    Service Name: SqlRequest
   Function Name: SqlRequest
            Load: 3
        Priority: 50
         Address: 0x1b
```

Monitoring

tmadmin can be run from a command line, and the following monitoring commands can be logged to a file, and hence processed to obtain performance metrics.

changetrace (chtr)

Change the runtime tracing behaviour of currently executing processes to *on* or *off*. To change the trace specification of a specific currently running server process, supply the -g and -i options. To change the configuration of currently running server processes in a specific group, supply the -g option without the –i option. To change the configuration of all currently running client and server processes on a particular machine, specify the -m option. If none of the -g, -i, and -m options is supplied, then all non-administrative processes on the default machine are affected. This command does not affect the behaviour of clients or servers that are not currently executing, nor /WS clients.

```
> chtr on
Trace configuration of clients and servers changed on machine simple
```

The enhanced trace is written to the TUXLOG file

```
192638.GO-FASTER-1!PSQCKSRV.370: TRACE:at:  { tpservice({"SqlRequest", 0x0,
0x17fa8f0, 245, 0, 0, {942087529, 0, 40}})
192638.GO-FASTER-1!PSQCKSRV.370: TRACE:at:    { tpalloc("CARRAY", "", 8192)
192638.GO-FASTER-1!PSQCKSRV.370: TRACE:at:    } tpalloc = 0x18158e8
192638.GO-FASTER-1!PSQCKSRV.370: TRACE:at:    { tpreturn(2, 0, 0x18158e8, 83,
0x0)
192638.GO-FASTER-1!PSQCKSRV.370: TRACE:at:    } tpreturn [long jump]
192638.GO-FASTER-1!PSQCKSRV.370: TRACE:at:  } tpservice
192640.GO-FASTER-1!PSQCKSRV.370: TRACE:at:  { tpservice({"RamList", 0x0,
0x1811608, 185, 0, 0, {942087529, 0, 40}})
192640.GO-FASTER-1!PSQCKSRV.370: TRACE:at:    { tpalloc("CARRAY", "", 8192)
192640.GO-FASTER-1!PSQCKSRV.370: TRACE:at:    } tpalloc = 0x18158e8
192640.GO-FASTER-1!PSQCKSRV.370: TRACE:at:    { tpreturn(2, 0, 0x18158e8, 58,
0x0)
192640.GO-FASTER-1!PSQCKSRV.370: TRACE:at:    } tpreturn [long jump]
192640.GO-FASTER-1!PSQCKSRV.370: TRACE:at:  } tpservice
192640.GO-FASTER-1!PSAPPSRV.277: TRACE:at:  { tpservice({"PprLoad", 0x0,
0x17f7a30, 214, 0, 0, {942087529, 0, 40}})
192640.GO-FASTER-1!PSAPPSRV.277: TRACE:at:    { tpalloc("CARRAY", "", 8192)
192640.GO-FASTER-1!PSAPPSRV.277: TRACE:at:    } tpalloc = 0x1814bd8
192640.GO-FASTER-1!PSAPPSRV.277: TRACE:at:    { tpreturn(2, 0, 0x1814bd8, 2312,
0x0)
192640.GO-FASTER-1!PSAPPSRV.277: TRACE:at:    } tpreturn [long jump]
192640.GO-FASTER-1!PSAPPSRV.277: TRACE:at:  } tpservice
```

Each of the service messages in and out of the server is logged. The length of the message can also be seen, and hence it is possible to detect memory leaks.

The Tuxedo client side trace does not work with the PeopleTools client.

printclient (pclt) [-m machine][-u usrname][-c cltname]

This command prints information for the specified set of client processes. If no arguments or defaults are set, then information on all clients is printed. The options or defaults can be used to restrict the information to any combination of machine, user name, or client name.

```
> pclt
LMID          User Name       Client Name    Time    Status  Bgn/Cmmt/Abrt
--------------- --------------- --------------- -------- ------- -------------
simple          NT              WSH             1:17:09 IDLE    0/0/0
simple          NT              JSH             1:17:08 IDLE    0/0/0
simple          NT              WSH             1:17:06 IDLE    0/0/0
simple          PS              GO-FASTER-1     0:00:03 IDLE/W  0/0/0
simple          NT              tmadmin         0:04:02 IDLE    0/0/0
```

Note that the WSL, JSL and tmadmin processes are clients of the Bulletin Board as well as PeopleTools windows client (in *italics*) connected as operator PS from machine GO-FASTER-1.

printqueue (pq) [address]

Prints queue lists information for all application and administrative servers. The default is to print information about all queues. The queue command line or default can be used to restrict information to a specific queue. Output includes the server name and the name of the machine on which the queues reside.

```
> pq
Prog Name      Queue Name  # Serve Wk Queued  # Queued  Ave. Len    Machine
---------      ------------------ ---------  --------  --------    -------
PSAPPSRV.exe   APPQ2             1         0         0       0.0     simple
JSL.exe        00095.00200       1         0         0       0.0     simple
BBL.exe        33998             1         0         0       0.0     simple
WSL.exe        00001.00120       1         0         0       0.0     simple
PSQCKSRV.exe   QCKQ2             1         0         0       0.0     simple
WSL.exe        00001.00130       1         0         0       0.0     simple
JREPSVR.exe    00094.00250       1         0         0       0.0     simple
PSAPPSRV.exe   APPQ              1         0         0       0.0     simple
TMSYSEVT.exe   00001.00200       1         0         0       0.0     simple
PSAUTH.exe     AUTHQ             1         0         0       0.0     simple
PSQCKSRV.exe   QCKQ              1         0         0       0.0     simple
PSQRYSRV.exe   QRYQ              1         0         0       0.0     simple
PSAPISRV.exe   APIQ              1         0         0       0.0     simple
PSSAMSRV.exe   SAMQ              1         0         0       0.0     simple
```

If the queue does not have an explicitly defined name, the derived name is the combination group and first server ID.

The average length column is only populated if load balancing is enabled.

printserver (psr) [-m machine][-g groupname][-I srvid][-a {0|1|2}][-q address][-s service]

Prints information for application and administrative servers. The –q, -m, -g and -i options can be used to restrict the information to any combination of queue address, machine, group or server.

```
> psr
Prog Name      Queue Name  Grp Name      ID RqDone Load Done Current Service
---------      ----------  --------      -- ------ --------- ---------------
BBL.exe        33998       simple         0      7       350 (  IDLE )
PSAUTH.exe     AUTHQ       BASE           1      2        90 (  IDLE )
JREPSVR.exe    00094.00250 JREPGRP      250      0         0 (  IDLE )
PSAPPSRV.exe   APPQ        APPSRV        10      0         0 (  IDLE )
PSAPPSRV.exe   APPQ2       APPSRV        20      0         0 (  IDLE )
PSSAMSRV.exe   SAMQ        APPSRV       100      0         0 (  IDLE )
TMSYSEVT.exe   00001.00200 BASE         200      6       300 (  IDLE )
JSL.exe        00095.00200 JSLGRP       200      0         0 (  IDLE )
WSL.exe        00001.00120 BASE         120      0         0 (  IDLE )
PSQCKSRV.exe   QCKQ        APPSRV        50      1         3 (  IDLE )
WSL.exe        00001.00130 BASE         130      0         0 (  IDLE )
PSQCKSRV.exe   QCKQ2       APPSRV        60     18       166 (  IDLE )
PSAPISRV.exe   APIQ        APPSRV       150      0         0 (  IDLE )
PSQRYSRV.exe   QRYQ        APPSRV        70      0         0 (  IDLE )
```

Each server running in the machine is reported. The number of requests and the total load done by each server process is reported.

printservice (psc) [-m machine] [-g groupname] [-i srvid] [-q qaddress] [-s service] [-a {0|1|2}]

Prints information for application and administrative services. The –q, -m, -g, -i and -s options can be used to restrict the information to any combination of queue address, machine, group, server or service. The -a option allows you to select the class of service; -a0 limits the display to application services, -a1 selects application services plus system services callable by an application, -a2 selects both of those plus system services callable by BEA TUXEDO.

```
> psc -i 50
Service Name Routine Name Prog Name  Grp Name  ID    Machine  # Done Status
------------ ------------ ---------  --------  --    -------  ------ ------
WamStartIns+ WamStartIns+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
WamChgInstS+ WamChgInstS+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
SamGetParms+ SamGetParms+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
StmGetTimeO+ StmGetTimeO+ PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
StmChgPswd   StmChgPswd   PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
SqlRequest   SqlRequest   PSQCKSRV.+ APPSRV    50     simple       1 AVAIL
RamList      RamList      PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
MgrClear     MgrClear     PSQCKSRV.+ APPSRV    50     simple       0 AVAIL
```

Each of the services available on each of the server processes are reported. Note that the service name is truncated to 11 characters. The only place where the full name is visible is the Tuxedo administrative Java applet.

serverparams (srp)

Print the parameters associated with the server specified by *groupname* and *srvid* for a group.

```
> srp -g APPSRV -i 50
         Prog Name: d:\ps\hr75\bin\server\winx86\PSQCKSRV.exe
        Queue Name: QCKQ
    Server Options: RESTARTABLE
    Max # Restarts: 3
   Restart Command: (restartsrv)
      Grace Period: 0 hour 1 mins
          Group ID: 99
         Server ID: 50
        Machine ID: simple
```

serviceparams (scp)

Print the parameters associated with the service specified by *groupname*, *srvid* and *service*.

```
> scp -g APPSRV -i 70 -s SqlQuery
    Service Name: SqlQuery
   Function Name: SqlRequest
            Load: 1630
        Priority: 50
         Address: 0
```

tmshutdown

When an application server domain is shutdown by *psadmin*, the *tmshutdowm* command is issued. This command has the same parameters as the shutdown (stop) command within *tmadmin*.

tmshutdown can be a useful way to force the application server to shutdown. If , for example, a cold database backup is to be performed. If any user is still connected to the domain, that can stop a normal shutdown or generate a prompt that must be responded to.

```
D:\ps\hr75\appserv>tmshutdown -k KILL -y -c
Shutting down all admin and server processes in
D:\ps\hr75\appserv\h75d_tuned\PSTUXCFG

Shutting down server processes ...

        Server Id = 130 Group Id = BASE Machine = simple:       SIGKILL
        Server Id = 200 Group Id = JSLGRP Machine = simple:     SIGKILL
        Server Id = 120 Group Id = BASE Machine = simple:       SIGKILL

Shutting down admin processes ...

