ORACLE
PeopleSoft

PeopleTools 8.60: Automated Configuration Management

July 2024

ORACLE

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at https://docs.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit https://docs.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit https://docs.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Contents

Preface: Preface ..... **vii**
Understanding the PeopleSoft Online Help and PeopleBooks vii
Hosted PeopleSoft Online Help vii
Locally Installed PeopleSoft Online Help vii
Downloadable PeopleBook PDF Files vii
Common Help Documentation vii
Field and Control Definitions viii
Typographical Conventions viii
ISO Country and Currency Codes ix
Region and Industry Identifiers ix
Translations and Embedded Help x
Using and Managing the PeopleSoft Online Help x
PeopleTools Related Links x
Contact Us x
Follow Us xi
Chapter 1: Getting Started with Automated Configuration Management **13**
Understanding Automated Configuration Management 13
Common Terms in Automated Configuration Management 14
Security Permissions Required for Automated Configuration Management 14
Chapter 2: Working with the Automated Configuration PIA Interface **17**
Understanding Automated Configuration Using PIA 17
Delivered Configuration Plug-ins 18
ConfigurePTF 18
PTIBActivateDomain 18
PTIBActivateLocalServices 19
PTIBActivateQueues 19
PTIBCleanUpNetworkData 19
PTIBConfigureDBNode 20
PTIBConfigureGatewayNodes 21
PTIBConfigureGatewayProperties 23
PTIBIntrospectionDeployment 24
PTIBNodeRegistration 24
PTIBRenameNode 24
PTIBRoutingDefinition 25
PTPerformanceMonitor 25
PTPNInterdomainConfig 28
PTPNSystemConfig 30
PTURLConfiguration 30
PTPPConfigurePortalCluster 31
PTUNConfigureRemoteFolders 31
PTProcessSchedulerReportNode 32
PTProcessSchedulerServerConfig 33
PTSFAdministerRemoteSearch 37
PTSFAdministerSearch 37
PTSFCleanupDeploymentData 38
PTSFConfigureSrchInstance 39

PTSFMonitorConfiguration....41
PTWebProfileConfig....42
PTWorkflowEmailConfiguration....51
PTWorkflowProfileUpdate....52
PTWorkflowSystemConfiguration....52
PTImportCertificatestoDB....52
Executing Configuration Plug-ins....53
Executing a Template....53
Monitoring a Configuration....56
Reviewing the Configuration Details....57
Running Functional Validation....58
Handling Errors During Plug-in Execution....60
Scheduling An Automated Configuration in PIA....62
Scheduling a Configuration Process....62
Chapter 3: Working with Custom Configuration Plug-ins and Templates....65
Understanding Custom Configuration Plug-ins and Templates....65
Working with Custom Configuration Plug-ins....65
Creating a Configuration Plug-in....65
Registering a Configuration Plug-in....67
Working with Custom Templates....69
Creating a Template....69
Editing Configuration Plug-in Properties....72
Defining Template Variables....73
Importing Template Variables....74
Importing a Template Using PIA....76
Adding a Template to a Permission List....77
Managing Configuration Plug-in Dependency....78
Understanding Resolving Plug-in Dependency Dynamically....78
Adding Configuration Plug-ins to Plug-in Groups....79
Managing Plug-ins within a Plug-in Group....79
Managing Dependencies Between Plug-in Groups....80
Initiating a Resolve Dependency Request....81
Chapter 4: Working with Automated Configuration Using the Command Line....85
Understanding Automated Configuration Using the Command Line....85
Understanding the Template File....85
Creating a Template File Manually....86
Working with the Structure and General Settings of a Template File....86
Specifying General Settings....87
Specifying Template Settings....88
Enabling Configuration Plug-ins....89
Creating a Template File in PIA....89
Exporting a Template....89
Running the PSRUNACM Script....91
Running PSRUNACM in Windows....91
Running PSRUNACM in UNIX....93
Running the PTEM_CONFIG Application Engine Program....95
Understanding the PTEM_CONFIG Program Run....96
Ensuring Sufficient Security Permissions....96
Setting the Environment Variables for the Command Line Session....96
Running the PTEM_CONFIG Application Engine Program....97
Verifying Your Configuration Settings....99

Applying Plug-in Property Updates Selectively....99
Handling Execution Errors and Status on the Command Line....103
Understanding the Console Output....104
Interpreting the Console Output....104
PSRUNACM Error Codes....105
Working with Pre and Post-Conditions with Template Files and the Console....105
Running the Console in Quiet Mode....107
Chapter 5: Working with Sensitive Data....109
Understanding Password Encryption in Template File....109
Chapter 6: Migrating Configuration Data....111
Migrating Configuration Data Using ADS....111
Tables for Configuration Data Migration....111
ADS Definition for Template....111
ADS Definition for Plug-in....112
Exporting Configuration Data....112
Chapter 7: Sample Template File....113
Sample Template File....113

Preface

Understanding the PeopleSoft Online Help and PeopleBooks

The PeopleSoft Online Help is a website that enables you to view all help content for PeopleSoft applications and PeopleTools. The help provides standard navigation and full-text searching, as well as context-sensitive online help for PeopleSoft users.

Hosted PeopleSoft Online Help

You can access the hosted PeopleSoft Online Help on the Oracle Help Center. The hosted PeopleSoft Online Help is updated on a regular schedule, ensuring that you have access to the most current documentation. This reduces the need to view separate documentation posts for application maintenance on My Oracle Support. The hosted PeopleSoft Online Help is available in English only.

To configure the context-sensitive help for your PeopleSoft applications to use the Oracle Help Center, see Configuring Context-Sensitive Help Using the Hosted Online Help Website.

Locally Installed PeopleSoft Online Help

If you're setting up an on-premises PeopleSoft environment, and your organization has firewall restrictions that prevent you from using the hosted PeopleSoft Online Help, you can install the online help locally. Installable PeopleSoft Online Help is made available with selected PeopleSoft Update Images and with PeopleTools releases for on-premises installations, through the Oracle Software Delivery Cloud.

Your installation documentation includes a chapter with instructions for how to install the online help for your business environment, and the documentation zip file may contain a README.txt file with additional installation instructions. See *PeopleSoft 9.2 Application Installation* for your database platform, "Installing PeopleSoft Online Help."

To configure the context-sensitive help for your PeopleSoft applications to use a locally installed online help website, see Configuring Context-Sensitive Help Using a Locally Installed Online Help Website.

Downloadable PeopleBook PDF Files

You can access downloadable PDF versions of the help content in the traditional PeopleBook format on the Oracle Help Center. The content in the PeopleBook PDFs is the same as the content in the PeopleSoft Online Help, but it has a different structure and it does not include the interactive navigation features that are available in the online help.

Common Help Documentation

Common help documentation contains information that applies to multiple applications. The two main types of common help are:

- Application Fundamentals

- Using PeopleSoft Applications

Most product families provide a set of application fundamentals help topics that discuss essential information about the setup and design of your system. This information applies to many or all applications in the PeopleSoft product family. Whether you are implementing a single application, some combination of applications within the product family, or the entire product family, you should be familiar with the contents of the appropriate application fundamentals help. They provide the starting points for fundamental implementation tasks.

In addition, the *PeopleTools: Applications User's Guide* introduces you to the various elements of the PeopleSoft Pure Internet Architecture. It also explains how to use the navigational hierarchy, components, and pages to perform basic functions as you navigate through the system. While your application or implementation may differ, the topics in this user's guide provide general information about using PeopleSoft applications.

Field and Control Definitions

PeopleSoft documentation includes definitions for most fields and controls that appear on application pages. These definitions describe how to use a field or control, where populated values come from, the effects of selecting certain values, and so on. If a field or control is not defined, then it either requires no additional explanation or is documented in a common elements section earlier in the documentation. For example, the Date field rarely requires additional explanation and may not be defined in the documentation for some pages.

Typographical Conventions

The following table describes the typographical conventions that are used in the online help.

Typographical Convention	***Description***
Key+Key	Indicates a key combination action. For example, a plus sign (+) between keys means that you must hold down the first key while you press the second key. For **Alt+W,** hold down the **Alt** key while you press the **W** key.
. . . (ellipses)	Indicate that the preceding item or series can be repeated any number of times in PeopleCode syntax.
{ } (curly braces)	Indicate a choice between two options in PeopleCode syntax. Options are separated by a pipe (|).
[ ] (square brackets)	Indicate optional items in PeopleCode syntax.
& (ampersand)	When placed before a parameter in PeopleCode syntax, an ampersand indicates that the parameter is an already instantiated object. Ampersands also precede all PeopleCode variables.

Typographical Convention	*Description*
⇒	This continuation character has been inserted at the end of a line of code that has been wrapped at the page margin. The code should be viewed or entered as a single, continuous line of code without the continuation character.

ISO Country and Currency Codes

PeopleSoft Online Help topics use International Organization for Standardization (ISO) country and currency codes to identify country-specific information and monetary amounts.

ISO country codes may appear as country identifiers, and ISO currency codes may appear as currency identifiers in your PeopleSoft documentation. Reference to an ISO country code in your documentation does not imply that your application includes every ISO country code. The following example is a country-specific heading: "(FRA) Hiring an Employee."

The PeopleSoft Currency Code table (CURRENCY_CD_TBL) contains sample currency code data. The Currency Code table is based on ISO Standard 4217, "Codes for the representation of currencies," and also relies on ISO country codes in the Country table (COUNTRY_TBL). The navigation to the pages where you maintain currency code and country information depends on which PeopleSoft applications you are using. To access the pages for maintaining the Currency Code and Country tables, consult the online help for your applications for more information.

Region and Industry Identifiers

Information that applies only to a specific region or industry is preceded by a standard identifier in parentheses. This identifier typically appears at the beginning of a section heading, but it may also appear at the beginning of a note or other text.

Example of a region-specific heading: "(Latin America) Setting Up Depreciation"

Region Identifiers

Regions are identified by the region name. The following region identifiers may appear in the PeopleSoft Online Help:

- Asia Pacific
- Europe
- Latin America
- North America

Industry Identifiers

Industries are identified by the industry name or by an abbreviation for that industry. The following industry identifiers may appear in the PeopleSoft Online Help:

- USF (U.S. Federal)

- E&G (Education and Government)

Translations and Embedded Help

PeopleSoft 9.2 software applications include translated embedded help. With the 9.2 release, PeopleSoft aligns with the other Oracle applications by focusing our translation efforts on embedded help. We are not planning to translate our traditional online help and PeopleBooks documentation. Instead we offer very direct translated help at crucial spots within our application through our embedded help widgets. Additionally, we have a one-to-one mapping of application and help translations, meaning that the software and embedded help translation footprint is identical—something we were never able to accomplish in the past.

Using and Managing the PeopleSoft Online Help

Select About This Help in the left navigation panel on any page in the PeopleSoft Online Help to see information on the following topics:

- Using the PeopleSoft Online Help.
- Managing hosted Online Help.
- Managing locally installed PeopleSoft Online Help.

PeopleTools Related Links

PeopleTools 8.60 Home Page

PeopleSoft Search and Insights Home Page

“PeopleTools Product/Feature PeopleBook Index” (Getting Started with PeopleTools)

PeopleSoft Online Help

PeopleSoft Information Portal

PeopleSoft Spotlight Series

PeopleSoft Training and Certification | Oracle University

My Oracle Support

Oracle Help Center

Contact Us

Send your suggestions to psoft-infodev_us@oracle.com.

Please include the applications update image or PeopleTools release that you’re using.

Follow Us

Icon	*Link*
	Watch PeopleSoft on YouTube
	Follow @PeopleSoft_Info on X.
	Read PeopleSoft Blogs
	Connect with PeopleSoft on LinkedIn

Chapter 1

Getting Started with Automated Configuration Management

Understanding Automated Configuration Management

With each new release, Oracle PeopleTools provides new technology to our infrastructure, which adds rich, new features to support the functional requirements of our applications. While the new technology enhances the user experience and capabilities of our applications, the new technology often brings additional steps to the environment configuration and implementation process. This can create challenges for system administrators, development teams, and testing teams who routinely set up numerous environments that need to be refreshed on a regular basis. For example, with each implementation, system administrators routinely deal with multiple copies of various types of environments, such as production environments, testing environments, demonstration environments, , development environments, and so on. Oracle PeopleTools and PeopleSoft application teams share this same challenge.

For this reason, Oracle PeopleTools provides an automated configuration management framework that enables you to store your environment configuration settings in a template stored in the database or an external template file. Because these settings are set once, and then saved, you can leverage the stored settings and reapply them easily when needed. PeopleTools provides two methods of running the automated configuration process — using the Automated Configuration Manager browser interface in PIA or using the command line. When running the process from the command line, you can use a batch file or shell script or the Application Engine command line options.

After setting up the basic infrastructure of a PeopleSoft environment, including database, application server, Process Scheduler server, and PIA domain, you run the configuration program (either using PIA or command line). This configuration program reads your configuration settings (referred to as environment properties) that you have stored in the template or template file, and inserts the stored values into the database, saving you from updating the settings manually each and every time you create or refresh an environment.

The settings stored in the template or template file are those that you typically enter on a configuration page in PeopleTools and save to the database. For example, the value for your Integration Gateway URL, which you would normally add manually on the Gateways page, can be entered in your template or template file once, and then retrieved by the automated configuration management framework and inserted into the database each time you refresh that environment.

A template is a composite of configuration plug-ins in which the properties and its values are defined. The configuration program fires configuration plug-ins that take the properties specified in the plug-in and configures the feature associated with that plug-in. Each plug-in provided is focused on a particular setting or functional area of your configuration. You enable configuration plug-ins by referencing them in your template or template file and specifying the settings for that configuration plug-in to insert into the appropriate fields in the configuration interface.

Common Terms in Automated Configuration Management

Common terms used in automated configuration management:

Term	*Description*
Plug-in	A plug-in is an application class that corresponds to a configuration. For example, to configure Integration Broker Gateway nodes, the application class (plug-in) PTIBConfigureGatewayNodes in the application package PTEM_CONFIG is called.
Template	Templates are used for organizing, editing property values, and running the configuration program through PIA and command line. Template values can be stored in the database or exported to an external template file.
Property	A property is the parameter or environment setting for setting the value of a configuration. For example, env.default_local_ node is the property name used for setting the value of default local node while configuring Integration Broker. This property will be present in the template if configuring through PIA or in the template file if configuring through command line.
Template Variable	A template variable can be assigned a value, which then can be used to assign values for properties in cases where values are recurring in multiple places in a template or template file. For example, a host name would likely occur numerous times when setting URL values, so it is typically expressed as a variable.

Security Permissions Required for Automated Configuration Management

Before you begin using the automated configuration management framework, ensure that your user ID has the required roles.

Ensure that the following roles are assigned to the user IDs that will be used to run the configuration processes, which include:

- ACM Administrator
- Portal Administrator
- Search Administrator
- Search Server
- Search Developer

- Integration Administrator

Also, when you use or create a template, you must ensure that the template is added to the required permission list.

See Adding a Template to a Permission List.

Chapter 2

Working with the Automated Configuration PIA Interface

Understanding Automated Configuration Using PIA

The automated configuration management framework allows you to automate product configurations using your browser (PIA) or from the command line.

This section describes the use of the Automated Configuration Manager interface in PIA. Automated Configuration Manager allows you to configure an environment with the required product settings without navigating manually to the product's actual configuration pages.

Important! PeopleSoft recommends that product configuration through PIA be used only to test a template. The actual product configurations in a system must be performed through the command line.

Automated Configuration Manager uses a template to store settings and run a configuration program. In a template, you can include all the product configurations called plug-ins that you require for an environment, and you can group the plug-ins based on the product. For example, plug-ins required for an Integration Broker configuration can belong to one group, whereas the plug-ins required for a Search Framework configuration can be another group in the same template. This enables you to configure more than one product in a single configuration program run, and you can also control the sequence in which products are configured. For example, you can set Integration Broker to be setup and configured first and the Search Framework second to account for the dependencies the Search Framework has upon Integration Broker.

Your configuration templates can be exported from one environment and imported by others so you can reuse templates in different environments by editing the configuration plug-in properties and values as required for an environment.

Automated Configuration Manager allows you to:

- Create, edit, manage, import, and export templates.
- Define template variables.
- Register configuration plug-ins.
- Specify template processing modes.
- Monitor a configuration run.

Delivered Configuration Plug-ins

PeopleTools delivers these plug-ins for customer usage. The properties of each plug-in are described in the following sections.

ConfigurePTF

Application class - PTEM_CONFIG:ConfigurePTF

The ConfigurePTF plug-in grants PTF User role to the PTF user and allows to accept non SSL requests.