        Server Id = 0 Group Id = simple Machine = simple:       shutdown
succeeded
1 process stopped.
```

tmunloadcf

psappsrv.ubb is compiled to *Pstuxcfg* by *tmloadcf*. *tmunloadcf* reverses the process.

```
set TUXCONFIG=D:\ps\hr75\appserv\h75d_tuned\PSTUXCFG
tmunloadcf  > tmunloadcf.ubb
```

It can be useful to use this utility to generate a Tuxedo configuration file that includes all the configuration variables, including the default values.

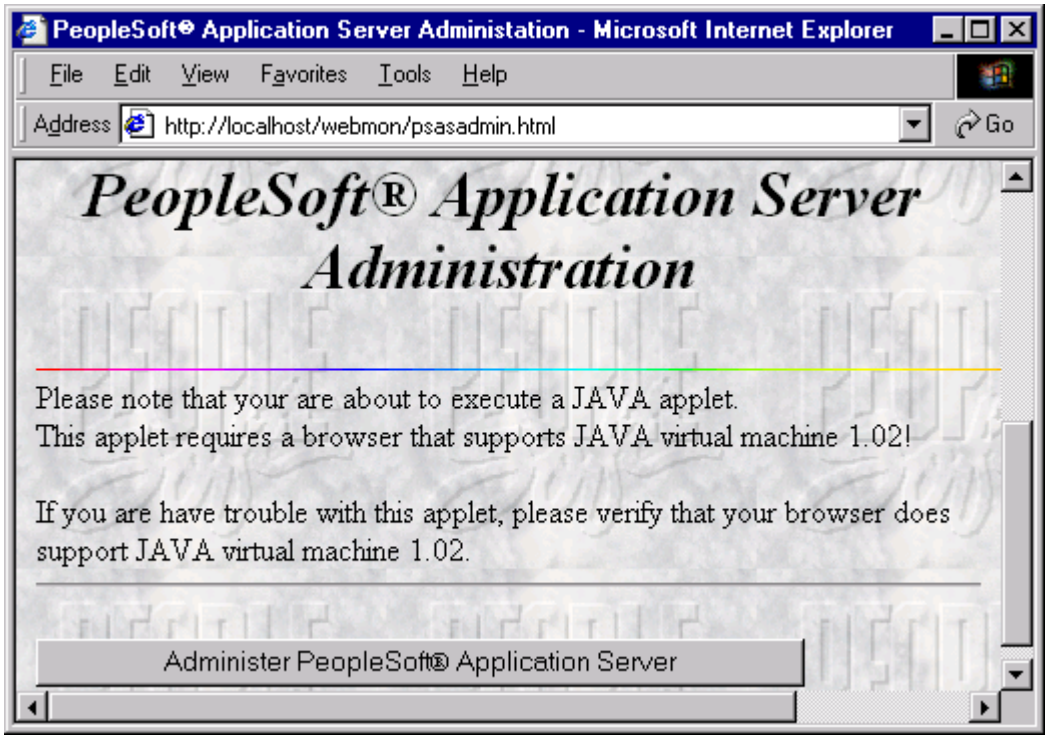
```
…
*GROUPS
"BASE"        LMID="simple"  GRPNO=1
      TMSCOUNT=3
"APPSRV"      LMID="simple"  GRPNO=99
      TMSCOUNT=3
"JREPGRP"     LMID="simple"  GRPNO=94
      TMSCOUNT=3
"JSLGRP"      LMID="simple"  GRPNO=95
      TMSCOUNT=3

*NETGROUPS

*SERVERS
"PSAUTH"      SRVGRP="BASE"  SRVID=1
      CLOPT="-r -e LOGS/AUTHQ.stderr -A -- -C psappsrv.cfg -D H75D -S PSAUTH"
      SEQUENCE=1
      RQADDR="AUTHQ"
      RQPERM=0660     REPLYQ=Y        RPPERM=0660     MIN=1   MAX=2   CONV=N
      SYSTEM_ACCESS=FASTPATH
      MAXGEN=3 GRACE=60        RESTART=Y
…
*SERVICES
"GetCertificate"
      LOAD=40  PRIO=50
      BUFTYPE="ALL"
      TRANTIME=30
      AUTOTRAN=N
      SVCTIMEOUT=300
…
```

CHAPTER 5

WEB ADMINISTRATION

PeopleSoft Sample Web Site

The sample PeopleSoft web site, that provides a demonstration of the Java web client, also includes a link to a pack that will initiate the Tuxedo Administrative Java Applet.



http://…/webmon/psasadmin.html, %PS_HOME%/appserv/webmon/psasadmin.html

The push button is simple coded in HTML as:

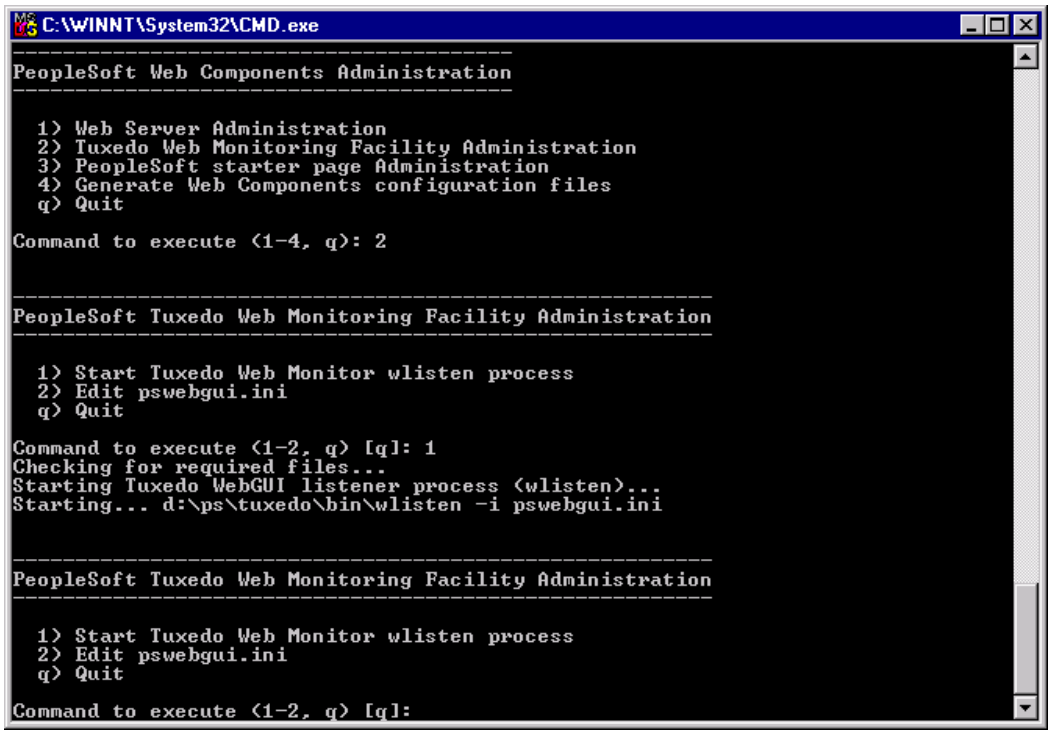
```
<FORM ACTION="/cgi-bin/tuxadm.exe">
<INPUT TYPE=HIDDEN NAME="TUXDIR" VALUE="d:\ps\tuxedo">
<INPUT TYPE=HIDDEN NAME="INIFILE" VALUE="D:\ps\hr75\appserv\pswebgui.ini">
<INPUT TYPE=SUBMIT VALUE="Administer PeopleSoft&reg Application Server">
</FORM>
```

extract from http://l.../webmon/psasadmin.html, %PS_HOME%/appserv/webmon/psasadmin.html

The hard coded literal value are inserted by psadmin when the web pages are generated. The template for psasadmin.html is psasadmin.htx.

Web Gui Listener (wlisten)

Before attempting to start the Tuxedo Administrative Applet the wlisten process must be started. This can be started from the command line, or from the psadmin utility.



Starting wlisten in psadmin

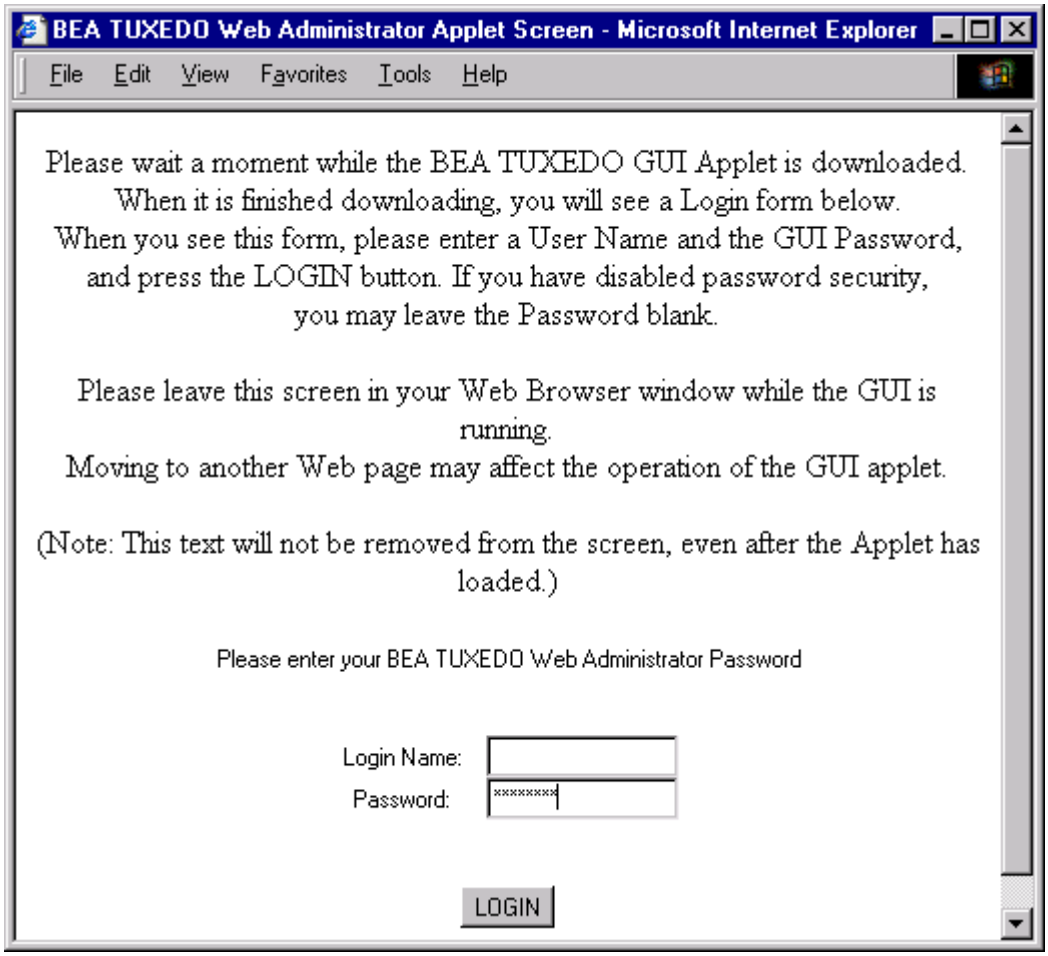
The pswebgui.ini lists the domains that can be viewed with the administrative applet. It also specifies the location for the listener program.

```
Web GUI initialization file.
# Created Sun Oct 24 09:04:34 1999 by TUXEDO System installation program.
#
TUXDIR=d:\ps\tuxedo
INIFILE=D:\ps\hr75\appserv\pswebgui.ini
NADDR=//GO-FASTER-1:4003
CODEBASE=/java
DOCBASE=http://GO-FASTER-1:80/doc
SNAPDIR=D:\ps\hr75\appserv\webmon
SNAPBASE=/java/snapshot
#
# In order to configure one or more domains as part of the Web GUI pull-down
# menu, add lines to this file of the form DOMAIN=domainname;tuxconfig
DOMAIN=h75d_mp;D:\ps\hr75\appserv\h75d_mp\PSTUXCFG
DOMAIN=h75d_pt754;D:\ps\hr75\appserv\h75d_pt754\PSTUXCFG
DOMAIN=h75d_pt756;D:\ps\hr75\appserv\h75d_pt756\PSTUXCFG
DOMAIN=h75d_smp;D:\ps\hr75\appserv\h75d_smp\PSTUXCFG
DOMAIN=h75d_tuned;D:\ps\hr75\appserv\h75d_tuned\PSTUXCFG
FOLDERS=YNNNYNNNNNNNNNNNNNN;YNNNYYNNYN;YNY;Y;Y;YNN;Y;Y;Y;Y;;;
DETAILS=DETVIEW
SORT=SORTNAME
```

%PS_HOME%/appserv/pswebgui.ini

If the first names domain in pswebgui.ini is not available, then the applet may not open the specified domain. The domain name and configuration file name may need to be entered manually.

Tuxedo Web Administrative Applet



http://localhost/cgi-bin/tuxadm.exe?TUXDIR=d%3A%5Cps%5Ctuxedo&INIFILE=D%3A%5Cps%5Chr75%5Cappserv%5Cpswebgui.ini

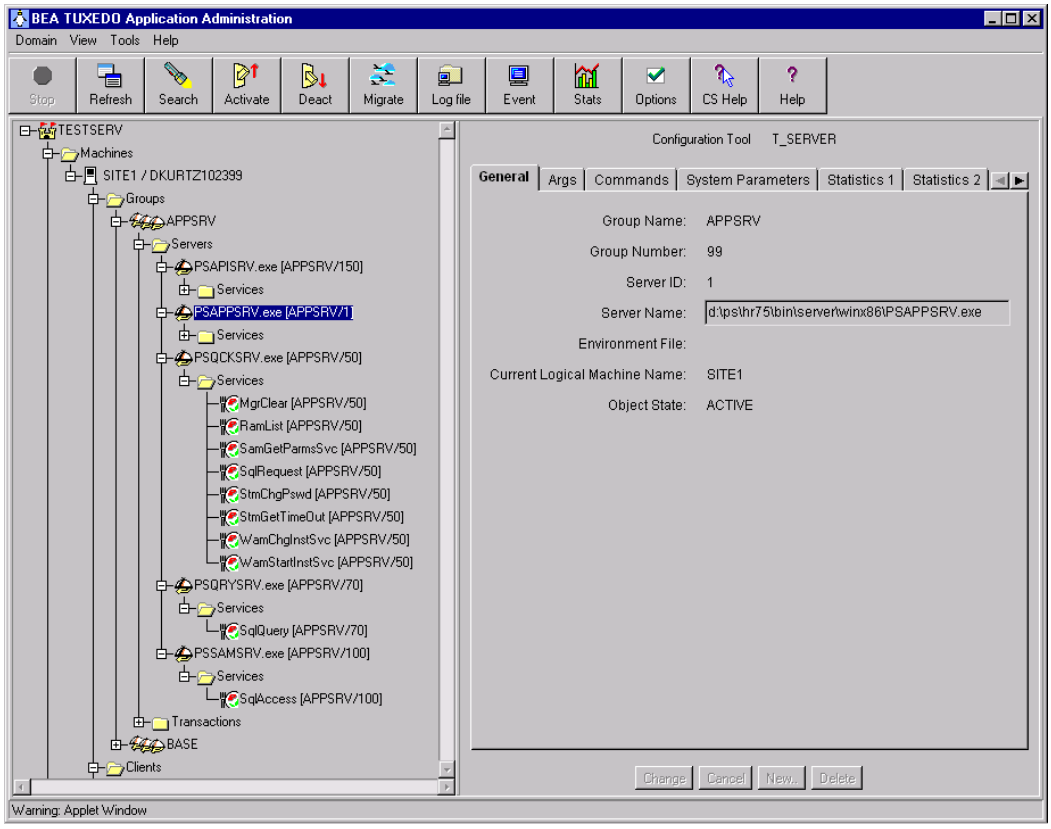
This login screen is then presented. The login name is not used to log into the administrative applet. If anything is entered it is simply ignored. However, the password must correspond to the password in the file *%TUXDIR%\udataobj\tlisten.pw*.

```
tuxedo
▯
```

%TUXDIR%\udataobj\tlisten.pw

The password is in plain text. So the file should not be readable to casual users of the application server, if there are any.

This is a screenshot of the administrative applet.



Tuxedo Application Administration

Tuxedo Manual: BEA Tuxedo Web-based GUI	
V6.4	tuxedo/tux64/fgui.htm
V6.5	tuxedo/tux65/admingd/tools.htm

The structure of the domain is shown as a tree. The Domain ID, *TESTSERV*, is broken down into machines. The logical machine *SITE1* is running on the physical machine *GO-FASTER-1*. The machine is split into groups *APPSERV* and *BASE*. This domain was not configured for web clients, so there are no other groups.

Each server process is shown individually, the program name, followed by the group and server ID in brackets. The services advertised on each server are shown. The eight services advertised on PSQCKSRV can be seen.

The right hand window shows information corresponding to the element within the tree that is selected. It is possible to see command line parameters, the number of requests that a server has completed, or the number of requests for a particular service that has been completed.

The tuxedo broker process is not configured in the Tuxedo configuration as shipped by PeopleSoft because PeopleTools makes no use direct use of it. However it does improve the logging information that can be seen within the administrative applet. All that is required is that the two lines be uncommented in *psappserv.ubx*, and the configuration file regenerated.

```
#
# Tuxedo System Event Server
#
TMSYSEVT        SRVGRP=BASE
                SRVID=200
```

extract from psappsrv.ubx

Tuxedo Manual: Event Broker/Monitor	
V6.5	tuxedo/tux65/admingd/events.htm - 997149

Icons

Having produced a convoluted product name that made form a meaningful acronym, the icons in the Windows and Java GUI followed on effortlessly.

As explained in the Introduction (page 7), in America, generally only restaurant waiters wear a Tuxedo. This theme runs through a number of icons used in the applet.

A group of platters indicates a server group.

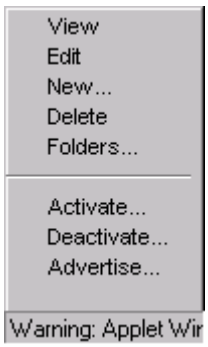
An empty platter without a cover indicates a server that is down.

A platter with a cover indicates an active server

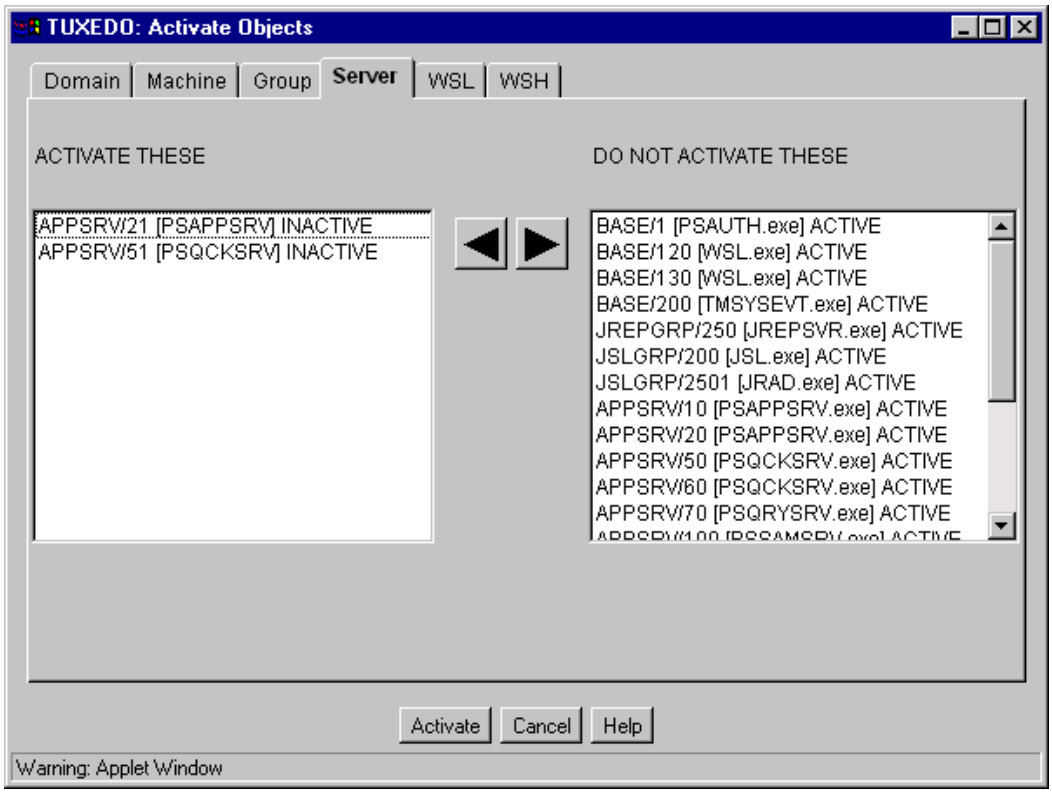
The services are shown as a plate with a knife and fork.

Starting Up/Shutting Down Servers

Right clicking with the mouse on a server, or server group, brings up the following menu.

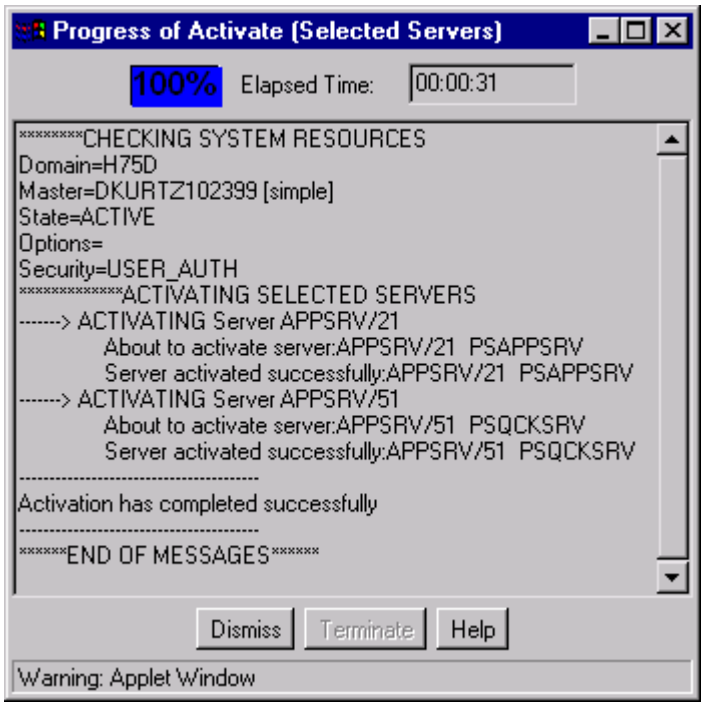


On selecting Activate the follow screen comes up.



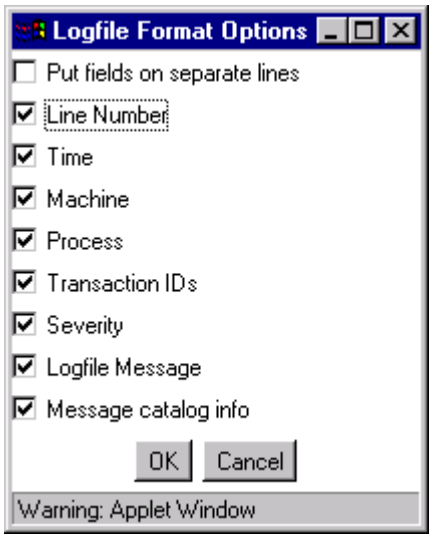
All the server processes listed in the left hand window will be started by pressing the Activate button at the bottom of the window.

The following screen shows the servers as they are started.



Tuxedo Log

If the event broker is running then it is also possible to view the TUXLOG file via the web applet.



CHAPTER 6

APPENDIX

Vanilla PT7.56 Application Server Domain Configuration Files

psappsrv.cfg