Property	***Default Value***	***Description***
env.ptf_user	@userid@	Enter the PTF user. The userid variable is a template variable and @userid@ resolves at run time to the provided value. See Defining Template Variables.
env.ptf_allow_untrusted	True	Flag to allow non-trusted SSL certificates.
env.ptf_use_page_prompt	True	Select to use Page Prompt and Prompt OK steps during recording in place of menu navigation. The Use Page Prompt option is also available on the PTF Test Recorder toolbar. The option selected here is the default for all users in this environment. The option selected on the PTF Test Recorder toolbar overrides this selection for that recording session.
env.ptf_use_message_recognition	True	Select to create entries for the Message Recognition feature during recording automatically. The Use Message Recognition option is also available on the PTF Test Recorder toolbar. The option selected here is the default for all users in this environment. The option selected on the PTF Test Recorder toolbar overrides this selection for that recording session.

PTIBActivateDomain

Application class - PTEM_CONFIG:PTIBActivateDomain

The PTIPActivateDomain plug-in activates pub/sub domain.

Property	Default Value	Description
domain.activate_retry_count	10	Enter the number of retries.
domain.activate_wait_time	10	Enter the wait time for domain activation.

PTIBActivateLocalServices

Application class: PTEM_CONFIG:PTIBActivateLocalServices

Property	Default Value	Description
env.activate_local_integration_groups		A comma separated list of the Integration Groups to be activated.
env.activate_service		A comma separated list of Services to be activated.

PTIBActivateQueues

Application class - PTEM_CONFIG:PTIBActivateQueues

The PTIBActivateQueues plug-ins activates the Integration Broker queue.

Property	Default Value	Description
queue.activate_queue_list	PS_ALL	Use PS_ALL to activate all queues. If you want to activate selected queues, enter queue names separated by commas.
queue.activate_queue_status	1	Valid values: • 1 - activate a queue • 0 - pause a queue Default value is 1.

PTIBCleanUpNetworkData

Application class: PTEM_CONFIG:PTIBCleanUpNetworkData

PTIBCleanUpNetworkData clears Integration Broker Network configuration data from the system.

Property	Default Value	Description
env.network_nodenames_to_cleanup	NODE1,NODE2	Comma delimited list of node names to have configuration data cleared from network. If left bank, the plug-in clears configuration data from the network for all nodes currently in the network, except for the default local node.

PTIBConfigureDBNode

Application class - PTEM_CONFIG:PTIBConfigureDBNode

The PTIBConfigureDBNode plug-in configures the database node.

Property	Default Value	Description
env.pia_webserver_host	@host@.@domain@	Server host of the PIA domain.
env.pia_webserver_port	@httpport@	HTTP port on which the PIA domain listens.
env.pia_webserver_ssl_port	@sslport@	HTTPS port on which the PIA domain listens.
env.pia_site_name	PS	PeopleSoft site name.
env.gateway_host	@host@.@domain@	If using multiple machines, enter the gateway server host.
env.gateway_port	@httpport@	If using multiple machines, enter the gateway HTTP port.
env.gateway_ssl_port	@sslport@	If using multiple machines, enter the gateway HTTPS port.
env.use_ssl_gateway	False	This Boolean value specifies whether an SSL gateway is configured for the PeopleSoft system.
env.use_ssl_webserver	False	This Boolean value specifies whether SSL is used or not. It is based on this flag that the security mode is set. If the flag is not set, HTTP is used, else HTTPS is used for node URI.

Property	*Default Value*	*Description*
env.default_user_id	@userid@	Default user ID for the environment.
env.default_local_node_pass		Default local node password.
env.anonymous_default_user_id	@userid@	Specifies the default user ID for message node name ANONYMOUS.
env.default_local_node	@nodename@	Default local node
env.wsdl_external_user_id	@userid@	Specifies the external user ID for wsdl_ node.
env.wsdl_external_pass		Specifies the external user password for wsdl_node.
env.configure_wsdl_node	False	Flag to configure WSDL node
env.wsdl_node_tokentype	NONE	Specifies the authentication token type for WSDL. Valid values: • NONE • STSD - SAML token • USRT - User name token
env.wsdl_node_tokenencrypted	0	Encryption for WSDL node
env.wsdl_node_tokensigned	0	Digital Signature for WSDL node
env.wsdl_node_ibencryptionlevel	A	Specifies the encryption level for WSDL node. Valid values: • A - All • B - Body • H - Header level

PTIBConfigureGatewayNodes

Application class - PTEM_CONFIG:PTIBConfigureGatewayNodes

The PTIBConfigureGatewayNodes plug-in configures gateway URL, load connectors, and defines node in the gateway.

Property	***Default Value***	***Description***
env.gateway_host	@host@.@domain@	Gateway host
env.gateway_port	@httpport@	Gateway port
env.gateway_ssl_port	@sslport@	Gateway SSL port
env.use_ssl_gateway	False	Flag to determine whether the configuration is secure or non secure. Valid values: • False - indicates non secure (http) configuration • True - indicates secure (https) configuration
env.default_local_node	@nodename@	Default local node
env.gateway_user	administrator	Specify the gateway administrator user ID.
env.gateway_password		Specify the password for the gateway administrator user ID.
env.ib_appserver_host	@host@.@domain@	Server host of the application server domain.
env.ib_jolt_port	@jslport@	Jolt port.
env.ib_node_proxy_userid	@userid@	User ID for proxy, if used.
env.ib_node_proxy_password		Password for proxy, if used.
env.tools_release	@tools_release@	PeopleTools release level.
env.ib_appserver_domain_password		Password for the application server domain.

Property	*Default Value*	*Description*
env.ib_virtual_node		When you want a PeopleSoft node to process an inbound request that was sent without a destination node, the integration system directs the request to the node specified for this property.
env.ib_set_as_default_node	True	This flag is set for the node which hosts the common gateway. The flag is true only for the cluster node which hosts the common gateway.

PTIBConfigureGatewayProperties

Application class - PTEM_CONFIG:PTIBConfigureGatewayProperties

The PTIBConfigureGatewayProperties plug-in sets keystore password, proxy host, proxy port, and non proxy host in gateway properties file.

Property	*Default Value*	*Description*
env.gateway_host	@host@.@domain@	Gateway host.
env.gateway_port	@httpport@	Gateway port.
env.gateway_ssl_port	@sslport@	Gateway SSL port.
env.use_ssl_gateway	false	Flag to determine whether the configuration is secure or non-secure. Valid values: • True - HTTPS • False - HTTP
env.gateway_user	administrator	Gateway user name.
env.gateway_password		Gateway user password.
env.gateway_keystore_password		Gateway keystore password.
env.gateway_proxy_host		Gateway proxy host.
env.gateway_proxy_port		Gateway proxy port.

Property	Default Value	Description
env.gateway_non_proxy_hosts	@host@.@domain@|localhost|*.example.com	Gateway non proxy hosts.
env.ib_synchronize_map_files	Y	Execute MAP File Synchronization

PTIBIntrospectionDeployment

Application class: PTEM_CONFIG:PTIBIntrospectionDeployment

PTIBIntrospectionDeployment introspects and deploys the selected services.

Property	Default Value	Description
route.nodes.PT_LOCAL.NODEGROUP1	NODE1,NODE2,NODE3	Comma delimited list of nodes to introspect.
route.integration_groups.PT_LOCAL.NODEGROUP1	INTGROUP1,INTGROUP2	Comma delimited list of integration groups to introspect.
route.local_rest_introspection.PT_LOCAL	Y	Enables local REST introspection for this database.

PTIBNodeRegistration

Application class - PTEM_CONFIG:PTIBNodeRegistration

The PTIBNodeRegistration plug-in registers new nodes to the Integration Broker network.

Property	Default Value	Description
integration_network.NODEGROUP1	NODE1,NODE2,NODE3,NODE4	Comma-separated list of nodes to register.

PTIBRenameNode

Application class - PTEM_CONFIG:PTIBRenameNode

The PTIBRenameNode plug-in renames the default local node.

Property	Default Value	Description
env.default_local_node	@nodename@	Default local node.

Property	*Default Value*	*Description*
env.app_msg_purge_all_dms	true	Purge application server messages.

PTIBRoutingDefinition

Application class: PTEM_CONFIG:PTIBRoutingDefinition

PTIBRoutingDefinition configures the Integration Broker routing definition.

Property	*Default Value*	*Description*
env.routingdefinitionname		Integration Broker routing definition name.
env.connector_prop_id		Connector property ID.
env.connector_prop_name		Connector property name.
env.connector_prop_value		Connector property value.
env.add_connector_properties	Y	A flag determining if connector properties need to be added.
env.delete_connector_properties	N	A flag determining if connector properties need to be deleted.

PTPerformanceMonitor

Application class - PTEM_CONFIG:PTPerformanceMonitor

Before you run the PTPerformanceMonitor plug-in, ensure that the following pre-requisites are set in the monitoring system and in the monitored system:

Note: The PTPerformanceMonitor plug-in must be run in the monitoring and in the monitored systems.

- Perf Collator should be set to *Yes* (enabled) for the corresponding application server domain.
- In the psappsrv.cfg file and the psprcs.cfg file, the EnablePPM Agent parameter should be set to *1*.
- Process scheduler domain must be up and running so that the reaper and archive jobs can be scheduled successfully.
- In the web profile configuration, set the following:
  - In the General tab, select the EnablePPM Agent parameter.
  - In the Custom Properties tab, set the value of PPMConsole parameter to True.

If the PPMConsole parameter is not present, add the parameter. The parameter must be of Boolean type.

Additionally, to enable Usage Monitor as part of the performance monitor configurations, the following pre-requisites are required:

- In the psappsrv.cfg file, the Usage Monitoring State parameter should be set to 1 or 2.
- On the System Defaults page (PeopleTools, Performance Monitor, Administration, System Defaults), the Enable Usage Monitoring check box should be selected. Alternatively, in the PTPerformanceMonitor plug-in properties, you can set the env.usemon_sw property to 1.

Important! After executing the PTPerformanceMonitor plug-in, clear the cache and restart the application server and Web server for the configurations to take effect.

Property	*Default Value*	*Description*
env.pm_sampling_rate	0	Agent PMU sample rate (1/X).
env.pm_buff_int	0	Agent buffering interval (in seconds).
env.pm_sample_int	0	Agent event sample rate (in seconds).
env.pm_ping_int	0	Agent heartbeat interval (in seconds).
env.pm_user_trace		Allow performance trace.
env.pm_max_buff	0	Agent maximum buffer size (in bytes).
env.pm_max_hist_age	0	Retention period (in days).
env.usemon_sw	0	Enable usage monitoring.
env.pm_max_trans_tmout	0	PMU timeout (in days).
env.pm_archive_mode		Valid values for archiving: • 0 - Archive nothing • 1 - Archive data • 2 - Delete data • 3 - Delete system

Property	*Default Value*	*Description*
env.pm_filter_level2		Filter level values for log documentation: • 01 - Standby • 02 - Error • 03 - Warning • 04 - Standard • 05 - Verbose • 06 - Debug
env.monitor_url		Enter an URL in the following format or enter NONE. Format: http(s)://host[:port]/monitor/[site]/ If you enter NONE, the performance monitor components on the monitored system are not enabled.
env.ppmi_user_name		PPMI user ID.
env.ppmi_password		PPMI password.
env.collator_row_limit		Collator row limit. Use 0 for unlimited rows.
env.ppmi_url		PPMI URL. Format: http://host[:port]/monitor/[site]/
env.schedule_reaper_job	Y	Flag to determine whether the Reaper job needs to be scheduled.
env.schedule_lookup_job	Y	Flag to determine whether the Lookup job needs to be scheduled.
env.schedule_archive_job	Y	Flag to determine whether the Archive job needs to be scheduled.
env.schedule_umlookup_job	Y	Flag to whether the Usage Monitor Lookup job needs to be scheduled.

Property	*Default Value*	*Description*
env.cluster_list		List of performance monitor clusters.
env.insert_cluster_urls	N	Flag to determine whether performance monitor clusters need to be inserted or not.
env.delete_cluster_urls	N	Flag to determine whether performance monitor clusters need to be deleted or not.

PTPNInterdomainConfig

Application class - PTEM_CONFIG: PTPNInterdomainConfig

The PTPNInterdomainConfig plug-in configures notifications from Process Scheduler domain for Inter Domain Event Configuration.

Note: The PTPNInterdomainConfig plug-in can be executed only through command line.

The PTPNInterdomainConfig plug-in can be used in two scenarios:

- Application server and Process Scheduler domain are on the same host.
- Application server and Process Scheduler domain are on different hosts.

Application Server and Process Scheduler Domain on the Same Host

Property	*Default Value*	*Description*
env.appserverdomainname		Application server domain name.
env.prcsdomainname		Process Scheduler domain name.
env.joltport	9033	Sets the JSL port number.
env.hostnameappserv		Sets the fully qualified domain name of the application server.
env.appserverport	7988	Sets the application server port number.
env.prcsport	8988	Sets the Process Scheduler port number.
env.prcsscheduleserver		Process Scheduler server name.

Property	*Default Value*	*Description*
env.hostnameprcs		Sets the fully qualified domain name of the Process Scheduler.

Application Server and Process Scheduler Domain on Different Hosts

You should run the plug-in separately on the hosts where the application server and Process Scheduler are running.

Properties for configuring Process Scheduler on one host:

Property	*Default Value*	*Description*
env.prcsdomainname		Process Scheduler domain name.
env.domainid		Application server domain name which is to be configured.
env.joltport	9033	Sets the JSL port number.
env.hostnameappserv		Sets the fully qualified domain name of the application server.
env.appserverport	7988	Sets the application server port number.
env.prcsport	8988	Sets the Process Scheduler port number.

Properties for configuring application server on another host:

Property	*Default Value*	*Description*
env.appserverdomainname		Application server domain name.
env.prcsscheduleserver		Process Scheduler server name.
env.hostnameprcs		Sets the fully qualified domain name of the Process Scheduler.
env.prcsport	8988	Sets the Process Scheduler port number.

Property	Default Value	Description
env.appserverport	7988	Sets the application server port number.

PTPNSystemConfig

Application class: PTEM_CONFIG:PTPNSystemConfig

The PTEM_CONFIG:PTPNSystemConfig plug-in sets the system configurations required for Push Notification.

Property	Default Value	Description
env.tbe_status	N	Enables TBE.
env.ptpn_row_cnt	10	Sets the maximum number of rows to be displayed in the notification window.
env.ptpn_cat_cnt	10	Sets the maximum number of categories.
env.ptpn_max_cnt	100	Sets the maximum number of notifications to load.

PTURLConfiguration

Application class - PTEM_CONFIG:PTURLConfiguration

The PTURLConfiguration plug-in configures URL configurations.

Property	Default Value	Description
env.url_config_url_id	TEST	URL ID.
env.url_config_url_descr	DESCR	URL description.
env.url_config_url	http://	URL name.
env.url_config_comments	Comments	Comments
env.url_config_property_name		Property name.
env.url_config_property_value		Property value.

Property	Default Value	Description
env.password_property_value		Holds the property value in case the env.url_config_property_name equals "PASSWORD.”

PTPPConfigurePortalCluster

Application class - PTEM_CONFIG:PTPPConfigurePortalCluster

The PTPPConfigurePortalCluster plug-in configures portal host nodes of the portal cluster.

Property	Default Value	Description
env.portal_cluster_nodes	PA,HRMS,ERP,EPM,CRM	Comma separated values determine the names of nodes that needs to be part of the portal cluster.

PTUNConfigureRemoteFolders

Application class - PTEM_CONFIG:PTUNConfigureRemoteFolders

The PTUNConfigureRemoteFolders plug-ins configures unified navigation remote folders.

Property	Default Value	Description
env.remote_folders	HRMS	Comma separated values specify remote node names that need to be accessed for Unified Navigation.
env.remote_folders	NODE1	Comma separated values specify remote node names that need to be accessed for Unified Navigation.
env.portal_cluster_nodes		Comma separated values determine the names of nodes that need to be part of the portal cluster.
NODE1.label		Label for the folder, which will appear in the portal system's menu.
NODE1.portal_name	EMPLOYEE	Portal name.
NODE1.remote_node_name	NODE1	Remote node name.
NODE1.remote_folder_name	PORTAL_ROOT_OBJECT	Remote folder name.

Property	Default Value	Description
NODE1.local_parent_folder_name	PORTAL_ROOT_OBJECT	Parent of the remote folder.

PTProcessSchedulerReportNode

Application class - PTEM_CONFIG:PTProcessSchedulerReportNode

The PTProcessSchedulerReportNode plug-in configures Process Scheduler report node configuration.