```
[Startup]
;============================================================================
; Database Signon settings
;============================================================================
DBName=h75d
DBType=ORACLE
OprId=PS
OprPswd=PS
ConnectId=
ConnectPswd=
ServerName=


[Workstation Listener]
;============================================================================
; Settings for Workstation Listener
;============================================================================
;Address Note: Can be either Machine Name or IP address.
;Address Note: %PS_MACH% will be replaced with THIS machine's name
```

```
Address=localhost
Port=7000
Encryption=0
Min Handlers=1
Max Handlers=1
Max Clients per Handler=60
Client Cleanup Timeout=60
Init Timeout=5
Tuxedo Compression Threshold=5000


[JOLT Listener]
;===================================================================
; Settings for JOLT Listener
;===================================================================
;Address Note: Can be either Machine Name or IP address.
;Address Note: %PS_MACH% will be replaced with THIS machine's name
Address=%PS_MACH%
Port=9000
Encryption=0
Min Handlers=1
Max Handlers=1
Max Clients per Handler=60
Client Cleanup Timeout=60
Init Timeout=5
Client Connection Mode=ANY
Jolt Compression Threshold=5000


[JOLT Relay Adapter]
;===================================================================
; Settings for JOLT Relay Adapter (JRAD)
;===================================================================
;Listener Address Note: Can be either Machine Name or IP address.
;Listener Address Note: %PS_MACH% will be replaced with THIS machine's name
Listener Address=%PS_MACH%
Listener Port=9100


[Domain Settings]
;===================================================================
; General settings for this Application Server.
;===================================================================

;-------------------------------------------------------------------
; TUXEDO Domain for this Application Server (8 characters or less).
;
Domain ID=TESTSERV

;-------------------------------------------------------------------
; Additional directories for the Application Server's PATH environment
; variable.  This should include the location of the database DLL's.
```

```
;
Add to PATH=d:\orant80\bin

;-----------------------------------------------------------------------
; Mark servers restartable or not.
;
Restartable=Y

;-----------------------------------------------------------------------
; Location of TUXEDO and PeopleSoft Application Server log files for this
; Application Server.
;
;Log Directory=%PS_SERVDIR%\LOGS

;-----------------------------------------------------------------------
; Logging detail level
;
; Level     Type of information
; -----     -------------------
;  -100     - Suppress logging
;  -1       - Protocol, memory errors
;   0       - Status information
;   1       - General errors
;   2       - Warnings
;   3       - Tracing Level 1 (default)
;   4       - Tracing Level 2
;   5       - Tracing Level 3
LogFence=3

;-----------------------------------------------------------------------
; IMPORTANT NOTE: THIS SETTINGS IS NOT APPLICABLE FOR THE NT PLATFORM!
;
;
Character Set=latin1


[Trace]
;=======================================================================
; SQL and PeopleCode Trace flags
;=======================================================================

;-----------------------------------------------------------------------
; SQL Tracing Bitfield
;
; Bit       Type of tracing
; ---       ---------------
; 1         - SQL statements
; 2         - SQL statement variables
; 4         - SQL connect, disconnect, commit and rollback
; 8         - Row Fetch (indicates that it occurred, not data)
; 16        - All other API calls except ssb
; 32        - Set Select Buffers (identifies the attributes of columns
```

```
;             to be selected).
; 64        - Database API specific calls
; 128       - COBOL statement timings
; 256       - Sybase Bind information
; 512       - Sybase Fetch information
; 4096      - Manager information
; 8192      - Message Agent information
TraceSql=0
TraceSqlMask=12319

;------------------------------------------------------------------------
; PeopleCode Tracing Bitfield
;
; Bit       Type of tracing
; ---       ---------------
; 1         - Trace entire program
; 2         - List the program
; 4         - Show assignments to variables
; 8         - Show fetched values
; 16        - Show stack
; 64        - Trace start of programs
; 128       - Trace external function calls
; 256       - Trace internal function calls
; 512       - Show parameter values
; 1024      - Show function return value
; 2048      - Trace each statement in program
TracePC=0
TracePCMask=0


[Cache Settings]
;========================================================================
; Settings for managed object caching
;========================================================================

;------------------------------------------------------------------------
; Server caching settings
;
; EnableServerCaching = 1 to enable caching, 0 to disable it
EnableServerCaching=1
; CacheBaseDir = the base cache directory
;CacheBaseDir=%PS_SERVDIR%\CACHE


[Database Options]
;========================================================================
; Database-specific configuration options
;========================================================================

DB2InputMessageSize=
DB2OutputMessageSize=
SybasePacketSize=
```

```
; Please see Chapter "Tuning and Administration", in
; Oracle Installation and Administration Guide for details
UseLocalOracleDB=0
EnableDBMonitoring=0


[RemoteCall]
;============================================================================
; Settings for RemoteCall
;============================================================================

;----------------------------------------------------------------------------
; RemoteCall child process output redirection
;
; If this parameter is non-zero, the child process output is saved to
; <Domain Settings\Log Directory>\<program>_<oprid>.out, and any error
; output is saved to <program>_<oprid>.err.
; By default, the output is not saved
;
RCCBL Redirect=0

;----------------------------------------------------------------------------
; Location of COBOL programs
; By default, RemoteCall looks for the COBOL programs in %PS_HOME%\cblbin
;
;RCCBL PRDBIN=%PS_HOME%\cblbin


[PSAUTH]
;============================================================================
; Settings for PSAUTH
;============================================================================

;----------------------------------------------------------------------------
; UBBGEN settings
Min Instances=1
Max Instances=1
; Note: Spawn is not valid for this (MSSQ) server!

;----------------------------------------------------------------------------
; Number of services after which PSAUTH will automatically restart.
; If the recycle count is set to zero, PSAUTH will never be recycled.
; The default value is zero.
Recycle Count=100000

;----------------------------------------------------------------------------
; Number of consecutive service failures after which PSAUTH will
; automatically restart.
; If this is set to zero, PSAUTH will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=0
```

```
;------------------------------------------------------------------------
; Use the database for additional signon authentication.
; The default value is zero.
Validate Signon with Database=0


[PSAPPSRV]
;========================================================================
; Settings for PSAPPSRV
;========================================================================

;------------------------------------------------------------------------
; UBBGEN settings
Min Instances=1
Max Instances=1
; Note: Spawn may only be 1 if Max Instances is greater than Min Instances!
Spawn=0
Service Timeout=300

;------------------------------------------------------------------------
; Number of services after which PSAPPSRV will automatically restart.
; If the recycle count is set to zero, PSAPPSRV will never be recycled.
; The default value is zero.
Recycle Count=100000

;------------------------------------------------------------------------
; Number of consecutive service failures after which PSAPPSRV will
; automatically restart.
; If this is set to zero, PSAPPSRV will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=2

; Max Fetch Size -- max result set size in KB for a SELECT query
; Default is 5000KB. Use 0 for no limit.
Max Fetch Size=5000


[PSSAMSRV]
;========================================================================
; Settings for PSSAMSRV
;========================================================================

;------------------------------------------------------------------------
; UBBGEN settings
Min Instances=1
Max Instances=1
; Note: Spawn is not valid for this (conversational) server!