Property	Default Value	Description
env.pt_prop_name		Property name.
env.wrkpassword		Node password.
env.wrkcnfpassword		Confirm password.
env.wrkoperpswd		Operator password (encrypted).
env.cdm_ssl_mode		Valid values for SSL mode: • 0 - EXPLICIT • 1 - IMPLICIT
env.url		URL ID.
env.uri_host		URI host.
env.pt_fileencpasswd		Encrypted password.
env.distnodename		Distribution node name.
env.ftpaddress		File transfer protocol address.
env.winnetworkpath		Windows network path.
env.cdm_trprotocol		Transfer protocol.
env.descrlong		Description
env.uri_port	0	URI port.

Property	*Default Value*	*Description*
env.uri_resource		URI resource.
env.operpswd		Operator password (encrypted).
env.cdm_proto		Valid values for protocol: • 0 - HTTP • 1 - HTTPS • 2 - XCOPY • 3 - FTP • 4 - FTPS • 5 - SFTP
env.opsys		Valid values for operating system: • 0 - DOS • 1 - NT/Win95 Client • 2 - Windows • 3 - OS/2 • 4 - UNIX • 5 - VMS • 6 - MPE/XL • 7 - OS390 • 9- OS/400
env.ftpid		File transfer ID.
env.ftpdirectory		Directory for FTP.
env.pt_prop_value		Property value.

PTProcessSchedulerServerConfig

Application class - PTEM_CONFIG:PTProcessSchedulerServerConfig

The PTProcessSchedulerServerConfig plug-in configures Process Scheduler server configuration.

Property	*Default Value*	*Description*
env.daemonprcsinst	0	Process instance.
env.maxconcurrent	0	Maximum concurrent processes.
env.notifywhensusp	0	Notify when suspended.
env.transfermaxretry	0	Maximum transfer retries.
env.srvrloadbaloptn		Server load balancing option. Valid values: • 0 - Do Not Use for Load Balancing • 1 - Use for Load Balancing
env.redistwrkoption		Redistribute workload option. Valid values: • 0 - Do not redistribute • 1 - Redistribute with same O/S • 2 - Redistribute to any O/S
env.minmem	0	Minimum memory required.
env.distid		Distribution ID.
env.servername		Server name.
env.prcsnotifyfreq	0	Notify frequency.
env.start_hours		Start hours.
env.notifywhenstartd	0	Notify when started.
env.maxapiaware	0	Maximum API aware tasks.
env.daemonenabled		Daemon enabled.
env.end_hours		End hours.
env.end_minutes		End minutes.

Property	*Default Value*	*Description*
env.notifydisabled		Notify disabled.
env.prcspriority		Process priority. Valid values: • 1 - Low • 5 - Medium • 9 - High
env.maxcpu	0	Maximum CPU usage required.
env.prcstype		Process type.
env.day_ofweek		Day Valid values: • 0 - Sunday • 1 - Monday • 2 - Tuesday • 3 - Wednesday • 4 - Thursday • 5 - Friday • 6 - Saturday
env.distnodename		Distribution node name.
env.prcscategory		Process category.
env.notifyservererror	0	Server errors.
env.notifywhendown	0	Notify when down.
env.sleeptime	0	Sleep time.
env.heartbeat	0	Heartbeat
env.transferlogfiles		Transfer log files to content.

Property	*Default Value*	*Description*
env.transferinterval	0	Interval for transfer attempt.
env.daemongroup		Daemon procedure group.
env.start_minutes		Time
env.distidtype		Distribution ID type. Valid values: • 2 - User ID • 3 - Role name
env.daemonsleeptime	0	Daemon sleep time.
env.daemoncyclecnt	0	Daemon recycle count.
env.descr		Description
env.opsys		Operating system. Valid values: • 0 - DOS • 1 - NT/Win95 Client • 2 - Windows • 3 - OS/2 • 4 - UNIX • 5 - VMS • 6 - MPE/XL • 7 - OS390 • 9- OS/400
env.processtypepriority	1	Process type priority. Valid values: • 1 - Low • 5 - Medium • 9 - High

Property	Default Value	Description
env.processtypemaxconcurrent	3	Process type maximum concurrent value.

PTSFAdministerRemoteSearch

Application class - PTEM_CONFIG:PTSFAdministerRemoteSearch

The PTSFAdministerRemoteSearch plug-in configures remote search groups as specified in search network.

Property	Default Value	Description
env.search_network.operation	REGISTER	Valid values: • REGISTER • UNREGISTER
env.search_network.dest_node_names	NODE1	Comma-separated list of nodes from which search groups need to be imported from.
env.search_network.NODE1		Comma-separated list of nodes from which search groups need to be imported to the NODE1 node.

PTSFAdministerSearch

Application class - PTEM_CONFIG:PTSFAdministerSearch

The PTSFAdministerSearch plug-in deploys, undeploys, and schedules index generation for search definitions/categories from a selected list.

Property	Default Value	Description
env.ptsf_selection_type	Global	Valid values: • All • Global • List
env.ptsf_include_definitions	True	Comma separated list of search category names to be included. You can use % as an operator.

Property	*Default Value*	*Description*
env.ptsf_exclude_definitions	True	Comma separated list of search category names to be excluded. You can use % as an operator.
env.ptsf_check_audit_errors	True	If true check for access to query/ connected query, or invalid objects and stop if errors found.
env.ptsf_admin_operations	DEPLOY, INDEX	Comma-separated list of administration operations. Valid values: • Deploy • Index • Undeploy • Deploy,Index You can use any combination of comma-separated values.
env.ptsf_index_all_languages	False	If true the schedules are created to index all languages.
env.report_schedule_status_after_ minutes		Maximum minutes to wait before reporting scheduling status. Enter 0 to wait till finish; leave blank to skip report.
env.ptsf_schedule_on_server		Specify the Process Scheduler to run the indexing on. Leave this blank to use master scheduler.

PTSFCleanupDeploymentData

Application class: PTEM_CONFIG:PTSFCleanupDeploymentData

PTSFCleanupDeploymentData deletes deployment configuration data related to Search Framework from database.

Property	*Default Value*	*Description*
env.ptsf_cleanup_operations	CLEANUP	Indicates the operation to perform.
env.ptsf_cleanup_remotegroups	true	Indicates whether there are remote search groups to clear from the database in clustered environments.

PTSFConfigureSrchInstance

Application class: PTEM_CONFIG:PTSFConfigureSrchInstance

The PTSFConfigureSrchInstance plug-in configures search instance, nodes, and deploys/undeploys search definitions.

Property	***Default Value***	***Description***
env.ptsf_search_instance	PTSF_SEARCH	The search instance name.
env.search_provider	OS/ES	Name of the search engine. Valid values are OS for OpenSearch and ES for Elasticsearch.
env.search_nodes	1	The number of nodes you want to create in the search engine cluster. If you create multiple nodes, then you will see additional properties specific to each node, such as, env.node1_search_host, env.node2_search_host, env.node3_search_host.
env.node1_search_host	@searchhost@	Search engine host name.
env.node1_search_port	9200	Search engine port. The default port number is 9200.
env.node1_search_use_ssl	False	SSL option.
env.node1_search_admin_user		Search engine administrator user name.
env.node1_search_admin_password		Search engine administrator password.
env.node1_search_read_user		User name for query service.
env.node1_search_read_password		Password for query service.
env.insights_host	ss.us.oracle.com	The server name for the host where OpenSearch Dashboards/Kibana is running.
env.insights_port	3456	The port on which OpenSearch Dashboards/Kibana listens for requests. Enter the value supplied when installing the search components DPK. The default you see during installation is 5601.

Property	*Default Value*	*Description*
env.insights_use_ssl	false	Flag to determine whether the configuration is secure or non-secure for OpenSearch Dashboards/Kibana. Valid values are: • False — indicates non-secure (HTTP) configuration • True — indicates secure (HTTPS) configuration
env.logstash_host	dd.us.oracle.com	The server name for the host where Logstash is running.
env.logstash_port	1234	The port on which Logstash listens for requests. Enter the value supplied when installing the search components DPK. The default you see during installation is 9800.
env.logstash_use_ssl	false	Flag to determine whether the configuration is secure or non-secure for Logstash. Valid values are: • False — indicates non-secure (HTTP) configuration • True — indicates secure (HTTPS) configuration
env.search_call_back_user	@userid@	Call back user name.
env.search_call_back_password		Call back password.
env.ps_search_administrator_user	@userid@	Search administrator user name.
env.ps_search_developer_user	@userid@	Search developer user name.
env.gateway_host	@host@.@domain@	Gateway host.
env.gateway_port		Gateway port.
env.gateway_ssl_port		Gateway SSL port.

Property	*Default Value*	*Description*
env.use_ssl_gateway	False	Flag to determine whether the configuration is secure or non secure. Valid values: • False - indicates non secure (http) configuration • True - indicates secure (https) configuration
env.default_local_node	@nodename@	Default local node.
env.enable_global_menu_search	ALL	Enables global search and menu search for All, Local or comma separated list of portals.

PTSFMonitorConfiguration

Application Class: PTEM_CONFIG:PTSFMonitorConfiguration

The PTSFMonitorConfiguration plug-in configures the monitoring server, deploys the PeopleSoft Health Center dashboards, and deploys the system monitoring and search index metrics dashboards to PeopleSoft Insights.

The configuration parameters are similar to the Monitoring Server page in PIA (**PeopleTools** > **Search Framework** > **Administration** > **Monitoring Server**).

Property	*Default Value*	*Description*
env.monitor_es_host		Sets the host name of the monitoring search server.
env.monitor_es_port		Sets the port of the search monitoring server.
env.monitor_es_ssl	False	Flag to determine whether the monitoring search server configuration is secure or non secure. Valid values: • False - indicates non-secure (http) configuration. • True - indicates secure (https) configuration.
env.monitor_es_user	osadmin	Search engine administrator user name.

Property	*Default Value*	*Description*
env.monitor_es_password		Search engine administrator password. The password should be encrypted.
env.monitor_kibana_host		Sets the host name of the monitoring PeopleSoft Insights server.
env.monitor_kibana_port		Sets the port of the PeopleSoft Insights server.
env.monitor_kibana_ssl		Flag to determine whether the PeopleSoft Insights server configuration is secure or non secure.
env.scan_interval	5	Sets the time interval (in seconds) to retrieve system metrics from search server.
env.purge_retention	1	Sets the number of days to store the system and indexing metrics.
env.enable_index_metrics	No	Flag to determine whether index metrics should be collected or not.
env.enable_system_metrics	No	Flag to determine whether system metrics should be collected or not.
env.dashboard_selection	ALL	Specifies the dashboards to be deployed. The valid values are: • ALL • LIST
env.dashboard_list		Comma separated list of PeopleSoft Health Center dashboards to be deployed to PeopleSoft Insights.

PTWebProfileConfig

Application Class: PTEM_CONFIG:PTWebProfileConfig

The PTWebProfileConfig plug-in sets the supported values of the web profile. The property tables are categorized by the page they appear on in the Web Profile component. For more details on each web profile setting, see “Configuring Web Profiles” (Portal Technology).

Note: Not every web profile setting that you see on the web profile configuration pages in PIA is accessible in the configuration plug-in.

General Tab Properties

Property	*Default Value*	*Description*
env.webprofilename	PROD	Sets the web profile name to be modified.
env.descr		Sets the Description Property
env.authtokendomain	example.com	Sets the Authentication Domain property.
env.helpurl	http://www.example.com	Sets the Help URL property.
env.compressresponse	N	Sets the Compress Responses property.
env.compresscachefiles	N	Sets the Compress Response References property.
env.compressmimetypes	application/x-javascript	Sets the Compress MIME Types property.
env.compressquery=N	N	Sets the Compress Query property.
env.saveconfirmdisplay	3111	Sets the Save Confirmation Display Time property. (Milliseconds)
env.enableprocesswait	N	Sets the Enable Processing Message property.
env.enablenewwindow	N	Sets the Enable New Window property.
env.enableppmagent	N	Sets the Enable PPM Agent property (enable PeopleSoft Performance Monitor agent).
env.ppmmonitorbuffsize	51211	Sets the PPM Monitor Buffer Size property (PeopleSoft Performance Monitor buffer size).

Property	*Default Value*	*Description*
env.threaddelay	1111	Sets the Single Thread Delay property.
env.physicalpath		Sets the Non-standard Base Path property.
env.reportrepositorypt		Sets the Report Repository Path property.
env.enablereportrepos	Y	Sets the Enable Report Repository property.
env.compressreports	A	Sets the Compress Report Output property.
env.enableprint	Y	Sets the Enable Print property.

Security Tab

Property	*Default Value*	*Description*
env.useridcookieage	7	Sets the Days to Autofill User ID property.
env.viewfilettl	0	Sets the View File Time to Live property.
env.portalusehttp	N	Sets the PIA use HTTP Same Server (PeopleSoft Pure Internet Architecture use HTTP same server) property.
env.allowunregcontent	Y	Sets the Allow Unregistered Content property.
env.sslrequired	N	Sets the Secured Access Only property.
env.usesecurecookiessl	Y	Sets the Secure Cookie with SSL property.
env.warningtimeout	1011	Sets the Inactivity Warning property.
env.sessiontimeout	1211	Inactivity Logout property.

Property	*Default Value*	*Description*
env.authuserhttpintrvl	0	Sets the HTTP Session Inactivity property.
env.timeoutwarnscript	WEBLIB_TIMEOUT.PT_ TIMEOUTWARNING.FieldFormula. IScript_TIMEOUTWARNING	Timeout Warning Script property.
env.bypasssignon	N	Sets the Allow Public Access property.
env.defaultuserid		Sets the User ID property for public access.
env.defaultpwd		Sets the Password property for public access.
env.publicsesstimeout	0	Sets the HTTP Session Inactivity property for public access.
env.tuxnetdistimeout	0	Sets the Disconnect Timeout property.
env.tuxedosendtimeout	51	Sets the Send Timeout property.
env.tuxedorcvtimeout	1311	Sets the Receive Timeout property.
env.defaultxmllinkuse=		Sets the User ID property.
env.defaultxmllinkpwd		Sets the Password property.
env.xmllnkhttpsamserv	N	Sets the XML Link Use HTTP Same Server property.

Virtual Addressing Tab

Property	*Default Value*	*Description*
env.relativeurl	Y	Sets the Generate Relative URLs property.
env.defaultscheme		Sets the Protocol property.

Property	*Default Value*	*Description*
env.pswebservername		Sets the Name property.
env.defaultport	0	Sets the Port property.
env.scheme		Sets the Protocol (RPS) property.
env.hostname		Sets the Host (RPS) property.
env.httpport		Sets the HTTP (RPS) property.
env.httpsport		Sets the HTTPS (RPS) property.

Cookie Rules Tab

Property	*Default Value*	*Description*
env.cookiepattern	*abcd*	Sets the Cookie Pattern property.
env.forwarddomain	test	Sets the Cookies Passed to Server property.
env.blockdomain	test	Sets the Cookies Not Passed to Server property.
env.deleteonlogout	Y	Sets the Delete Cookie on Logout property.
env.proxied	Y	Sets the Proxied property.
env.secure	N	Sets the Secure property.

Authorized Site Tab

Property	*Default Value*	*Description*
env.pttokendomaincomp	N	This parameter enables or disables the ps-token check across domains in a cluster setup.

Caching Tab

Property	*Default Value*	*Description*
env.enablebrowsercache	Y	Sets the Cache Generated HTML property.
env.maxsavedstate	6	Sets the Number of States Supported property.
env.statusblocktimeout	0	Sets the State Discard Interval property.
env.portalcachehpbrow	N	Sets the Cache Homepage property.
env.portalhpstaleint	1211	Sets the Homepage Stale Interval property.
env.useragentid		Sets the User Agent ID
env.cachehomepage		Sets the Cache Homepage property.
env.portalusecacheprox	N	Sets the Cache Proxied JavaScripts property.
env.portalcacheobjects	N	Sets the Cache Portal Objects property.
env.portalcachestale	86399	Sets the Cache Stale Interval property.
env.cachetargetcontent	N	Sets the Cache Target Content property.
env.cachemenuonwebsrvr	N	Sets the Cache Menu property.
env.cachepurgeallhitct	1011	Sets the Cache Purge All Hit Count property.
env.imagedirphys	/cache	Sets the Image Directory property.
env.imagedirweb	/cache	Sets the Image Web Directory property.
env.cssdirphys	/cache	Sets the CSS Directory (cascading style sheet directory) property.

Property	*Default Value*	*Description*
env.cssdirweb	/cache	Sets the CSS Web Directory property.
env.enablenoversion	Y	Sets the Copy Image/CSS (No Versioning) property.
env.chartdirphys	/cache/chart	Sets the Chart Directory property.
env.chartdirweb	/cache/chart	Sets the Chart Web Directory property.
env.jsdirphys	/cache	Sets the JavaScript Directory property.
env.jsdirweb	/cache	Sets the JavaScript Web Directory property.
env.pt_cache_search	N	Sets the Enable Caching property.
env.pt_cache_search_ws	N	Sets the Enable Caching on Web Browser property.

Debugging Tab

Property	*Default Value*	*Description*
env.ppmtrace	N	Trace Monitoring Server
env.traceppmagent	N	Trace PPM Agent
env.connectinfo	Y	Show Connection Information
env.enabletrace	Y	Show Trace Link at Signon
env.debugshowlayout	Y	Show Layout
env.debugoverlap	Y	Show Overlapping Fields
env.debuginliness	Y	Show StyleSheet Inline HTML
env.debuginlinejs	Y	Show JavaScript Inline HTML

Property	*Default Value*	*Description*
env.testing	N	Generate HTML for testing.
env.enabledebugdumpfl	N	Write dump file.
env.debugsavefile	Y	Create file from PIA HTML page.

Look and Feel Tab

Property	*Default Value*	*Description*
env.startpage	WEBLIB_PTBR.ISCRIPT1. FieldFormula.IScript_StartPage	Sets the Page (Start) property.
env.startpagescript		Sets the Override property.
env.expirepage	start.html,start.wml	Sets the Page (Expire) property.
env.expirepagecontname	PT_EXPIRE,PT_EXPIRE_WML env. exceptionpage=start.html,start.wml	Sets the Content Name property.
env.pageleterrorpage		Sets the Pagelet Error Page property.
env.portaldtlerrpage		Sets the Portal Detail Error Page property.
env.portaltrgterrpage		Sets the Portal Target Error Page property.
env.mcfauthfailurepage		Sets the MCF Auth Failure Page (multichannel framework authorization failure page) property.
env.authtokenablepage		Sets the Auth Token Enable Page (authorization token enable page) property.
env.signontracepage		Sets the Enable Trace Page property.
env.cookierequiredpage		Sets the Cookies Required Page property.

Property	*Default Value*	*Description*
env.sslrequiredpage		Sets the SSL Required Page property.
env.userprofilepage		Sets the User Profile Page property.
env.signonpage	signon.html	Sets the Signon Page property.
env.signonresultdocpag	start.html	Sets the Signon Result Doc Page (signon result document page) property.
env.signonerrorpage	signin.html	Sets the Signon Error Page property.
env.logoutpage	signin.html	Sets the Logout Page property.
env.passwordexpiredpag	start.html	Sets the Password Expired Page property.
env.passwordwarnpage	start.html	Sets the Password Warning Page property.
env.chgpwdonexpire	MAINTAIN_SECURITY.EXPIRE_CHANGE_PSWD.GBL	Sets the Change Password on Expire property.
env.chgpwdonwarn	MAINTAIN_SECURITY.CHANGE_PASSWORD.GBL	Sets the Change Password On Warning property.
env.languagesupport	zh_HK=ZHT1	Sets the Language Support property.

Custom Properties Tab

Property	*Default Value*	*Description*
env.updateonlycustomproperty	N	This flag if set to true will update the custom properties (based on values of parameters below) and will exit without performing any other configuration. Used in the scenario where you only want to update a custom property.

Property	*Default Value*	*Description*
env.propertyname		Custom property name. For example: EnablePCModal PPMConsole checkForDuplicateCookies
env.validationtype	1	Validation type: 1= Boolean 2 = Number 3 = String
env.longvalue	false	Custom property value.

PTWorkflowEmailConfiguration

Application class - PTEM_CONFIG:PTWorkflowEmailConfiguration

Used to configure email details for the logged-in user. On PIA, you can navigate to the Edit Email Addresses link using **PeopleTools** > **Security** > **User Profiles** > **User Profiles**, click the General tab, and then click the Edit Email Addresses. For more information on setting email addresses, see “Setting General User Profile Attributes” (Security Administration).

Property	*Default Value*	*Description*
env.email_id_types		Valid values are: • BB — Blackberry • BUS • HOME • OTH • WORK
env.email_ids		If you enter multiple email IDs, separate email IDs by using comma.
env.primary_email		Primary email account.

PTWorkflowProfileUpdate

Application class - PTEM_CONFIG:PTWorkflowProfileUpdate

Used to configure the routing preferences for the logged-in user. On PIA, you can navigate to the Workflow tab using **PeopleTools** > **Security** > **User Profiles** > **User Profiles**, and then click the Workflow tab. For more information on setting routing preferences, see “Specifying Workflow Settings” (Security Administration).

Property	***Default Value***	***Description***
env.worklists_user	Y	Flag to set routing preference to worklist, that is, the system will deliver work items to a worklist.
env.email_user	Y	Flag to set routing preferences to an email address, that is, the system will deliver work items to an email address.

PTWorkflowSystemConfiguration

Application class - PTEM_CONFIG:PTWorkflowSystemConfiguration

Used to configure system wide route settings. On the PIA, you can navigate to the Worklist System Defaults page by using **PeopleTools** > **Workflow** > **Defaults and Messages** > **Set Workflow System Defaults**. For more information on Worklist System Defaults, see “Setting Workflow Defaults” (Workflow Technology).

Property	***Default Value***	***Description***
env.worklists_active	Y	Flag to enable worklists for your users.
env.email_active	Y	Flag to enable the system to send email to workflow users.
env.hr_installed		Flag to specify whether you are using PeopleSoft HCM applications.

PTImportCertificatestoDB

Application class - PTEM_CONFIG:PTImportCertificatestoDB

The PTImportCertificatestoDB plug-in enables you to import digital certificates (SSL) into database.

Note: The PTImportCertificatestoDB plug-in can be executed only through command line.

You can enter either the certificate data as a string (env.cert_string) or the path to the certificate file containing PEM formatted data (env.cert_file_path). If both the parameter values are provided, then env.cert_string takes precedence.

Property	***Default Value***	***Description***
env.cert_type	ROOT,NODE	Defines the type of certificate whether it is RootCA or Remote Node. If it is RootCA, enter certificate type as ROOT, else enter NODE for public certificate.
env.cert_alias		Enter an alias name for the RootCA and local node name for Remote Node in Alias.
env.cert_issuer_alias		Issuer alias name will be same as the RootCA alias name.
env.cert_string		Enter certificate PEM data as a string.
env.cert_file_path		Enter the path to certificate file containing certificate PEM formatted data.

Executing Configuration Plug-ins

This section describes the topics related to executing configuration plug-ins.

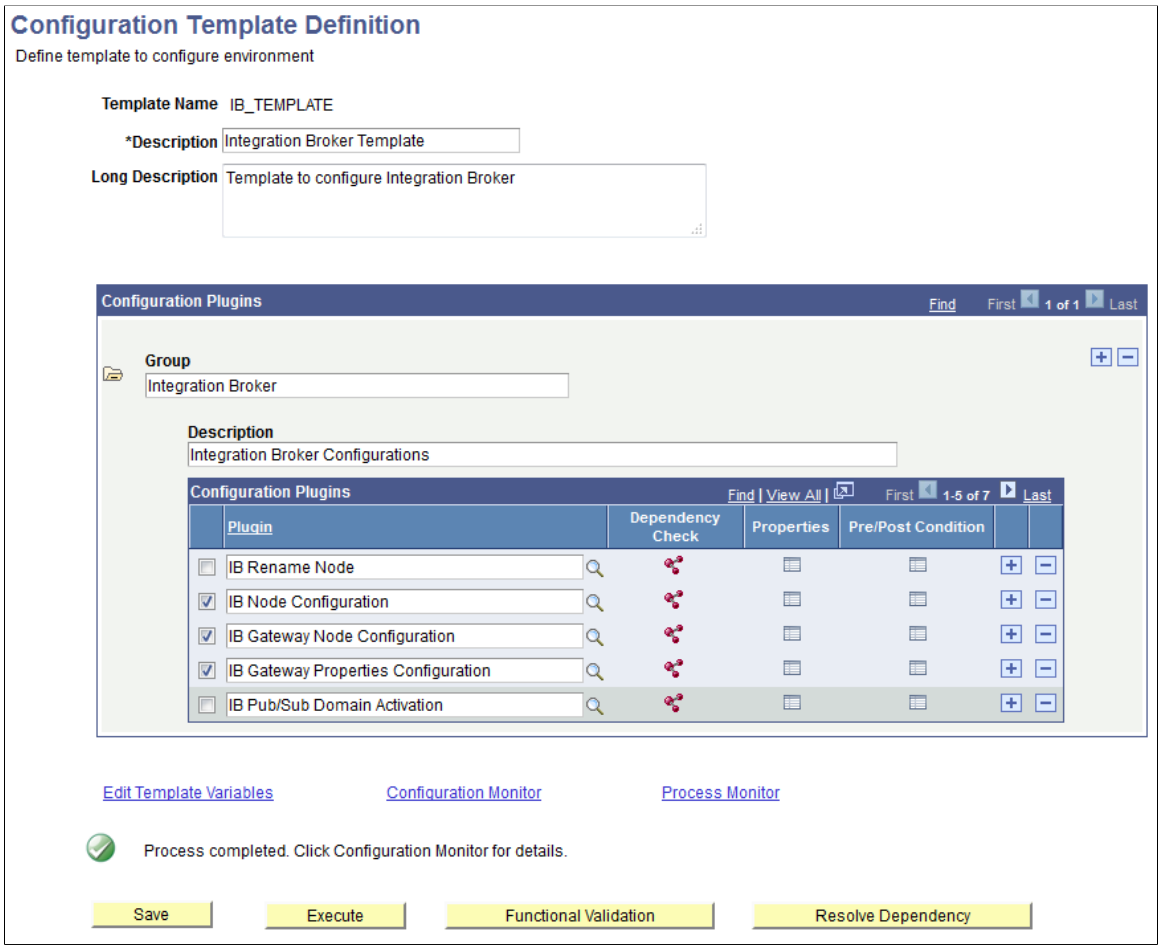
Executing a Template

Use the Configuration Template Definition page to add or delete plug-ins, check the plug-in dependency, execute a configuration, and monitor a configuration.

Access the Configuration Template Definition page using the following navigation path:

PeopleTools > **Automated Config Manager** > **ACM Templates** > **Define ACM Templates**

This example illustrates the fields and controls on the Configuration Template Definition page. You can find definitions for the fields and controls later on this page.



Field or Control	*Description*
Template Name	The name of the template that is entered while creating a template. The template name cannot be edited.
Group	A group enables you to create a functional grouping of plug-ins related to a configuration. For example, you can group plug-ins related to Integration Broker configuration. You may add multiple groups In a template to configure different products. For example, you can add a group of plug-ins that configure the Search Framework in addition to Integration Broker.
Plugin	Use the check box to select a plug-in that you want to include in the configuration program run.

Field or Control	*Description*
 **Dependency Check**	Click the Dependency Check icon to review whether any dependencies exist for the current plug-in. If any dependency exists, you must execute the plug-in before you run the current configuration plug-in.
 **Properties**	Click the Properties icon to review the values that are defined for each property and edit if needed. If you do not edit any of the properties, the default property values will be used in the configuration program run.
Pre/Post Condition	Enables you to add control over the configuration template execution by specifying conditions that may need to be in place prior to a configuration plug-in execution or by specifying how the execution should behave after it has run. This is described in more detail in a subsequent section. For more information on handling errors and setting preconditions and post-conditions, see Handling Errors During Plug-in Execution.
Edit Template Variables	Click the Edit Template Variables link to review the property values that will be used at run time. The template variables are common to all the groups within a template.
Configuration Monitor	After clicking the Execute button to commence a configuration program run, click the Configuration Monitor link to review the configuration program status. See Monitoring a Configuration.
Process Monitor	Click the Process Monitor link to review the process status. The Process Monitor link is active only if the processing mode is Scheduled. See Scheduling An Automated Configuration in PIA.
Save	If you modify a template, save the template before running the configuration program.
Execute	Click to execute a configuration program run. After clicking Execute and allowing processing to complete, the system displays the overall status of the request using an icon, such as the Success icon or the Error icon, with the text: *Process completed. Click Configuration Monitor for details.* The message that you see depends on the processing mode you selected in the ACM Options page.

Field or Control	*Description*
Functional Validation	Click to verify the subsystems related to the product configuration. This does not report as to whether the configuration process succeeded. It reports whether the various pieces of the system interrelate successfully. You can access the report using the Configuration Monitor. For more information on this option, see Running Functional Validation. After clicking Functional Validation and allowing processing to complete, the system displays the overall status of the request using an icon, such as the Success icon or the Error icon, with the text: *Process completed. Click Configuration Monitor for details.*
Resolve Dependency	Click to resolve dependencies between plug-ins and plug-in groups within the configuration template definition. While this is a dynamic process, meta data related to plug-in dependency needs to have been established. For more information on working with plug-in dependency, see Managing Configuration Plug-in Dependency.

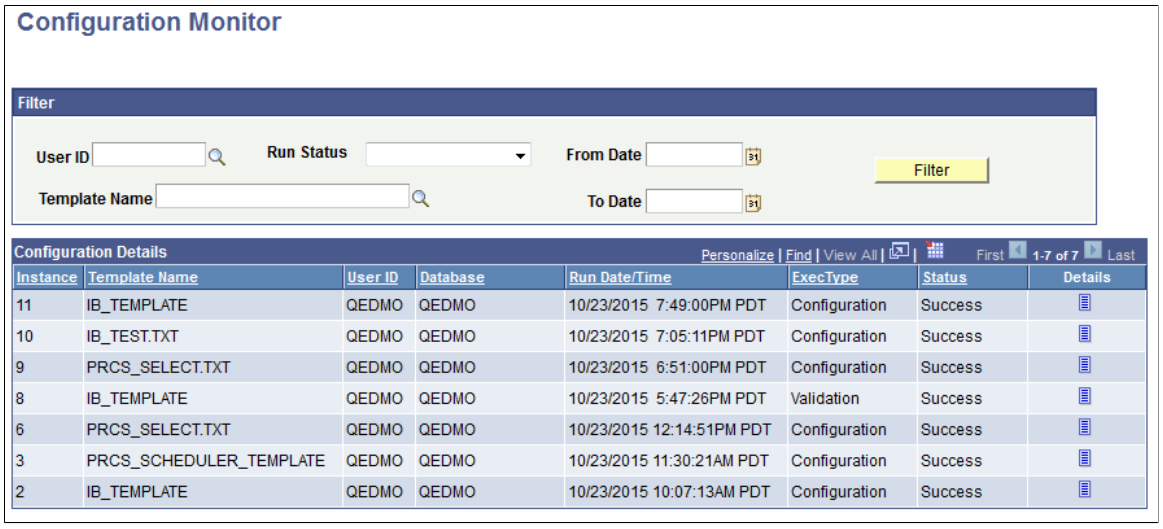
Monitoring a Configuration

Use the Configuration Monitor page to review the status of the configuration template and to review the details.

Access the Configuration Monitor page using the following navigation path:

PeopleTools > **Automated Config Manager** > **ACM Configuration Monitor**

This example illustrates the fields and controls on the Configuration Monitor page. You can find definitions for the fields and controls later on this page.