Service Timeout=300

;------------------------------------------------------------------------
; Number of services after which PSSAMSRV will automatically restart.
```

```
; If the recycle count is set to zero, PSSAMSRV will never be recycled.
; The default value is zero.
Recycle Count=100000

;------------------------------------------------------------------------
; Number of consecutive service failures after which PSSAMSRV will
; automatically restart.
; If this is set to zero, PSSAMSRV will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=2

; Max Fetch Size -- max result set size in KB for a SELECT query
; Default is 32KB. Use 0 for no limit
Max Fetch Size=32


[PSQCKSRV]
;========================================================================
; Settings for PSQCKSRV
;========================================================================

;------------------------------------------------------------------------
; UBBGEN settings
Min Instances=1
Max Instances=1
; Note: Spawn may only be Y if Max Instances is greater than Min Instances!
Spawn=0
Service Timeout=300

;------------------------------------------------------------------------
; Number of services after which PSQCKSRV will automatically restart.
; If the recycle count is set to zero, PSQCKSRV will never be recycled.
; The default value is zero.
Recycle Count=100000

;------------------------------------------------------------------------
; Number of consecutive service failures after which PSQCKSRV will
; automatically restart.
; If this is set to zero, PSQCKSRV will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=2

; Max Fetch Size -- max result set size in KB for a SELECT query
; Default is 5000KB. Use 0 for no limit.
Max Fetch Size=5000

[PSQRYSRV]
;========================================================================
; Settings for PSQRYSRV
;========================================================================

;------------------------------------------------------------------------
```

```
; UBBGEN settings
Min Instances=1
Max Instances=1
; Note: Spawn may only be 1 if Max Instances is greater than Min Instances!
Spawn=0
Service Timeout=300

;---------------------------------------------------------------------
; Number of services after which PSQRYSRV will automatically restart.
; If the recycle count is set to zero, PSQRYSRV will never be recycled.
; The default value is zero.
Recycle Count=100000

;---------------------------------------------------------------------
; Number of consecutive service failures after which PSQRYSRV will
; automatically restart.
; If this is set to zero, PSQRYSRV will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=2

; Max Fetch Size -- max result set size in KB for a SELECT query
; Default is 10000KB. Use 0 for no limit.
Max Fetch Size=10000
;Use dirty-read(uncommitted read) on DB2/MVS for PSQRYSRV only
Use dirty-read on DB2/MVS=0


[PSAPISRV]
;=====================================================================
; Settings for PSAPISRV
;=====================================================================

;---------------------------------------------------------------------
; UBBGEN settings
Min Instances=1
Max Instances=1
; Note: Spawn may only be 1 if Max Instances is greater than Min Instances!
Spawn=0
Service Timeout=300

;---------------------------------------------------------------------
; Number of services after which PSAPISRV will automatically restart.
; If the recycle count is set to zero, PSAPISRV will never be recycled.
; The default value is zero.
Recycle Count=100000

;---------------------------------------------------------------------
; Number of consecutive service failures after which PSAPISRV will
; automatically restart.
; If this is set to zero, PSAPISRV will never be recycled.
; The default value is zero.
Allowed Consec Service Failures=0
```

```
[SMTP Settings]
;============================================================================
; Settings for SMTP mail
;============================================================================

SMTPServer=
SMTPPort=25
SMTPServer1=
SMTPPort1=0
SMTPSender=PeopleSoft@peoplesoft.com
SMTPSourceMachine=
```

psappsrv.cfg

psappsrv.val

```
#----------------------------------------------------------------------
#
# Confidentiality Information:
#
# This module is the confidential and proprietary information of
# PeopleSoft, Inc.; it is not to be copied, reproduced, or transmitted
# in any form, by any means, in whole or in part, nor is it to be used
# for any purpose other than that for which it is expressly provided
# without the written permission of PeopleSoft.
#
# Copyright (c) 1988-1999 PeopleSoft, Inc.  All Rights Reserved.
#
#----------------------------------------------------------------------
#
# SourceSafe Information:
#
# \$Header::                                                          \$
#
#----------------------------------------------------------------------

#
# Default PeopleSoft Application Server validation file for the Windows NT
platform.
#
# Note: This is the REQUIRED format for this file!
#
#       Element={type}(criteria)
#       Three types are allowed: int, path and string.
#
#       Format for each is as follows:
#              Element={int}(range)
#              Element={path}(filename) Note: filename is relative to the path
entered or listed in the cfg file
#              Element={string, MAXIMUM length}(optional criteria list)
#
#       If type is string, there MUST be a MAXIMUM length associated as in:
{string,8}
#       This means that a string entered MUST be less than or equal to 8
characters!
#       There can also be a (criteria) list associated with a string type.
#       If type is other than string, there MUST be criteria.
#       There should be no extraneous characters or spaces!
#
#
[Startup]
DBName={string,8}
DBType={string,8}(DB2,DB2400,DB2ODBC,DB2UNIX,INFORMIX,MICROSFT,ORACLE,SQLBASE,SY
BASE)
OprId={string,8}
OprPswd={string,8}
#
```

```
[Workstation Listener]
Port={int}(1025-65536)
Encryption={string,3}(0,40,128)
#
[JOLT Listener]
Client Connection Mode={string,9}(RETAINED,RECONNECT,ANY)
#
[Domain Settings]
Character Set={nodisplay}
```

psappsrv.val

psappsrv.ubx

```
##############################################################################
#------------------------------------------------------------------------
#
# Confidentiality Information:
#
# This module is the confidential and proprietary information of
# PeopleSoft, Inc.; it is not to be copied, reproduced, or transmitted
# in any form, by any means, in whole or in part, nor is it to be used
# for any purpose other than that for which it is expressly provided
# without the written permission of PeopleSoft.
#
# Copyright (c) 1988-1999 PeopleSoft, Inc.  All Rights Reserved.
#
#------------------------------------------------------------------------
#
# SourceSafe Information:
#
# $Logfile:: /PT75/APPSERV/psappsrv.ubx                                  $
# $Revision:: 83                                                         $
# $Author:: Lzhuang                                                      $
# $Date:: 6/21/99 1:53p                                                  $
##############################################################################

*PS_DEFINES
# This section defines the variables used in the ubb file.  The ubb config
# file generation utility, ubbgen, processes this section and replaces them
# with their values in the rest of the file.  The resulting ubb file can
# be used to generate the tuxconfig file directly, or can be modified
# further with any text editor.
#
# USAGE:
#   ubbgen -t psappsrv.ubx -c appsrv.cfg -o psappsrv.ubb -q y
#   ubbgen will open the file psappsrv.ubx, process it, and produce the file
#   psappsrv.ubb.
#   The -q y option allows for command line processing when ubbgen is called
from psadmin. (Quiet mode)
#
# FILE FORMAT:
#   Substitution variables are enclosed in braces ('{', '}'), and
#   must be defined in the PS_DEFINE block (delimited with the *PS_DEFINES
#   and *END tokens).  There are four types of substitution variables:
#       - Environment variables.  The name of the variable starts with '$'.
#         ubbgen gets the value of these variables with getenv.
#         ($TUXCONFIG)
#       - Config file variables.  These variables are read from the
#         configuration file passed in to ubbgen (appsrv.cfg, by default).
#         The name of the variable starts with '$', followed by the name of
#         the sub-section in the registry, then a slash ('\'), and then the
#         name of the key.
#         ($Domain Settings\Application Server Home)
#       - Special variables.  ubbgen recognizes these variables by name, and
```

```
#         performs special handling to get their values.
#         (DOMAINID, IPCKEY, MACH, WSNADDR, UID GID)
#       - Prompted variables.  Any variable which does not fall into the
#         catergories above is assumed to be of this type.  ubbgen prompts
#         the user to specify the value of the variable.  The line of text
#         immediately following the variable definition is used as the prompt.
#         (APPDIR, TUXDIR)
#
#   Any text following the pound symbol ('#') is interpreted as a comment,
#   and is not searched for substitution variables.
#
#   The PS_DEFINES block is not copied into the output file.

{QUICKSRV} Move quick PSAPPSRV services into a second server (PSQCKSRV) (y/n)?
[y]:
{QUERYSRV} Move long-running SqlQuery service into a second server (PSQRYSRV)
(y/n)? [y]:
{JOLT} Do you want JOLT configured (y/n)? [n]:
{JRAD} Do you want JRAD configured (y/n)? [n]:

*END

############################################################################
#
# This is a skeletal TUXEDO configuration file - "psappsrv.ubb" designed
# to be used for PeopleTools 7.5 app server and the Remote Call mechanism.
# To configure additional resources, machines, servers, services, etc.
# please refer to "ubbconfig" in section 5 of the TUXEDO System Reference
# Manual.
#
############################################################################

*RESOURCES
IPCKEY          {IPCKEY}        # ( 32768 < IPCKEY < 262143 )
MASTER          SITE1
DOMAINID        {$Domain Settings\Domain ID}
MODEL           SHM
LDBAL           N
#
MAXMACHINES     256             # min, default=256
MAXGROUPS       100             # min, default=100
{MAXSERVERS}
{MAXSERVICES}
#
MAXACLGROUPS    1               # def=16K (incl only to save BB space)
MAXGTT          0               # def=100 (----------   " ----------)
MAXCONV         50              # def=10  (----------   " ----------)
#
SECURITY        USER_AUTH
AUTHSVC         PSAUTH
SYSTEM_ACCESS   FASTPATH
#
```

```
PERM            0660            # can override on a per m/c basis
#
SCANUNIT        5               # Time in seconds between scans by the BBL
                                # for old transactions and timed-out blocking
                                # calls.

SANITYSCAN      2               # The BBL indicates to the DBBL that it
                                # is alive every (SCANUNIT * SANITYSCAN
                                # seconds.

DBBLWAIT        2               # (SCANUNIT * DBBLWAIT) is the time in
                                # seconds after which the DBBL will time
                                # out an unresponsive BBL.

BBLQUERY        30              # (SCANUNIT * BBLQUERY) is the frequency
                                # of status verification by the DBBL.
                                # If the DBBL has not received an "I'm OK"
                                # message from a BBL during this period,
                                # the DBBL will send a status query message
                                # to the BBL.  If no response is received,
                                # the BBL's node is partitioned.