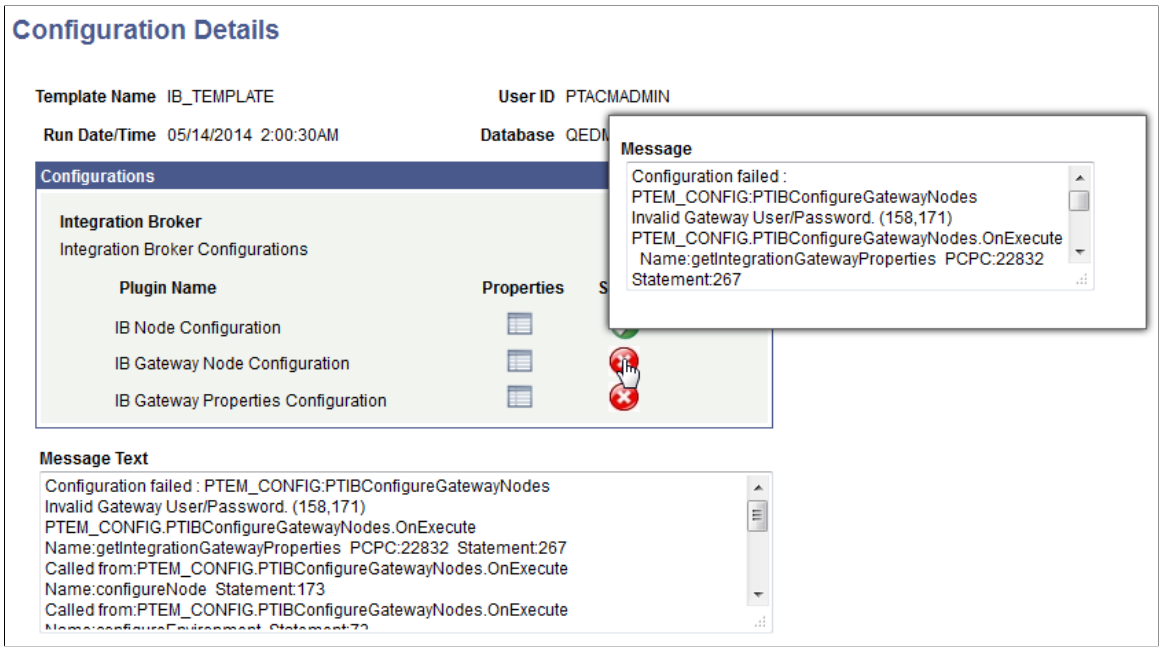
Field or Control	*Description*
User ID	Select a user ID by which you want to filter the search results.
Run Status	Select either Error or Success to filter the search results.
Template Name	Select a template name by which you want to filter the search results.
From Date	Enter a date range by which you want to filter the search results.
To Date	Enter a date range by which you want to filter the search results.
Filter	Click the Filter button to show results based on the filter criteria.
Details	Click the Details icon to view the configuration details.
Instance	Indicates the sequence in which the request was run.
Template Name	Identifies the template definition name that was the subject of the request.
User ID	User ID used to run the request.
Database	Database on which the request was applied or run.
Run Date/Time	Indicates the date and time the process ran.
ExecType	Indicates the type of process initiated: Configuration (clicking Execute for example) or Validation (clicking Functional Validation for example).
Status	Indicates the outcome of the process: Success or Error.
Details	Click to view the Configuration Details page, which displays information for the selected process run.

Reviewing the Configuration Details

Use the Configuration Details page to review the configuration details after running a template and to review the errors that caused a configuration to fail.

To view the Configuration Details page, click the Details icon on the Configuration Monitor page.

This example illustrates the fields and controls on the Configuration Details page. You can find definitions for the fields and controls later on this page.



Field or Control	*Description*
Success Status icon	Indicates a successful configuration.
Properties icon	Click the Properties icon to display a read-only listing of property values for the selected plug-in.
Error Status icon	Hover on the error status icon to view details of the failed configuration.
	Indicates that the verification of configuration failed.

Running Functional Validation

Configuring all of the elements of the various products within the PeopleTools infrastructure can become complex when you consider the dependency a product may have with another product. For example, in order to configure the PeopleSoft Search Framework, you must first configure Integration Broker. Likewise, even within a product itself, there are various elements that depend on each other. It is not only required that these elements be configured, but it is also important that these elements are performing successfully. Running the functional validation test ensures that a product element is not only configured, but operational too.

For example, with Integration Broker once the gateway configuration is complete, it is expected that the gateway is active and the nodes added to the gateway can be pinged. Building in tests for these types of configurations ensures that the functionality is tested along with automating the configuration.

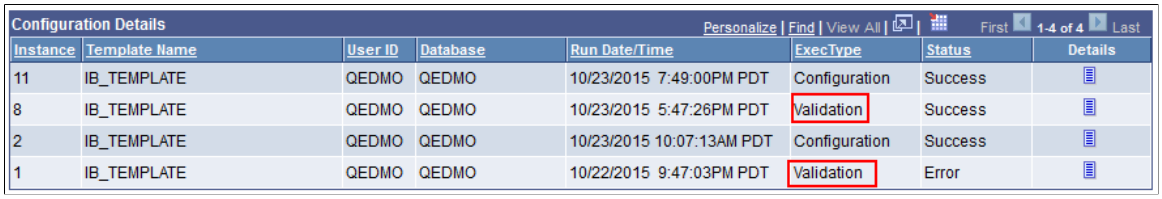
PeopleTools tests selected elements of a configuration as the element is being configured. For example, the automated system adds a node to the Integration Broker gateway, the following checks are made to validate the configuration:

- Ping node: The check is made as to verify if the node present in the database pings successfully.
- Ping gateway: The check is made to ensure that the gateway configured is up and active.
- Ping gateway node: The node added to the gateway is checked to verify if it pings successfully.

You can view the verification report on the Configuration Monitor page with each run of a template. The report consists of the tests performed while configuring the environment to ensure the correctness of the configuration for that template.

You can also verify a configuration after the configuration run by clicking the Functional Validation button on the Configuration Template Definition page. View the verification report on the Configuration Monitor page.

This example illustrates the execution type (ExecType) to identify when looking for functional validation reports.



Configuration Details — Personalize | Find | View All | First 1-4 of 4 Last

Instance	Template Name	User ID	Database	Run Date/Time	ExecType	Status	Details
11	IB_TEMPLATE	QEDMO	QEDMO	10/23/2015 7:49:00PM PDT	Configuration	Success	
8	IB_TEMPLATE	QEDMO	QEDMO	10/23/2015 5:47:26PM PDT	Validation	Success	
2	IB_TEMPLATE	QEDMO	QEDMO	10/23/2015 10:07:13AM PDT	Configuration	Success	
1	IB_TEMPLATE	QEDMO	QEDMO	10/22/2015 9:47:03PM PDT	Validation	Error	

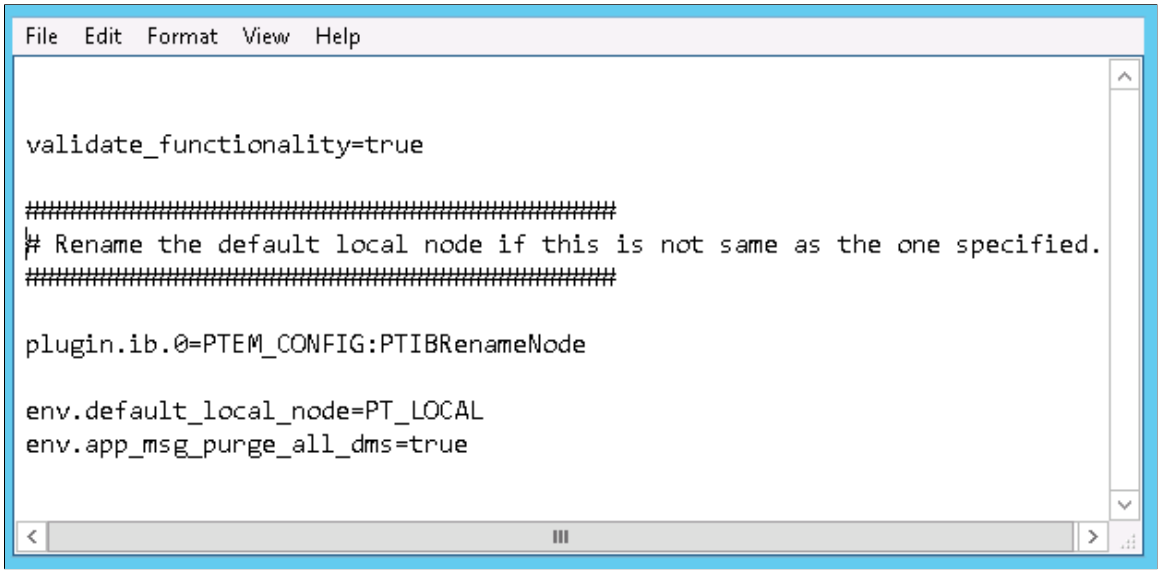
To view the validation report, click the Details icon for your request, which you can identify by the template name and using the ExecType column, which will display *Validation.*

The Functional Validation process can also be performed from command line. In the property file which is provided as input on the command line, enter this property:

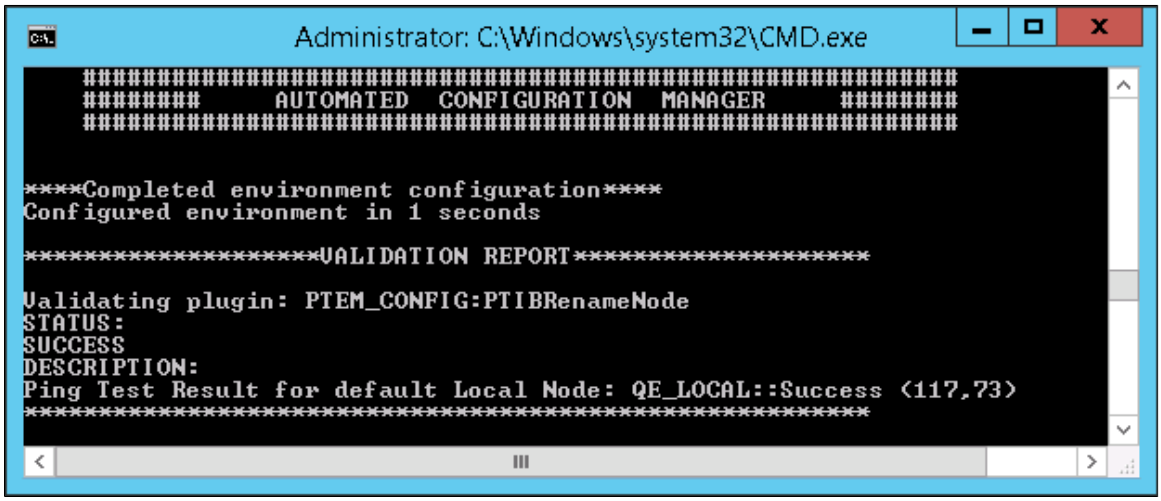
```
validate_functionality = true
```

For example:

This example illustrates adding functional validation to a template file for command line execution.



This example illustrates a sample output after running functional validation at the command line.

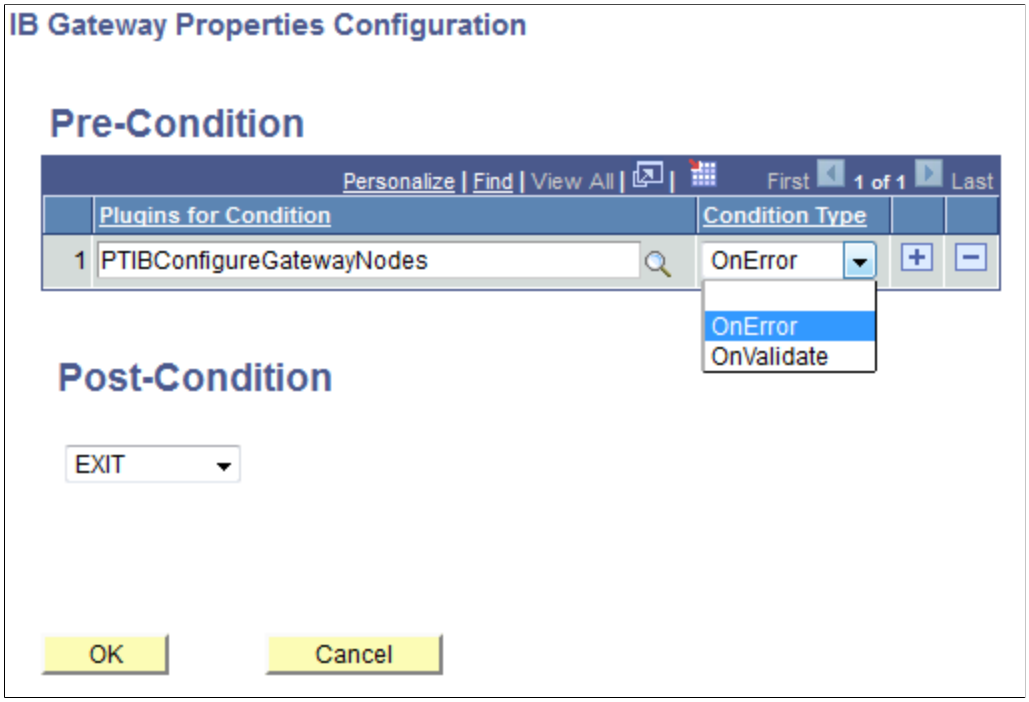


Handling Errors During Plug-in Execution

While the final result of a configuration template run is displayed on the Configuration Template Definition page in PIA and in the console in the command line, PeopleTools provides you options for handling errors that may occur *within* the execution of a configuration template. Using these error handling options provide more control over the configuration as well as providing insight into trouble areas immediately, rather than after combing through log files. The result of the template execution is determined based on the status of each plug-in, and the overall result is displayed in a specific format.

You set the preconditions and post-conditions on the Configuration Template Definition page by clicking the icon in the Pre/Post Condition column. See Executing Configuration Plug-ins.

This example illustrates the options available for setting preconditions and post-conditions.



Setting Preconditions

For preconditions, you can specify a list of plug-ins that should be executed successfully as a precondition. For example, for Search Framework functionality to work, it is mandatory that the dependent Integration Broker configuration should be successfully configured and operational, in which case, the precondition list could contain a set off the Integration Broker plug-ins. If any of the plug-ins in the list fail, the system skips the selected Search Framework plug-in and executes the next plug-in for that template.

You can set these types of preconditions.

- *On Error.*

  With On Error, specify the conditional execution of plug-ins with respect to the execution off other plug-ins within the same template file. For example, if there is a precondition *On Error* for Search Framework plug-ins within a template and the Integration Broker related plug-ins are identified in the list, and the Integration Broker plug-ins *are in the same template,* the Search Framework plug-in execution would be skipped if the Integration Broker plug-in execution fails.

- *On Validate.*

  The precondition On Validate allows you to specify the conditional execution of plug-ins with respect to other plug-ins that *are not part of the same template file.* For example, if On Validate for Search Framework plug-ins identifies a list of Integration Broker-related plug-ins, the Search Framework plug-in execution would be skipped if the specified Integration Broker plug-in functionality fails. For instance, if the validation procedure of pinging a node or pinging the gateway fails, the Search Framework plug-ins would be skipped.

You selections on this page will appear in the template file similar to the following example.

```
plugin.ptsf.1=PTEM_CONFIG:PTSFAdministerSearch
```



```
PreConditionOnError= PTIBConfigureDBNode, PTIBConfigureGatewayNodes
PreConditionOnValidate= PTIBConfigureDBNode, PTIBConfigureGatewayNodes

...
```

Note: If the precondition section is absent for a plug-in, the system assumes there are no dependencies and the configuration execution takes place as if the plug–in is independent.

Setting Post-Conditions

If a configuration template execution should or should not continue based on the status of the current plug-in, you can set a post condition. For example, if you are configuring Integration Broker and you determine that the gateway node configuration is mandatory for proceeding with any further configurations, then select *Exit* from the Post-Condition drop-down list. This means that in the event of failure of the IBGatewayNode plugin execution, the system halts the execution of the current configuration template. By leaving the Post-Condition drop-down list empty, you indicate for the template execution to continue.

The following example illustrates how the post-condition would appear in the template file:

```
plugin.ib.2=PTEM_CONFIG:PTIBConfigureGatewayNodes
PostCondition=EXIT
```

Assign EXIT to PostCondition in the template file to indicate that the template execution should halt if the current plug-in fails. Assign CONTINUE to PostCondition for the plug-in to carry on processing despite failure of the current plug-in, or you may remove or comment the line to achieve the same effect. The default behavior is similar to the case when the keyword CONTINUE is assigned to PostCondition.

Scheduling An Automated Configuration in PIA

Automated configuration management framework provides two processing modes for a configuration process. The two processing modes are:

- Interactive Mode
- Scheduled Mode

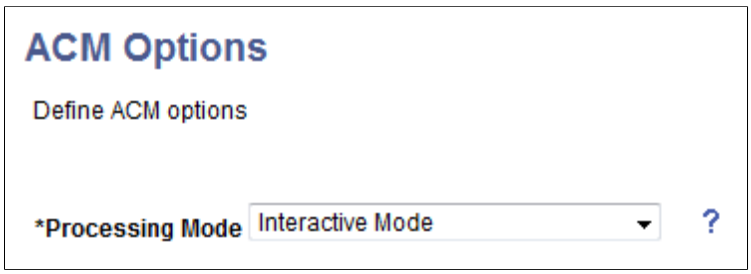
Scheduling a Configuration Process

Use the ACM Options page to choose a processing mode for a configuration process.

Access the ACM Options page using the following navigation path:

PeopleTools > **Automated Config Manager** > **ACM Utilities** > **ACM Options**

This example illustrates the fields and controls on the ACM Options page.



Interactive Mode

In interactive mode, the configuration process commences when the Execute button is clicked on the Configuration Template Definition page and does not allow a user to navigate out of the page.

Scheduled Mode

In scheduled mode, when the Execute button is clicked, the configuration process is scheduled in Process Scheduler and while it is being processed a user can navigate to other pages. If you select scheduled mode, you should ensure that the Process Scheduler domain is running.

Chapter 3

Working with Custom Configuration Plug-ins and Templates

Understanding Custom Configuration Plug-ins and Templates

In addition to the plug-ins and templates delivered by PeopleTools, PeopleTools allows you to create custom configuration plug-ins required for your specific environment and create templates.

A plug-in is an application class that corresponds to a configuration. For example, to configure IB Gateway nodes, the application class, PTIBConfigureGatewayNodes, in the application package PTEM_CONFIG is used. Similar plug-ins can be grouped together, which is called a group. A template is used to organize a group of plug-ins or multiple groups of plug-ins. A template also allows you to edit property values of plug-ins, and run the configuration program through PIA or command line.

Working with Custom Configuration Plug-ins

In addition to the plug-ins delivered by PeopleSoft, PeopleSoft allows you to create custom configuration plug-ins required for your specific environment.

A plug-in is an application class that corresponds to each configuration within a template. For example, to configure IB Gateway nodes, the application class (plug-in) PTIBConfigureGatewayNodes in the application package PTEM_CONFIG

Working with configuration plug-ins includes the following:

- Creating a configuration plug-in.
- Registering a configuration plug-in.
- Adding a configuration plug-in to a template.

  See Creating a Template.
- Migrating a plug-in using ADS definition.

  See Migrating Configuration Data Using ADS.

Creating a Configuration Plug-in

To create a configuration plug-in:

1. Use the PTEM_CONFIG application package to create a plug-in. If you need to use a custom package, ensure that the custom package starts with PTEM_CONFIG.

2. Write the plug-in class. The new plug-in class must extend from the base class — PTEM_CONFIG:EMConfigurationPlugin.

   Override and implement the four base class methods:

   - getProperties - to provide plug-in properties and their details.

     For example:

     ```
     method getProperties
        /+ Returns Array of PTEM_CONFIG:PTEMVariableProperty +/
        /+ Extends/implements PTEM_CONFIG:EMConfigurationPlugin.getProperties ⇒

     +/

        Local array of PTEM_CONFIG:PTEMVariableProperty &propArray = Null;

     ---add the properties as shown below---------------
         &variableProperty = create PTEM_CONFIG:PTEMVariableProperty("property⇒

     _name", "datatype", True/False (is required?), True/False(is password?), ⇒

     "default value", Message set number of property description, message numb⇒

     er, "default message", Null);   &propArray.Push(&variableProperty);  ----⇒

     ------------------------------------------------------------------------⇒

     -----------------------
        Return &propArray;
     end-method;
     ```

     Add the following base class methods:

     - validateVariables - to validate the plug-in property values.
     - configureEnvironment - to configure the environment with the provided values.
     - validateConfigurations - to validate whether the values are correctly configured.

   - dependant_plugins - to provide the names of dependent plug-ins.

     For example:

     ```
     method dependant_plugins
        /+ Returns Array of String +/
        /+ Extends/implements PTEM_CONFIG:EMConfigurationPlugin.dependant_plug⇒

     ins +/
        Local array of string &dependant_array;
        &dependant_array = CreateArray(""); -----add the plugins as comma sepr⇒

     ated values, that this plugin depends on for its execution
        Return &dependant_array;
     end-method;
     ```

   - getPluginHelpMessage - to provide a brief description of a plug-in.

     For example:

     ```
     method getPluginHelpMessage
        /+ Returns PTEM_CONFIG:PTEMHelpMessage +/
        /+ Extends/implements PTEM_CONFIG:EMConfigurationPlugin.getPluginHelpM⇒
     ```



```
essage +/
   Local PTEM_CONFIG:PTEMHelpMessage &tempMessage = Null;
   &tempMessage = create PTEM_CONFIG:PTEMHelpMessage(0, 0, "", Null);   -⇒

--- "add the message set number, message number and default message of th⇒

e help message"
   Return &tempMessage;
end-method;
```

- isInternalPlugin - to differentiate between plug-ins that can be used internally by developers and externally by customers.

  Valid values are:

  - False - specifies that the plug-in can be used by customers, and the plug-in can be added in a template in PIA.
  - True - implies that the plug-in can be used only through command line and will not be available in PIA.

  For example:

```
method isInternalPlugin
   /+ Returns Boolean +/
   /+ Extends/implements PTEM_CONFIG:EMConfigurationPlugin.isInternalPlug⇒

in +/
   Return False;
end-method;
```

Note: When you create a configuration plug-in an application package, the plug-in must be registered through the automated configuration management framework so that it can be included in a template for configuration through PIA. If a plug-in is not registered, it can be used only for execution through command line.

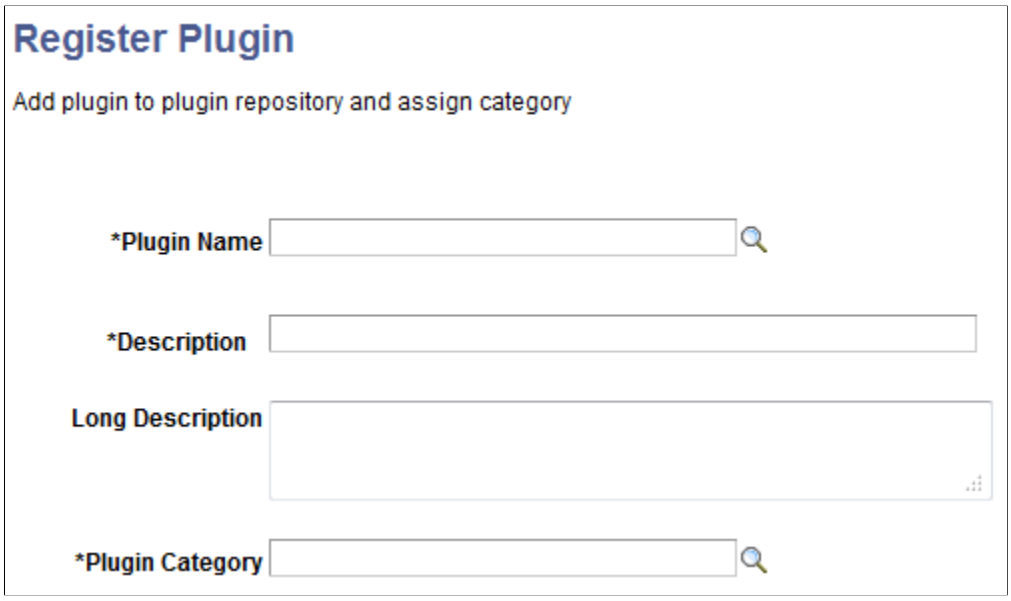
Registering a Configuration Plug-in

Use the Register Plugin page to register a configuration plug-in, which includes assigning a category for the plug-in.

Access the Register Plugin page using the following navigation path:

PeopleTools > **Automated Config Manager** > **ACM Utilities** > **Register Plugin**

This example illustrates the fields and controls on the Register Plugin page. You can find definitions for the fields and controls later on this page.



Field or Control	*Description*
Plugin Name	Select the plug-in that you created. **Note:** If the isInternalPlugin method of a plug-in class is defined as false, the plug-in will be available in the Plugin Name list.
Description	Enter a description for the plug-in.
Long Description	Optionally, you may enter a long description for the plug-in.
Plugin Category	Select a category to which the plug-in will belong. Currently, the following categories are available: • IB • Miscellaneous • Network Setup • Process Scheduler

See Creating a Configuration Plug-in.

Working with Custom Templates

Automated configuration management uses templates to organize configuration plug-ins, edit the configuration property values, and run the configuration process in PIA or the command line. This topic describes the tasks involved with creating and managing your templates.

See Migrating Configuration Data Using ADS.

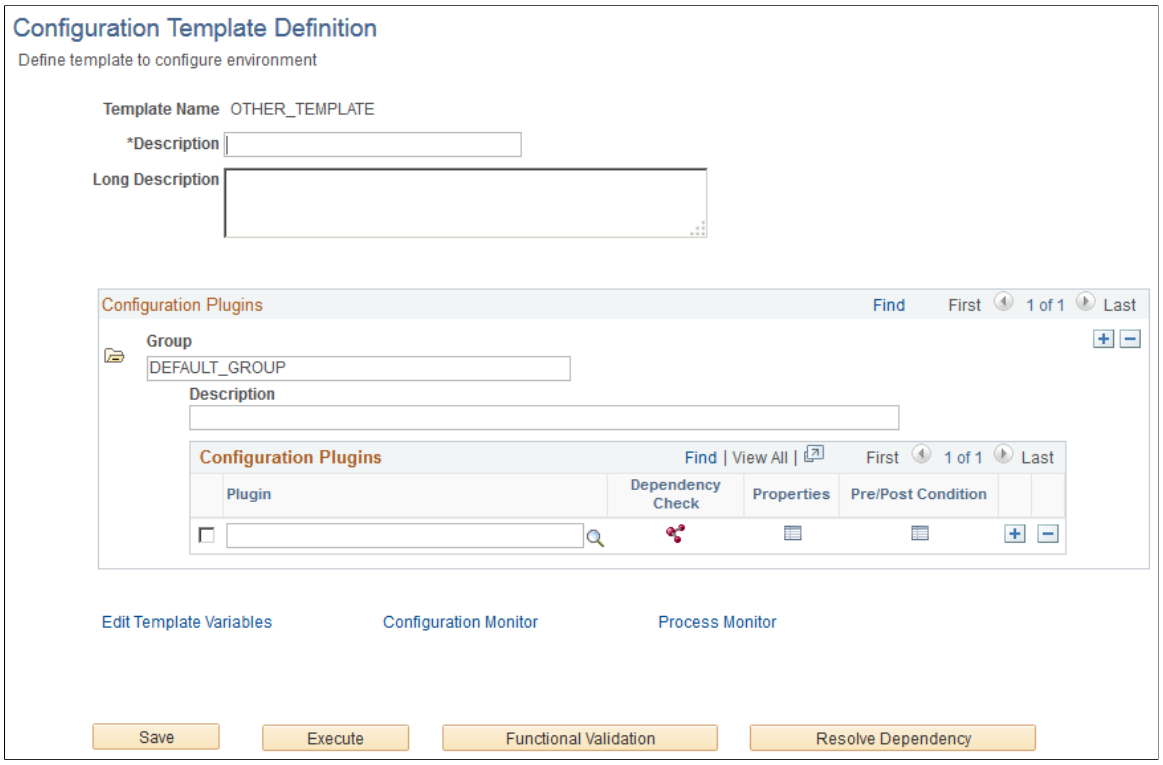
Creating a Template

Use the Configuration Template Definition page to create a new template, which includes adding configuration plug-ins, editing properties, and defining or importing template variables.

Access the Configuration Template Definition page using the following navigation path:

PeopleTools > Automated Config Manager > ACM Templates > Define ACM Templates > Add a New Value

This example illustrates the fields and controls on the Configuration Template Definition page - Creating Template. You can find definitions for the fields and controls later on this page.



Field or Control	*Description*
Template Name	The name of the template entered on the Template Definition page. You cannot edit the template name on this page (Configuration Template Definition page).

Field or Control	*Description*
Description	Enter a short description for the template.
Long Description	Enter a descriptive comment for the template.
Group	Enter a name for the group of plug-ins. A group enables you to create a functional grouping of plug-ins related to a configuration. For example, you can group plug-ins related to Integration Broker configuration. You may add multiple groups In a template to configure different products. For example, you can add a group of plug-ins that configure the Search Framework in addition to Integration Broker. DEFAULT_GROUP is the default name for the first group. When you add subsequent groups, these groups do not display a default name.
Description	Enter a descriptive comment for the group.
Plugin	Use the Look up Plugin button to list the available plug-ins and choose a plug-in. See Working with Custom Configuration Plug-ins on creating a custom plug-in.
Dependency Check	Click the Dependency Check icon to review whether any dependencies exist for the current plug-in. If any dependency exists, you must execute the plug-in before you run the current configuration plug-in.
Properties	Click the Properties icon to review the values that are defined for each property and edit if needed. If you do not edit any of the properties, the default property values will be used in the configuration program run. On the properties window, use the Current values in DB button to view the current value specified for a property of a plug-in in the database. See Editing Configuration Plug-in Properties.
Pre/Post Condition	Enables you to add control over the configuration template execution by specifying conditions that may need to be in place prior to a configuration plug-in execution or by specifying how the execution should behave after it has run. This is described in more detail in a subsequent section. For more information on handling errors and setting preconditions and post-conditions, see Handling Errors During Plug-in Execution.

Field or Control	*Description*
Edit Template Variables	Click the Edit Template Variables link to review the property values that will be used at run time. The template variables are common to all the groups within a template. For information on defining and importing template variables, see Defining Template Variables and Importing Template Variables.
Configuration Monitor	After clicking the Execute button to commence a configuration program run, click the Configuration Monitor link to review the configuration program status. See Monitoring a Configuration.
Process Monitor	Click the Process Monitor link to review the process status. The Process Monitor link is active only if the processing mode is Scheduled. See Scheduling An Automated Configuration in PIA.
Save	Save the template before you execute a configuration program run.
Execute	Click to execute a configuration program run. After clicking Execute and allowing processing to complete, the system displays the overall status of the request using an icon, such as the Success icon or the Error icon, with the text: *Process completed. Click Configuration Monitor for details.* The message that you see depends on the processing mode you selected in the ACM Options page.
Functional Validation	Click to verify the subsystems related to the product configuration. This does not report as to whether the configuration process succeeded. It reports whether the various pieces of the system interrelate successfully. You can access the report using the Configuration Monitor. For more information on this option, see Running Functional Validation. After clicking Functional Validation and allowing processing to complete, the system displays the overall status of the request using an icon, such as the Success icon or the Error icon, with the text: *Process completed. Click Configuration Monitor for details.*

Field or Control	*Description*
Resolve Dependency	Click to resolve dependencies between plug-ins and plug-in groups within the configuration template definition. While this is a dynamic process, meta data related to plug-in dependency needs to have been established. For more information on working with plug-in dependency, see Managing Configuration Plug-in Dependency.

Related Links

Executing a Template

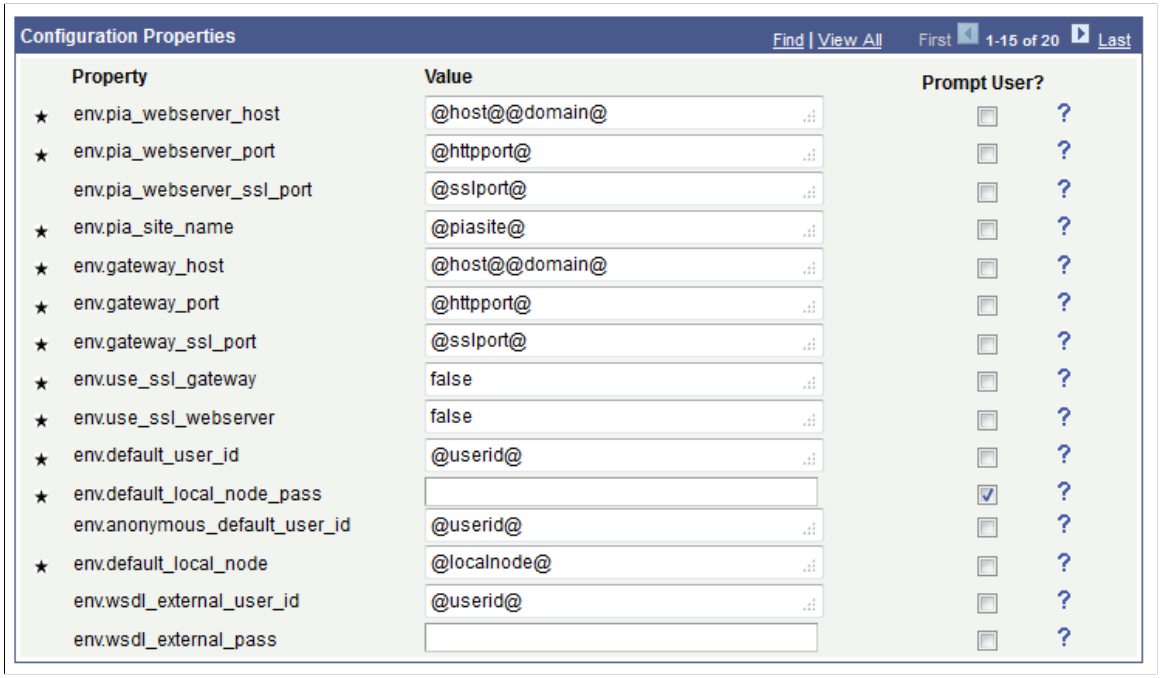
Editing Configuration Plug-in Properties

Use the Configuration Properties page to edit plug-in properties. You define properties and their values for a plug-in when you create the plug-in. When you create a template or run a configuration process, you may edit the property values.