# DEBUG
BLOCKTIME       6000            # (SCANUNIT * BLOCKTIME) is the time in
                                # seconds after which a blocking call
                                # will time out from the client.
#
CMTRET          COMPLETE
NOTIFY          DIPIN
USIGNAL         SIGUSR2

# ------------------------------------------------------------------------

*MACHINES
"{MACH}" LMID="SITE1"                           # Machine name must be upper
case
        TUXDIR="{$TUXDIR}"                      # Paths cannot end in '\'
        APPDIR="{$PS_SERVDIR}"                  # include the database path
        TUXCONFIG="{$TUXCONFIG}"
        ULOGPFX="{LOGDIR}{FS}TUXLOG"
        ENVFILE="{$PS_SERVDIR}{FS}{ENVFILE}"
        UID={UID}                               # Has to be 0 at this time.
        GID={GID}                               # Has to be 0 at this time.
{WINDOWS}
        TYPE="i386NT"
{WINDOWS}
        NETLOAD=0                               # We are not using multiple
machines.
        {MAXWSCLIENTS}
        {MAXACCESSERS}

# ------------------------------------------------------------------------
```

```
*GROUPS

#
# Tuxedo Groups
# For application group numbers for new machines (LMIDs)
# use group numbers 101-199;  201-299;  etc.
#
DEFAULT:
        LMID=SITE1

BASE    GRPNO=1

APPSRV  GRPNO=99

{JOLT}
#
# JOLT Groups
#
JREPGRP LMID=SITE1     GRPNO=94
JSLGRP  LMID=SITE1     GRPNO=95
{JOLT}

# ------------------------------------------------------------------------

*SERVERS

DEFAULT:
        CLOPT="-A"       # Advertise all services.
        REPLYQ=N         # Reply queue not needed for our simple setup.
        MAXGEN=3         # Max number of restarts in the grace period.
        GRACE=60         # Ten minutes grace period.
        RESTART={$Domain Settings\Restartable}
        SYSTEM_ACCESS=FASTPATH

#
# PeopleSoft Tuxedo Authentication Server
#
PSAUTH          SRVGRP=BASE
                MIN={$PSAUTH\Min Instances}
                MAX={$PSAUTH\Max Instances}
                SRVID=1
                RQADDR="AUTHQ"
                REPLYQ=Y
                CLOPT="-A -- -C {CFGFILE} -D {$Domain Settings\Domain ID} -S
PSAUTH"

#
# Workstation Listener
#  -I xx   Max time (seconds) for a client connect
#  -T xx   Max time (minutes) for a client to stay idle.
#  -m xx   Min number of workstation handlers
```

```
#  -M xx   Max number of workstation handlers
#  -x xxx  Multiplexing, the max number of clients per handler
#
#
WSL             SRVGRP=BASE
                SRVID=20
{WINDOWS}
                CLOPT="-A -- -n {$Workstation Listener\Address}:{$Workstation
Listener\Port} -z {$Workstation Listener\Encryption} -Z {$Workstation
Listener\Encryption} -I {$Workstation Listener\Init Timeout} {WSL Client Cleanup
Timeout} -m {$Workstation Listener\Min Handlers} -M {$Workstation Listener\Max
Handlers} -x {$Workstation Listener\Max Clients per Handler} -c {$Workstation
Listener\Tuxedo Compression Threshold}"
{WINDOWS}
{UNIX}
                CLOPT="-A -- -n {$Workstation Listener\Address}:{$Workstation
Listener\Port} -z {$Workstation Listener\Encryption} -Z {$Workstation
Listener\Encryption} -d {$PS_TUXDEV} -I {$Workstation Listener\Init Timeout}
{WSL Client Cleanup Timeout} -m {$Workstation Listener\Min Handlers} -M
{$Workstation Listener\Max Handlers} -x {$Workstation Listener\Max Clients per
Handler} -c {$Workstation Listener\Tuxedo Compression Threshold}"
{UNIX}

#
# Tuxedo System Event Server
#
#TMSYSEVT        SRVGRP=BASE
#                SRVID=200

#
# PeopleSoft Manager Application Server
#
PSAPPSRV        SRVGRP=APPSRV
                SRVID=1
                MIN={$PSAPPSRV\Min Instances}
                MAX={$PSAPPSRV\Max Instances}
                RQADDR="APPQ"
                REPLYQ=Y
{QUERYSRV}
                CLOPT="{$PSAPPSRV\Spawn Server} -A -- -C {CFGFILE} -D {$Domain
Settings\Domain ID} -S PSAPPSRV"
{QUERYSRV}

{!QUERYSRV}
                CLOPT="{$PSAPPSRV\Spawn Server} -A -sSqlQuery:SqlRequest -- -C
{CFGFILE} -D {$Domain Settings\Domain ID} -S PSAPPSRV"
{!QUERYSRV}

{QUICKSRV}
#
# PeopleSoft Manager Application Server
#
```

```
PSQCKSRV        SRVGRP=APPSRV
                SRVID=50
                MIN={$PSQCKSRV\Min Instances}
                MAX={$PSQCKSRV\Max Instances}
                RQADDR="QCKQ"
                REPLYQ=Y
{!QUERYSRV}
                CLOPT="{$PSQCKSRV\Spawn Server} -s
MgrClear,RamList,SqlRequest,StmChgPswd,StmGetTimeOut,SamGetParmsSvc,WamChgInstSv
c,WamStartInstSvc -sSqlQuery:SqlRequest -- -C {CFGFILE} -D {$Domain
Settings\Domain ID} -S PSQCKSRV"
{!QUERYSRV}

{QUERYSRV}
                CLOPT="{$PSQCKSRV\Spawn Server} -s
MgrClear,RamList,SqlRequest,StmChgPswd,StmGetTimeOut,SamGetParmsSvc,WamChgInstSv
c,WamStartInstSvc -- -C {CFGFILE} -D {$Domain Settings\Domain ID} -S PSQCKSRV"
{QUERYSRV}

{QUICKSRV}

{QUERYSRV}
#
# PeopleSoft Query Application Server
#
PSQRYSRV        SRVGRP=APPSRV
                SRVID=70
                MIN={$PSQRYSRV\Min Instances}
                MAX={$PSQRYSRV\Max Instances}
                RQADDR="QRYQ"
                REPLYQ=Y
                CLOPT="{$PSQRYSRV\Spawn Server} -sSqlQuery:SqlRequest -- -C
{CFGFILE} -D {$Domain Settings\Domain ID} -S PSQRYSRV"
{QUERYSRV}

#
# PeopleSoft SQL Access Application Server
#
PSSAMSRV        SRVGRP=APPSRV
                SRVID=100
                MIN={$PSSAMSRV\Min Instances}
                MAX={$PSSAMSRV\Max Instances}
                RQADDR="SAMQ"
                REPLYQ=Y
                CONV=Y
                CLOPT="-A -- -C {CFGFILE} -D {$Domain Settings\Domain ID} -S
PSSAMSRV"

#
# PeopleSoft API Application Server
#
PSAPISRV        SRVGRP=APPSRV
```

```
                SRVID=150
                MIN={$PSAPISRV\Min Instances}
                MAX={$PSAPISRV\Max Instances}
                RQADDR="APIQ"
                REPLYQ=Y
                CLOPT="{$PSAPISRV\Spawn Server} -A -- -C {CFGFILE} -D {$Domain
Settings\Domain ID} -S PSAPISRV"

{JOLT}
#
# JOLT Listener and Rep Server
#
JSL
                SRVGRP=JSLGRP
                SRVID=200
                CLOPT="-A -- {TUXDEV} -n {$JOLT Listener\Address}:{$JOLT
Listener\Port} -m {$JOLT Listener\Min Handlers} -M {$JOLT Listener\Max Handlers}
-I {$JOLT Listener\Init Timeout} -c {$JOLT Listener\Client Connection Mode} -x
{$JOLT Listener\Max Clients per Handler} {Jolt Encryption} {Jolt Client Cleanup
Timeout} -j {$JOLT Listener\Jolt Compression Threshold} -w JSH"

JREPSVR
                SRVGRP=JREPGRP
                SRVID=250
                CLOPT="-A -- -W -P {$PS_SERVDIR}{FS}jrepository"
{JOLT}

{JRAD}
#
# JOLT Internet Relay (Back End)
#
JRAD
                SRVGRP=JSLGRP
                SRVID=2501
                CLOPT="-A -- -l {$JOLT Relay Adapter\Listener Address}:{$JOLT
Relay Adapter\Listener Port} -c {$JOLT Listener\Address}:{$JOLT Listener\Port}"
{JRAD}

*SERVICES

GetCertificate  SRVGRP=BASE
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

MenuList        SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

MgrGetObject    SRVGRP=APPSRV
                LOAD=50 PRIO=50
```

```
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

MgrUpdObject    SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

MgrClear        SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

MgrPeek         SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

JavaMgrGetObj   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprLoad         SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprSave         SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprFullSave     SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprChangeValue  SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprFieldChange  SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprDefault      SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"
```

```
PprInsertRow    SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprDeleteRow    SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprSearchStart  SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprSearchSave   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprSecStart     SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprMItemSelPC   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PprPrePopupPC   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PsmSchedPrcs    SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

PsdGetListSvc   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

QdmGetListSvc   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

RamList         SRVGRP=APPSRV
```

```
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

RemoteCall      SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

SamGetParmsSvc  SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

SqlAccess       SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSSAMSRV\Service Timeout}
                BUFTYPE="ALL"

SqlRequest      SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

SqlQuery        SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQRYSRV\Service Timeout}
                BUFTYPE="ALL"

StmChgPswd      SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

StmGetTimeOut   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

StmGetExplain   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

OpnQryDescribe  SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
                BUFTYPE="ALL"

OpnQryExecute   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPPSRV\Service Timeout}
```

```
                BUFTYPE="ALL"

WamChgInstSvc   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

WamStartInstSvc SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSQCKSRV\Service Timeout}
                BUFTYPE="ALL"

MsgAPI          SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT={$PSAPISRV\Service Timeout}
                BUFTYPE="ALL"
# ------------------------------------------------------------------------
*PS_ENVFILE
{WINDOWS}
PATH={$PS_HOME}\bin\server\winx86;{$Domain Settings\Add to PATH}
{WINDOWS}
{UNIX}
LD_LIBRARY_PATH={$LD_LIBRARY_PATH}
LIBPATH={$LIBPATH}
SHLIB_PATH={$SHLIB_PATH}
COBPATH={$PS_HOME}/bin
PATH={$PS_HOME}/bin:{$Domain Settings\Add to PATH}
{UNIX}
INFORMIXSERVER={$Startup\ServerName}
{UNIX}
```

psappsrv.ubx

psappsrv.ubb

Note that the psappsrv.ubb file resembles the psappsrv.ubx but all the variables have been resolved to literal values.