Access the Configuration Properties page using the following navigation path:

PeopleTools > **Automated Config Manager** > **ACM Templates** > **Define ACM Templates** and click the Properties icon for a configuration plug-in that you want to edit.

This example illustrates the fields and controls on the Configuration Properties page. You can find definitions for the fields and controls later on this page.



Note: Property values enclosed within @ indicates that these values are supplied by template variables at run time. To edit a template variable, click the Edit Template Variables link on the Configuration Template Definition page.

Field or Control	*Description*
Required Property icon	Indicates a property that is required and a value must be defined.
Prompt User	Select the Prompt User check box if you want a user to enter a value at run time. Generally, a prompt for a user to enter a value is required for user name or password.
Help icon	Hover over the help icon to view a description of a property.
Current Values in DB	Click the Current Values in DB button to review the current values in the database for the properties. Initially, the values for properties are defined when creating a plug-in. Subsequently, if you modify the values, the modified values are displayed when you click the Current Values in DB button.

Defining Template Variables

You assign a value to a template variable which is in-turn used to assign values for properties in cases where values are recurring in multiple places or plug-ins in a template or template file.

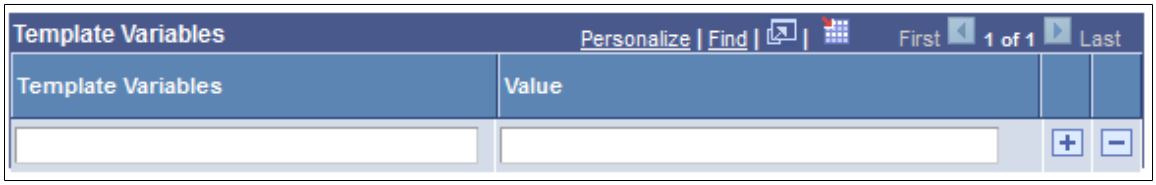
Important! Passwords defined as template variables are *not* encrypted or masked when you create a template in PIA. Therefore, Oracle PeopleSoft recommends that you *do not* define passwords as template variables in order to maintain templates securely.

Use the Template Variables dialog box to assign values to a template variable.

Access the Template Variables dialog box using the following navigation path:

PeopleTools > **Automated Config Manager** > **ACM Templates** > **Define ACM Templates** and click the Edit Template Variables link.

This example illustrates the fields and controls on the Template Variables dialog box.



Note the following when you assign values for template variables:

Template Variable Value	*Description*
@webhist	Prefix a value with @webhist if the value will be supplied from web history. For example: @webhist.WEBSERVERNAME These values are valid only if a user logs on to PIA before running the configuration program using Automated Configuration Manager.
%	Prefix a value with % if the variable is a PeopleSoft system variable that will be supplied at run time. For example: %ToolsRelease

Importing Template Variables

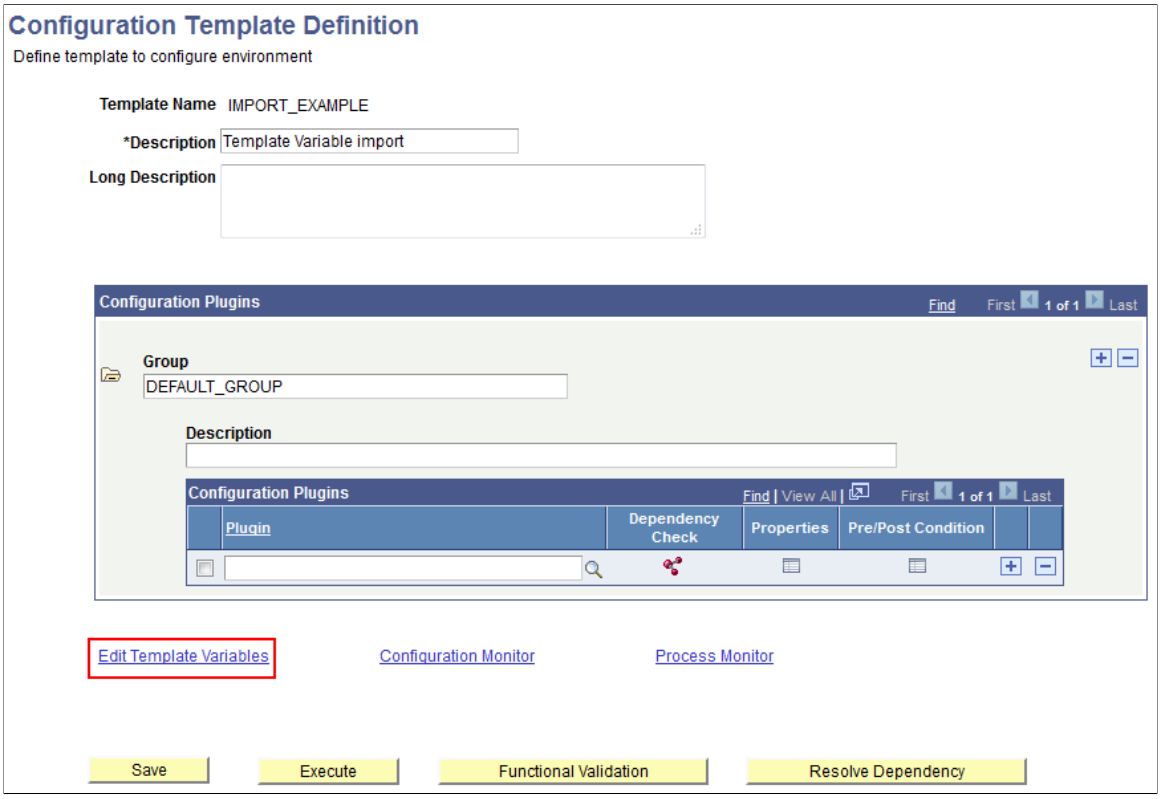
When creating new templates, it can be helpful to import existing template variables from another template if you require them in your new template. In this case, you are copying established and tested template variables from one template to another.

Note: If the target template already has a template variable defined that is imported from another template, then that variable is ignored.

To import a template variable:

1. Open the target template definition, and on the Configuration Template Definition page, click Edit Template Variables.

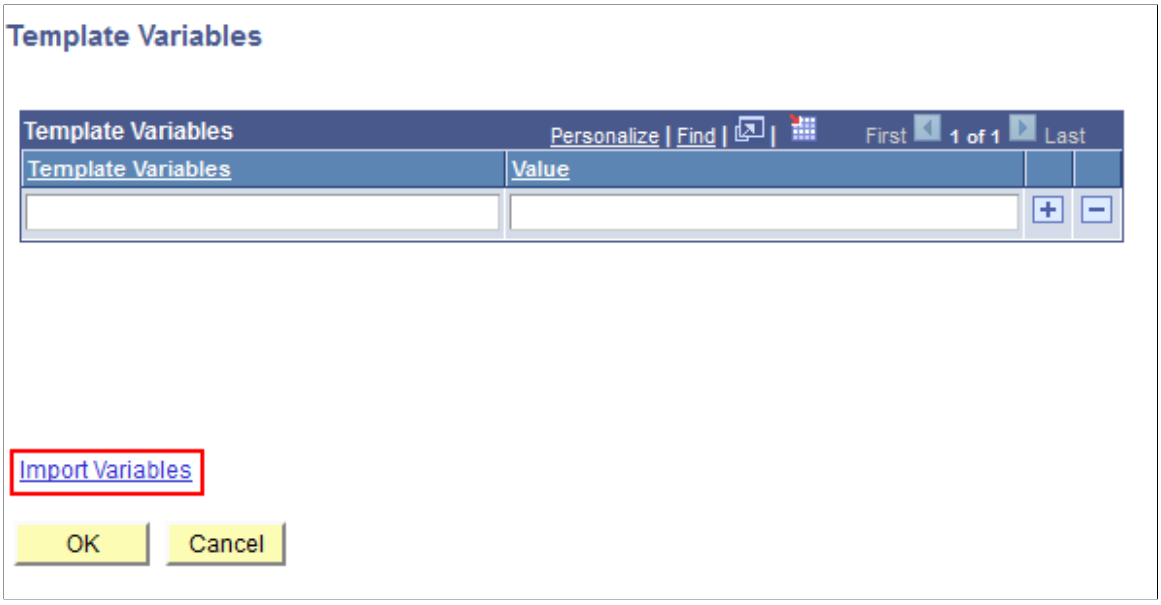
This example illustrates where to find the Edit Template Variables link.



2. On the Template Variables page, click Import Variables.

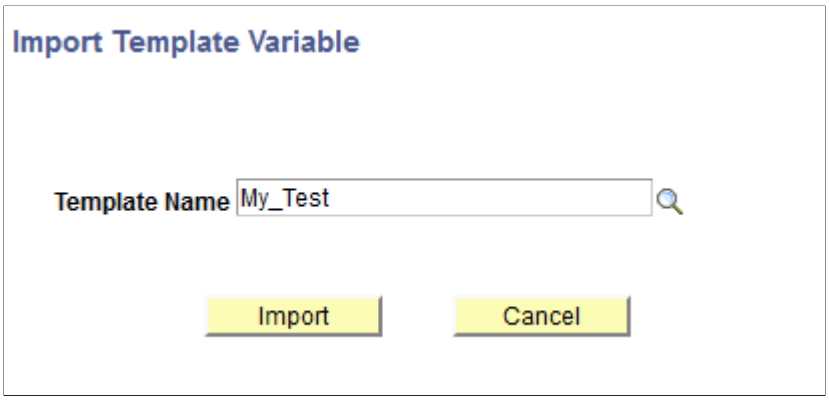
   If the template is new, the Template Variables page will be empty

   This example illustrates an empty template variable page. To populate the page with template variables from an existing template, click Import Variables.



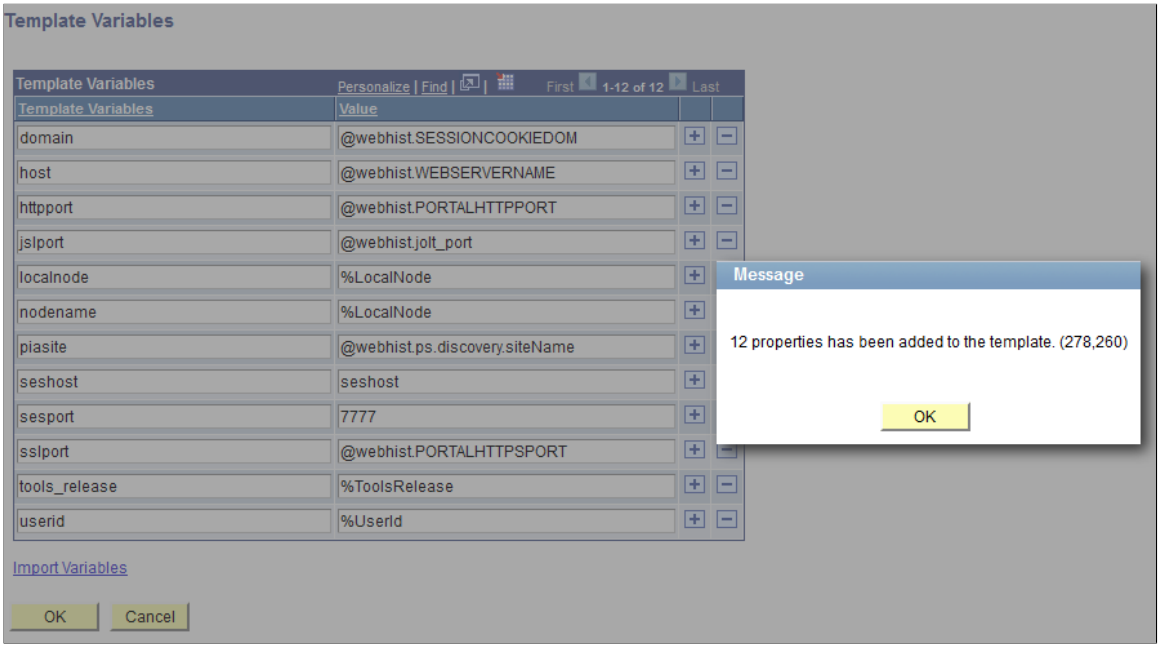
3. On the Import Template Variable page, select the source template name, from which you want to import template variables, and click Import.

   This example illustrates selecting an existing template as a source template from which you can import template variables into a target template.



4. Confirm the template variables that you require have been imported to the target template, and click OK.

   This example illustrates the previously empty template variables page now populated with template variables imported from an existing template.



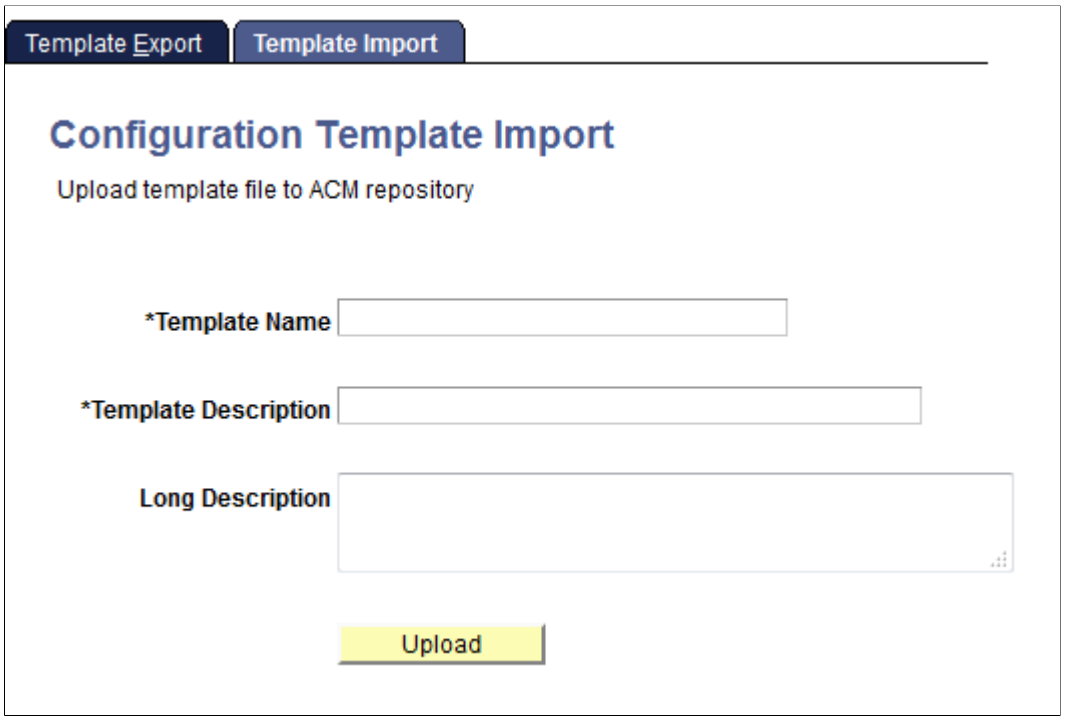
Importing a Template Using PIA

Use the Configuration Template Import page to import a template file.

Access the Configuration Template Import page by using the following navigation path:

PeopleTools > **Automated Config Manager** > **ACM Templates** > **Export/Import ACM Templates**

This example illustrates the fields and controls on the Configuration Template Import page. You can find definitions for the fields and controls later on this page.



Field or Control	*Description*
Template Name	Add a template name to uniquely identify the imported template.
Template Description	Add a description to identify the purpose of the template.
Long Description	If further information is provided to describe the purpose of the template or the configurations it controls, add here.
Upload	Click to upload the template text file. A dialog box appears enabling you to select the file from a network location and upload it.

Note: When a template is imported into an environment (from a text file), it is assumed that all passwords contained in the template file are encrypted, so the decrypt_password property in the template file must be set to True.