```
#########################################################################
#
# This is a skeletal TUXEDO configuration file - "psappsrv.ubb" designed
# to be used for PeopleTools 7.5 app server and the Remote Call mechanism.
# To configure additional resources, machines, servers, services, etc.
# please refer to "ubbconfig" in section 5 of the TUXEDO System Reference
# Manual.
#
#########################################################################

*RESOURCES
IPCKEY          33796        # ( 32768 < IPCKEY < 262143 )
MASTER          SITE1
DOMAINID        TESTSERV
MODEL           SHM
LDBAL           N
#
MAXMACHINES     256             # min, default=256
MAXGROUPS       100             # min, default=100
MAXSERVERS       19

MAXSERVICES      87

#
MAXACLGROUPS    1               # def=16K (incl only to save BB space)
MAXGTT          0               # def=100 (----------   " ----------)
MAXCONV         50              # def=10  (----------   " ----------)
#
SECURITY        USER_AUTH
AUTHSVC         PSAUTH
SYSTEM_ACCESS   FASTPATH
#
PERM            0660            # can override on a per m/c basis
#
SCANUNIT        5               # Time in seconds between scans by the BBL
                                # for old transactions and timed-out blocking
                                # calls.

SANITYSCAN      2               # The BBL indicates to the DBBL that it
                                # is alive every (SCANUNIT * SANITYSCAN
                                # seconds.

DBBLWAIT        2               # (SCANUNIT * DBBLWAIT) is the time in
                                # seconds after which the DBBL will time
                                # out an unresponsive BBL.

BBLQUERY        30              # (SCANUNIT * BBLQUERY) is the frequency
                                # of status verification by the DBBL.
                                # If the DBBL has not received an "I'm OK"
                                # message from a BBL during this period,
                                # the DBBL will send a status query message
                                # to the BBL.  If no response is received,
```

```
                                # the BBL's node is partitioned.

# DEBUG
BLOCKTIME       6000            # (SCANUNIT * BLOCKTIME) is the time in
                                # seconds after which a blocking call
                                # will time out from the client.
#
CMTRET          COMPLETE
NOTIFY          DIPIN
USIGNAL         SIGUSR2

# ------------------------------------------------------------------

*MACHINES
"GO-FASTER-1" LMID="SITE1"                          # Machine name must be
upper case
        TUXDIR="d:\ps\Tuxedo"                      # Paths cannot end in '\'
        APPDIR="D:\ps\hr75\appserv\h75d_pt756"                 # include the
database path
        TUXCONFIG="D:\ps\hr75\appserv\h75d_pt756\PSTUXCFG"
        ULOGPFX="D:\ps\hr75\appserv\h75d_pt756\LOGS\TUXLOG"
        ENVFILE="D:\ps\hr75\appserv\h75d_pt756\psappsrv.env"
        UID=0                               # Has to be 0 at this time.
        GID=0                               # Has to be 0 at this time.
        TYPE="i386NT"
        NETLOAD=0                               # We are not using multiple
machines.
        MAXWSCLIENTS=120

        MAXACCESSERS=139


# ------------------------------------------------------------------

*GROUPS

#
# Tuxedo Groups
# For application group numbers for new machines (LMIDs)
# use group numbers 101-199;  201-299;  etc.
#
DEFAULT:
        LMID=SITE1

BASE    GRPNO=1

APPSRV  GRPNO=99

#
# JOLT Groups
#
JREPGRP LMID=SITE1     GRPNO=94
```

```
JSLGRP  LMID=SITE1     GRPNO=95

# ------------------------------------------------------------------------

*SERVERS

DEFAULT:
        CLOPT="-A"       # Advertise all services.
        REPLYQ=N         # Reply queue not needed for our simple setup.
        MAXGEN=3         # Max number of restarts in the grace period.
        GRACE=60         # Ten minutes grace period.
        RESTART=Y
        SYSTEM_ACCESS=FASTPATH

#
# PeopleSoft Tuxedo Authentication Server
#
PSAUTH          SRVGRP=BASE
                MIN=1
                MAX=1
                SRVID=1
                RQADDR="AUTHQ"
                REPLYQ=Y
                CLOPT="-A -- -C psappsrv.cfg -D TESTSERV -S PSAUTH"

#
# Workstation Listener
#  -I xx   Max time (seconds) for a client connect
#  -T xx   Max time (minutes) for a client to stay idle.
#  -m xx   Min number of workstation handlers
#  -M xx   Max number of workstation handlers
#  -x xxx  Multiplexing, the max number of clients per handler
#
#
WSL             SRVGRP=BASE
                SRVID=20
                CLOPT="-A -- -n //localhost:7000 -z 0 -Z 0 -I 5 -T 60 -m 1 -M 1
-x 60 -c 5000"

#
# Tuxedo System Event Server
#
#TMSYSEVT        SRVGRP=BASE
#                SRVID=200

#
# PeopleSoft Manager Application Server
#
PSAPPSRV        SRVGRP=APPSRV
                SRVID=1
                MIN=1
                MAX=1
```

```
                RQADDR="APPQ"
                REPLYQ=Y
                CLOPT=" -A -- -C psappsrv.cfg -D TESTSERV -S PSAPPSRV"


#
# PeopleSoft Manager Application Server
#
PSQCKSRV        SRVGRP=APPSRV
                SRVID=50
                MIN=1
                MAX=1
                RQADDR="QCKQ"
                REPLYQ=Y

                CLOPT=" -s
MgrClear,RamList,SqlRequest,StmChgPswd,StmGetTimeOut,SamGetParmsSvc,WamChgInstSv
c,WamStartInstSvc -- -C psappsrv.cfg -D TESTSERV -S PSQCKSRV"


#
# PeopleSoft Query Application Server
#
PSQRYSRV        SRVGRP=APPSRV
                SRVID=70
                MIN=1
                MAX=1
                RQADDR="QRYQ"
                REPLYQ=Y
                CLOPT=" -sSqlQuery:SqlRequest -- -C psappsrv.cfg -D TESTSERV -S
PSQRYSRV"

#
# PeopleSoft SQL Access Application Server
#
PSSAMSRV        SRVGRP=APPSRV
                SRVID=100
                MIN=1
                MAX=1
                RQADDR="SAMQ"
                REPLYQ=Y
                CONV=Y
                CLOPT="-A -- -C psappsrv.cfg -D TESTSERV -S PSSAMSRV"

#
# PeopleSoft API Application Server
#
PSAPISRV        SRVGRP=APPSRV
                SRVID=150
                MIN=1
                MAX=1
                RQADDR="APIQ"
```

```
                REPLYQ=Y
                CLOPT=" -A -- -C psappsrv.cfg -D TESTSERV -S PSAPISRV"

#
# JOLT Listener and Rep Server
#
JSL
                SRVGRP=JSLGRP
                SRVID=200
                CLOPT="-A --   -n //GO-FASTER-1:9000 -m 1 -M 1 -I 5 -c ANY -x 60
-T 60 -j 5000 -w JSH"

JREPSVR
                SRVGRP=JREPGRP
                SRVID=250
                CLOPT="-A -- -W -P D:\ps\hr75\appserv\h75d_pt756\jrepository"


*SERVICES

GetCertificate  SRVGRP=BASE
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

MenuList        SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

MgrGetObject    SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

MgrUpdObject    SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

MgrClear        SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

MgrPeek         SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

JavaMgrGetObj   SRVGRP=APPSRV
                LOAD=50 PRIO=50
```

```
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprLoad         SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprSave         SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprFullSave     SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprChangeValue  SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprFieldChange  SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprDefault      SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprInsertRow    SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprDeleteRow    SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprSearchStart  SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprSearchSave   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"
```

```
PprSecStart     SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprMItemSelPC   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PprPrePopupPC   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PsmSchedPrcs    SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

PsdGetListSvc   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

QdmGetListSvc   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

RamList         SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

RemoteCall      SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

SamGetParmsSvc  SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

SqlAccess       SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

SqlRequest      SRVGRP=APPSRV
```

```
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

SqlQuery        SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

StmChgPswd      SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

StmGetTimeOut   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

StmGetExplain   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

OpnQryDescribe  SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

OpnQryExecute   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

WamChgInstSvc   SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

WamStartInstSvc SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"

MsgAPI          SRVGRP=APPSRV
                LOAD=50 PRIO=50
                SVCTIMEOUT=300
                BUFTYPE="ALL"
# ------------------------------------------------------------------------
#************************************************************************
#  ubbgen subtitution values:
#
```

```
# [ 0]:                        {IPCKEY} :                          33796
# [ 1]:   {$Domain Settings\Domain ID} :                        TESTSERV
# [ 2]:                    {MAXSERVERS} :           MAXSERVERS       19

# [ 3]:                   {MAXSERVICES} :           MAXSERVICES      87

# [ 4]:                          {MACH} :                   GO-FASTER-1
# [ 5]:                       {$TUXDIR} :                   d:\ps\Tuxedo
# [ 6]:                   {$PS_SERVDIR} :  D:\ps\hr75\appserv\h75d_pt756
# [ 7]:                    {$TUXCONFIG} : D:\ps\hr75\appserv\h75d_pt756\PSTUXCFG
# [ 8]:                        {LOGDIR} : D:\ps\hr75\appserv\h75d_pt756\LOGS
# [ 9]:                           {FS} :                               \
# [10]:                       {ENVFILE} :                   psappsrv.env
# [11]:                           {UID} :                              0
# [12]:                           {GID} :                              0
# [13]:                 {MAXWSCLIENTS} :              MAXWSCLIENTS=120

# [14]:                 {MAXACCESSERS} :              MAXACCESSERS=139

# [15]: {$Domain Settings\Restartable} :                              Y
# [16]:        {$PSAUTH\Min Instances} :                              1
# [17]:        {$PSAUTH\Max Instances} :                              1
# [18]:                        {CFGFILE} :                   psappsrv.cfg
# [19]: {$Workstation Listener\Address} :                     //localhost
# [20]:   {$Workstation Listener\Port} :                           7000
# [21]: {$Workstation Listener\Encryption} :                                0
# [22]: {$Workstation Listener\Init Timeout} :                              5
# [23]:   {WSL Client Cleanup Timeout} :                         -T 60
# [24]: {$Workstation Listener\Min Handlers} :                              1
# [25]: {$Workstation Listener\Max Handlers} :                              1
# [26]: {$Workstation Listener\Max Clients per Handler} :
60
# [27]: {$Workstation Listener\Tuxedo Compression Threshold} :
5000
# [28]:      {$PSAPPSRV\Min Instances} :                              1
# [29]:      {$PSAPPSRV\Max Instances} :                              1
# [30]:       {$PSAPPSRV\Spawn Server} :
# [31]:      {$PSQCKSRV\Min Instances} :                              1
# [32]:      {$PSQCKSRV\Max Instances} :                              1
# [33]:       {$PSQCKSRV\Spawn Server} :
# [34]:      {$PSQRYSRV\Min Instances} :                              1
# [35]:      {$PSQRYSRV\Max Instances} :                              1
# [36]:       {$PSQRYSRV\Spawn Server} :
# [37]:      {$PSSAMSRV\Min Instances} :                              1
# [38]:      {$PSSAMSRV\Max Instances} :                              1
# [39]:      {$PSAPISRV\Min Instances} :                              1
# [40]:      {$PSAPISRV\Max Instances} :                              1
# [41]:       {$PSAPISRV\Spawn Server} :
# [42]:                        {TUXDEV} :
# [43]:       {$JOLT Listener\Address} :                 //GO-FASTER-1
# [44]:          {$JOLT Listener\Port} :                           9000
# [45]:  {$JOLT Listener\Min Handlers} :                              1
```

```
# [46]:  {$JOLT Listener\Max Handlers} :                              1
# [47]:  {$JOLT Listener\Init Timeout} :                              5
# [48]: {$JOLT Listener\Client Connection Mode} :                           ANY
# [49]: {$JOLT Listener\Max Clients per Handler} :
60
# [50]:              {Jolt Encryption} :
# [51]:  {Jolt Client Cleanup Timeout} :                          -T 60
# [52]: {$JOLT Listener\Jolt Compression Threshold} :
5000
# [53]:    {$PSQCKSRV\Service Timeout} :                            300
# [54]:    {$PSAPPSRV\Service Timeout} :                            300
# [55]:    {$PSSAMSRV\Service Timeout} :                            300
# [56]:    {$PSQRYSRV\Service Timeout} :                            300
# [57]:    {$PSAPISRV\Service Timeout} :                            300
# [58]:                     {$PS_HOME} :                    d:\ps\hr75
# [59]: {$Domain Settings\Add to PATH} :                 d:\orant80\bin
# [60]:          {$Startup\ServerName} :
#*************************************************************************
#*************************************************************************
#  ubbgen control values:
#
# [ 0]:                         {UNIX} :                          FALSE
# [ 1]:                      {WINDOWS} :                           TRUE
# [ 2]:                     {QUICKSRV} :                           TRUE
# [ 3]:                     {QUERYSRV} :                           TRUE
# [ 4]:                    {!QUERYSRV} :                          FALSE
# [ 5]:                         {JOLT} :                           TRUE
# [ 6]:                         {JRAD} :                          FALSE
#*************************************************************************
```

psappsrv.ubb

NB: Not all platforms generate the summary of variables at the bottom of the psappsrv.ubb file.

Simple Application Source

Simple Client

```
/*   Copyright (c) 1998 BEA Systems, Inc.
     All rights reserved

     THIS IS UNPUBLISHED PROPRIETARY
     SOURCE CODE OF BEA Systems, Inc.
     The copyright notice above does not
     evidence any actual or intended
     publication of such source code.
*/

/*   Copyright 1996 BEA Systems, Inc.       */
/*   THIS IS UNPUBLISHED PROPRIETARY SOURCE CODE OF       */
/*   BEA Systems, Inc.                      */
/*   The copyright notice above does not evidence any     */
/*   actual or intended publication of such source code.  */

/* #ident    "@(#) apps/simpapp/simpcl.c   $Revision: 1.1 $" */

#include <stdio.h>
#include "atmi.h"           /* TUXEDO  Header File */


#if defined(__STDC__) || defined(__cplusplus)
main(int argc, char *argv[])
#else
main(argc, argv)
int argc;
char *argv[];
#endif

{

     char *sendbuf, *rcvbuf;
     long sendlen, rcvlen;
     int ret;

     if(argc != 2) {
      (void) fprintf(stderr, "Usage: simpcl string\n");
      exit(1);
     }

     /* Attach to System/T as a Client Process */
     if (tpinit((TPINIT *) NULL) == -1) {
      (void) fprintf(stderr, "Tpinit failed\n");
      exit(1);
```

```
        }

        sendlen = strlen(argv[1]);

        /* Allocate STRING buffers for the request and the reply */

        if((sendbuf = (char *) tpalloc("STRING", NULL, sendlen+1)) == NULL) {
         (void) fprintf(stderr,"Error allocating send buffer\n");
         tpterm();
         exit(1);
        }

        if((rcvbuf = (char *) tpalloc("STRING", NULL, sendlen+1)) == NULL) {
         (void) fprintf(stderr,"Error allocating receive buffer\n");
         tpfree(sendbuf);
         tpterm();
         exit(1);
        }

        (void) strcpy(sendbuf, argv[1]);

        /* Request the service TOUPPER, waiting for a reply */
        ret = tpcall("TOUPPER", (char *)sendbuf, 0, (char **)&rcvbuf, &rcvlen,
    (long)0);

        if(ret == -1) {
         (void) fprintf(stderr, "Can't send request to service TOUPPER\n");
         (void) fprintf(stderr, "Tperrno = %d\n", tperrno);
         tpfree(sendbuf);
         tpfree(rcvbuf);
         tpterm();
         exit(1);
        }

        (void) fprintf(stdout, "Returned string is: %s\n", rcvbuf);

        /* Free Buffers & Detach from System/T */
        tpfree(sendbuf);
        tpfree(rcvbuf);
        tpterm();
        return(0);
    }
```
simpcl.c

Simple Server

```
/*   Copyright (c) 1998 BEA Systems, Inc.
     All rights reserved

     THIS IS UNPUBLISHED PROPRIETARY
     SOURCE CODE OF BEA Systems, Inc.
     The copyright notice above does not
     evidence any actual or intended
     publication of such source code.
*/

/*   Copyright 1996 BEA Systems, Inc.      */
/*   THIS IS UNPUBLISHED PROPRIETARY SOURCE CODE OF       */
/*   BEA Systems, Inc.                     */
/*   The copyright notice above does not evidence any     */
/*   actual or intended publication of such source code.  */

/* #ident    "@(#) apps/simpapp/simpserv.c $Revision: 1.1 $" */

#include <stdio.h>
#include <ctype.h>
#include <atmi.h>    /* TUXEDO Header File */
#include <userlog.h> /* TUXEDO Header File */

/* tpsvrinit is executed when a server is booted, before it begins
   processing requests.  It is not necessary to have this function.
   Also available is tpsvrdone (not used in this example), which is
   called at server shutdown time.
*/

#if defined(__STDC__) || defined(__cplusplus)
tpsvrinit(int argc, char *argv[])
#else
tpsvrinit(argc, argv)
int argc;
char **argv;
#endif
{
     /* Some compilers warn if argc and argv aren't used. */
     argc = argc;
     argv = argv;

     /* userlog writes to the central TUXEDO message log */
     userlog("Welcome to the simple server");
     return(0);
}

/* This function performs the actual service requested by the client.
   Its argument is a structure containing among other things a pointer
   to the data buffer, and the length of the data buffer.
*/
```

```
#ifdef __cplusplus
extern "C"
#endif
void
#if defined(__STDC__) || defined(__cplusplus)
TOUPPER(TPSVCINFO *rqst)
#else
TOUPPER(rqst)
TPSVCINFO *rqst;
#endif
{

     int i;

     for(i = 0; i < rqst->len-1; i++)
      rqst->data[i] = toupper(rqst->data[i]);

     /* Return the transformed buffer to the requestor. */
     tpreturn(TPSUCCESS, 0, rqst->data, 0L, 0);
}
```

Simple UBB file

```
#ident       "@(#) apps/simpapp/ubbsimple  $Revision: 1.1 $"

#Skeleton UBBCONFIG file for the TUXEDO Simple Application.
#Replace the <bracketed> items with the appropriate values.

*RESOURCES
#IPCKEY              <Replace with a valid IPC Key>
IPCKEY               123456

#Example:
#IPCKEY              123456

DOMAINID     simpapp
MASTER               simple
MAXACCESSERS 10
MAXSERVERS   5
MAXSERVICES  10
MODEL SHM
LDBAL N

*MACHINES
DEFAULT:
#     APPDIR="<Replace with the current directory pathname>"
#     TUXCONFIG="<Replace with your TUXCONFIG Pathname>"
#     TUXDIR="<Directory where TUXEDO is installed>"
      APPDIR="d:\ps\Tuxedo\apps\simpapp"
      TUXCONFIG="d:\ps\Tuxedo\apps\simpapp\tuxconfig"
      TUXDIR="d:\ps\Tuxedo"
```

```
#Example:
#     APPDIR="/home/me/simpapp"
#     TUXCONFIG="/home/me/simpapp/tuxconfig"
#     TUXDIR="/usr/Tuxedo"

#<Machine-name>      LMID=simple
GO-FASTER-1   LMID=simple

#Example:
#beatux       LMID=simple

*GROUPS
GROUP1
      LMID=simple    GRPNO=1 OPENINFO=NONE

*SERVERS
DEFAULT:
      CLOPT="-A -r"

simpserv      SRVGRP=GROUP1 SRVID=1

*SERVICES
TOUPPER
```

ubbsimple

CHAPTER 7

OTHER THINGS

Web stuff

Distributed Domain

Slisten/tlisten

split node domain – per ubs

monitoring scripts – unix direct to oracle

tuxlog convertor – requires ksh for NT or awk for NT

TMTRACE *:ulog:dye chtr

client side trace

In PT7.5 more PeopleCode executes on the application server than in version 7.

Single MP domain

```
*RESOURCES
IPCKEY          33489        # ( 32768 < IPCKEY < 262143 )
MASTER          SITE1
DOMAINID        TESTSERV
MODEL           MP
LDBAL           Y
#
MAXMACHINES     256             # min, default=256
MAXGROUPS       100             # min, default=100
MAXSERVERS       22

MAXSERVICES      119


OPTIONS             LAN           # dmk - added this option for MP - MIGRATE
starts bridge
#
```

```
# ------------------------------------------------------------------------

*MACHINES
"GO-FASTER-1" LMID="SITE1"                          # Machine name must be
upper case
        TUXDIR="d:\ps\tuxedo"                      # Paths cannot end in '\'
        APPDIR="D:\ps\hr75\appserv\h75d_smp"                # include the
database path
        TUXCONFIG="D:\ps\hr75\appserv\h75d_smp\PSTUXCFG"
        ULOGPFX="D:\ps\hr75\appserv\h75d_smp\LOGS\TUXLOG"
        ENVFILE="D:\ps\hr75\appserv\h75d_smp\psappsrv.env"
        UID=0                                  # Has to be 0 at this time.
        GID=0                                  # Has to be 0 at this time.
        TYPE="i386NT"
        NETLOAD=0                                  # We are not using multiple
machines.
        MAXWSCLIENTS=60

        MAXACCESSERS=82


# ------------------------------------------------------------------------
# dmk added this section for MP

*NETWORK

# ------------------------------------------------------------------------

*GROUPS
```