Adding a Template to a Permission List

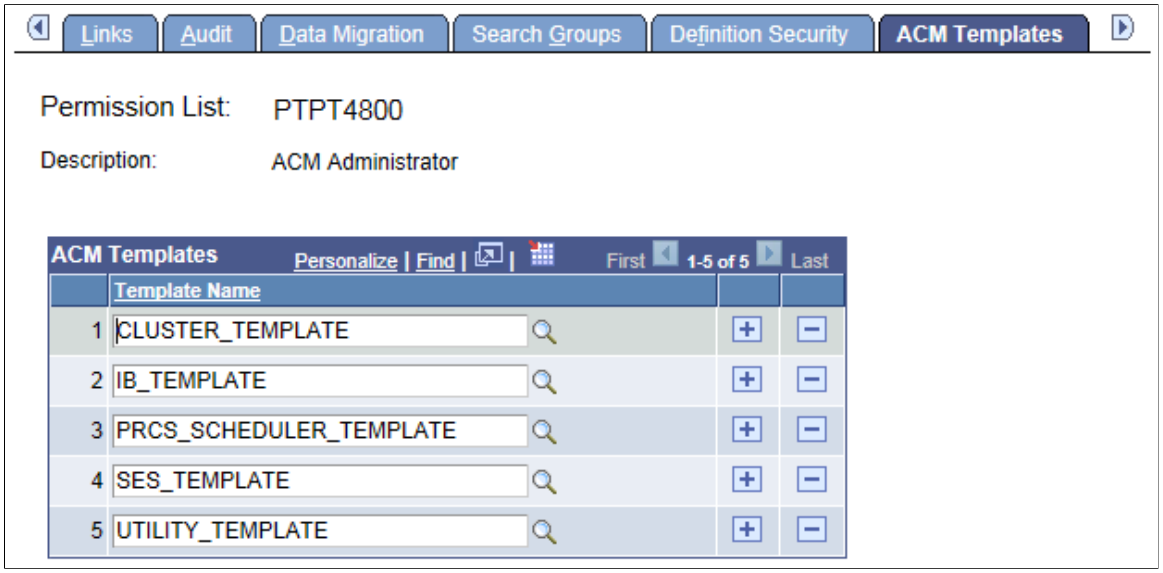
After you create a template, you must add the template to a permission list. PeopleTools provides the PTPT4800 specifically for the purpose of automated configuration.

Use the ACM Templates page to add a template to the PTPT4800 permission list.

Access the ACM Templates page by using the following navigation path:

PeopleTools > **Security** > **Permissions and Roles** > **Permission Lists** and click the ACM Templates tab.

This example illustrates the fields and controls on the Permission Lists - ACM Templates page.



Add a template to the permission list.

Note: If a template is not added to the PTPT4800 permission list, you cannot use the template for a configuration run.

Managing Configuration Plug-in Dependency

This topic describes the features you can use to reflect product dependency dynamically into your configuration templates.

Understanding Resolving Plug-in Dependency Dynamically

Numerous elements in the PeopleSoft system depend on other elements in the system to provide critical functionality. For example, Search Framework cannot operate unless the basic elements of Integration Broker have been configured, therefore Search Framework is dependent on Integration Broker. As such, various plug-ins within a configuration template need to reflect these dependencies in the system.

While adding a plug-in to a configuration template, the plug-ins on which the newly added plug-in depends can be added automatically to the template by clicking Resolve Dependency on the Configuration Template Definition page. See Executing a Template. Resolving dependency of a plug-in dynamically becomes possible by defining the plug-in dependency meta data in advance using the features described in this topic. Resolving plug-in dependency dynamically, involves:

- Adding the plug-in to a plug-in group.
- Managing plug-ins within a plug-in group.

- Managing dependencies between plug-in groups.

Adding Configuration Plug-ins to Plug-in Groups

When adding a new plug-in to a template, you can add it to a plug-in group (either new or existing group), which makes the plug-in a member of the repository. When the plug-in is available in the repository, the plug-in can:

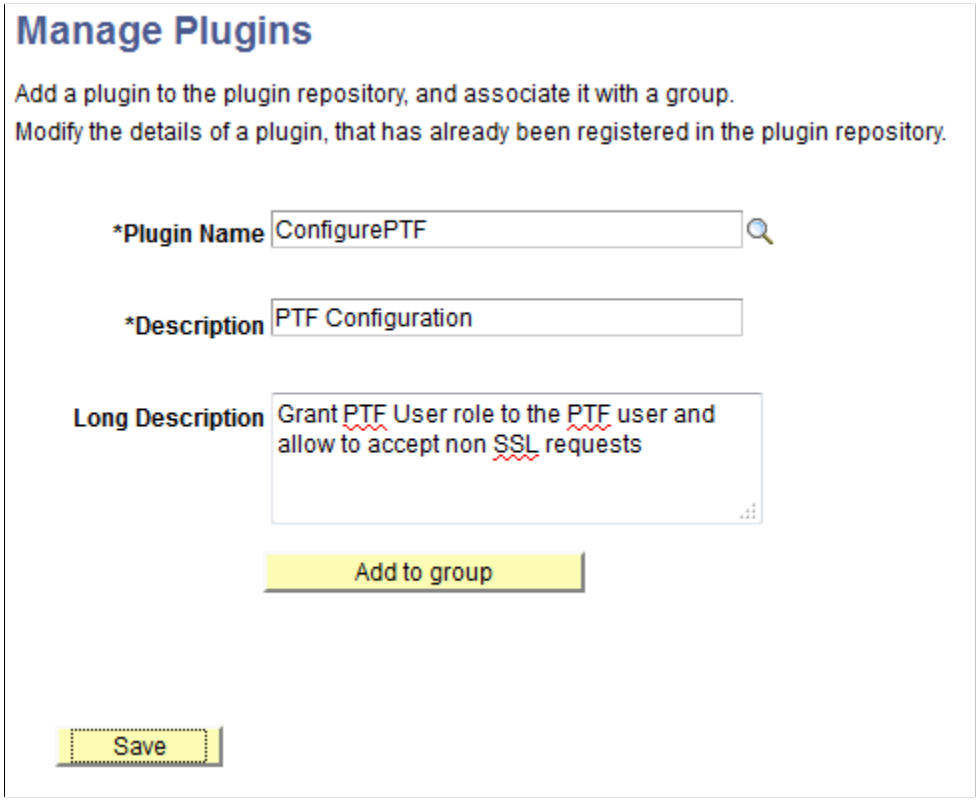
- Be executed through PIA.
- Have its dependencies resolved dynamically.

A plug-in group is a functional grouping of plug-ins. For example, Integration Broker plug-ins related to gateway configuration belong to the plug-in group *IB Gateway Configuration.*

Access the Manage Plug-in page to add a plug-in to a plug-in group.

PeopleTools > **Automated Config Manager** > **ACM Utilities** > **Manage ACM Plugins**

This example illustrates the fields and controls on the Manage Plug-ins page.



Select the plug-in by clicking search from the **Plug-in Name.** field. Click **Add to group** to add the plug-in to a plug-in group.

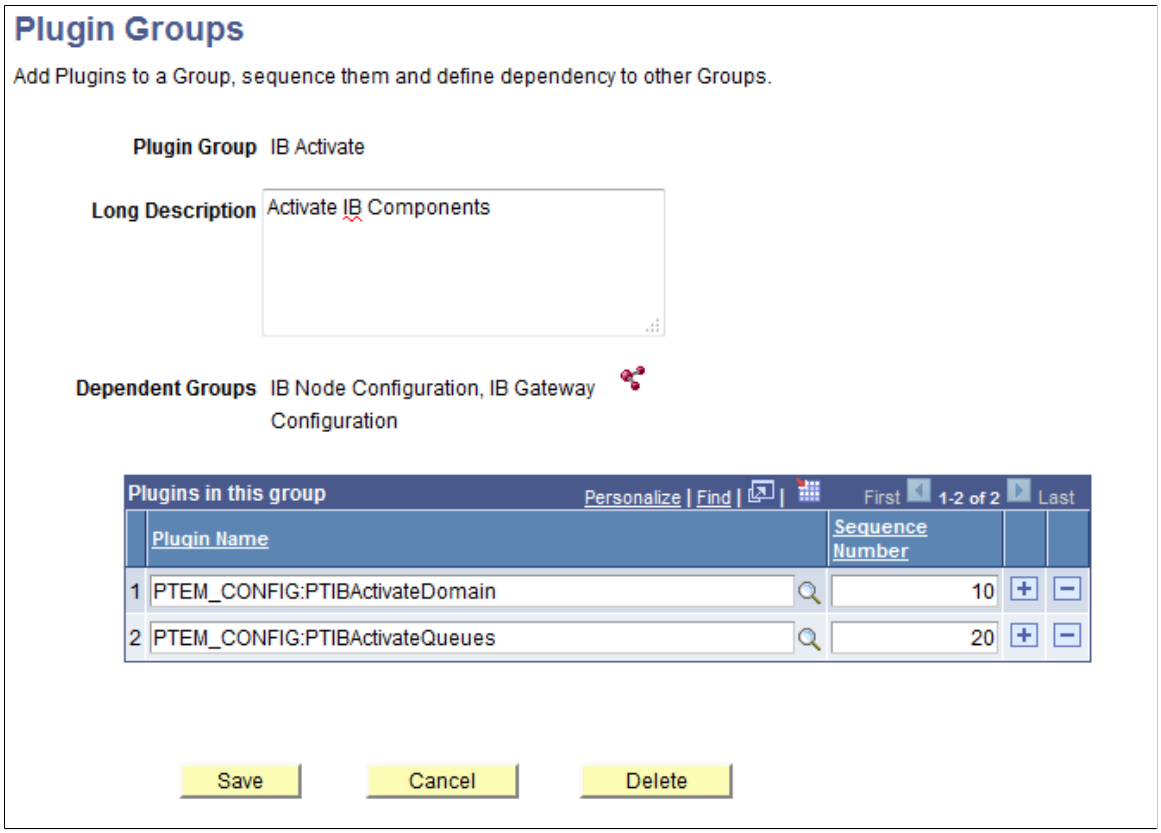
Managing Plug-ins within a Plug-in Group

Plug-ins can be added to an existing group or a new group according to the functionality of the new plug-in.

Access the Plug-in Groups page to manage plug-in groups.

PeopleTools > Automated Config Manager > ACM Utilities > ACM Plugin Groups

This example illustrates the fields and controls of the Plugin Groups page.



Term	*Definition*
Dependent Groups	Displays the dependent groups. Click the Add dependent groups icon to modify the list of dependent groups. See Managing Configuration Plug-in Dependency.
Plug-in Name	Select the plug-in names to add to the plug-in group list.
Sequence Number	Specifies the order of execution of the plug-ins within the group. The order of adding plug-ins dynamically to the template while resolving dependency is based on this sequence number.

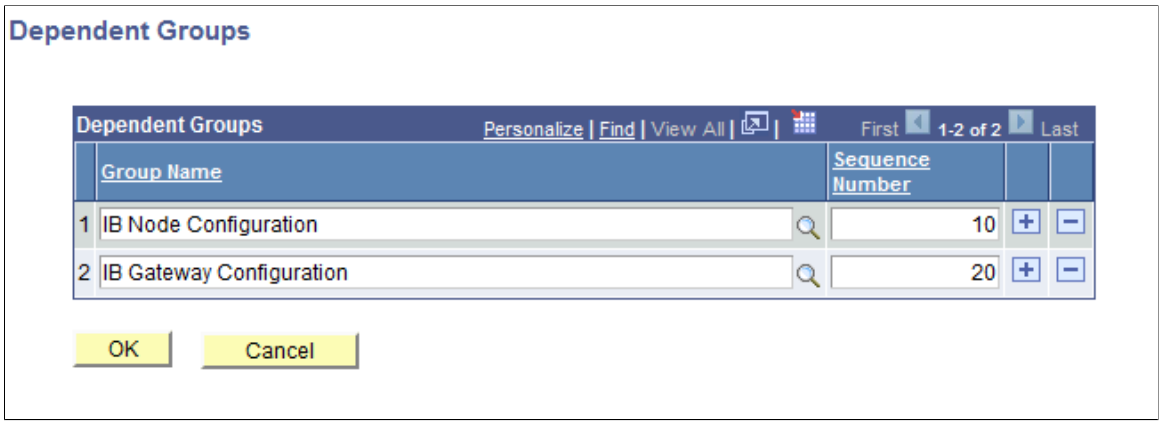
Managing Dependencies Between Plug-in Groups

The dependencies between plug-in groups is also taken into account when resolving dependency dynamically.

Access the Dependent Groups page to specify groups on which the current group is dependent.

PeopleTools > Automated Config Manager > ACM Utilities > ACM Plugin Groups > click the Add dependent groups icon

This example illustrates the fields and controls of the Dependent Groups page.

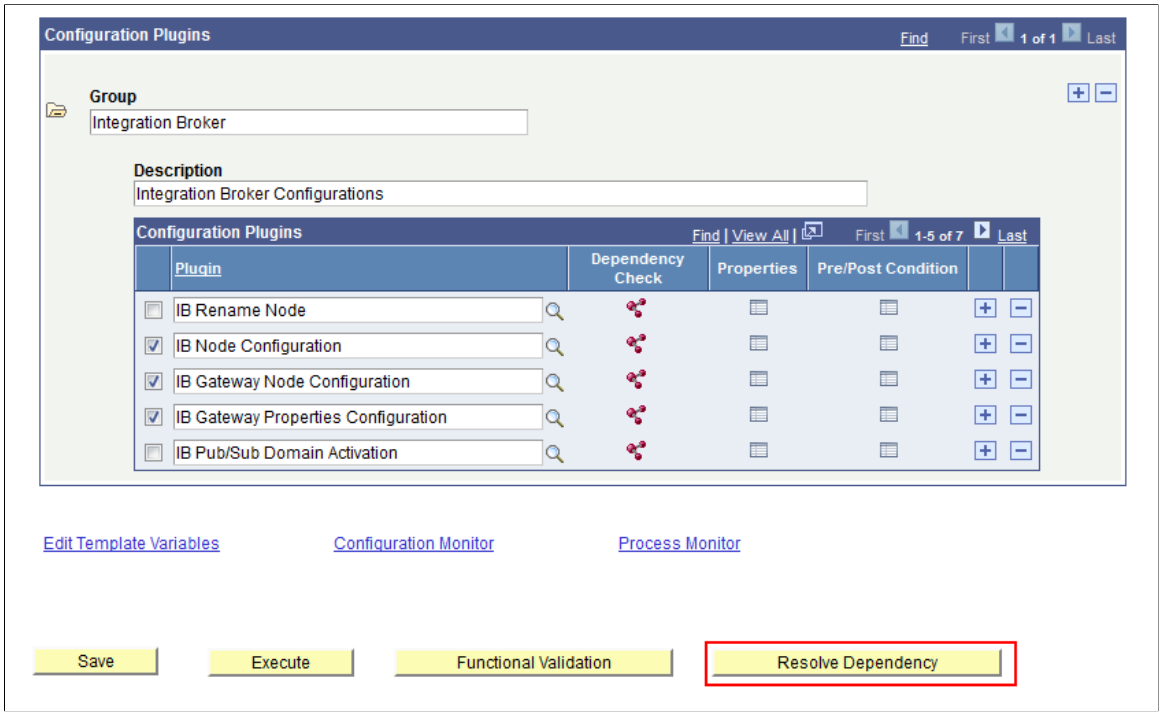


Term	*Definition*
Group Name	Select the group(s) on which the current group is dependent.
Sequence Number	The order in which the groups need to be configured.

Initiating a Resolve Dependency Request

You resolve plug-in dependency dynamically by click Resolve Dependency on the Configuration Template Definition page.

This example illustrates initiating a dynamic resolution of dependency by clicking Resolve Dependency.



With all the appropriate meta data set up as described in the previous sections, the system resolves dependency as described in this table and in the order presented.

Sequence	***Description***
1	The group dependency (based on the order of sequence number) is checked and all plug-ins belonging to groups that are dependant on the current plug-in group are added to the template. For example, assume these groups: Group 1: Plug-in 11, Plug-in 12 Group 2: Plug-in 21, Plug-in 22 Group 3: Plug-in 31, Plug-in 32, Plug-in 33 Group 3 is dependant on Group 1 and Group 2. If we add Plug-in 32 to a template, then all plug-ins in Group 1 and the plug-ins in Group 2 are added to the template automatically.

Sequence	*Description*
2	The plug-ins within the same group are added automatically. For example, assume these groups: Group 1: Plug-in 11, Plug-in 12 Group 2: Plug-in 21, Plug-in 22 Group 3: Plug-in 31, Plug-in 32, Plug-in 33. Group 3 is dependant on Group 1 and Group 2. If we add Plug-in 32 to a template, then all plug-ins in the same group will also be added (Plug-in 31 before Plug-in 32 and Plug-in 33 after Plug-in 32).
3	If the dynamically added plug-in is already part of the template, it is identified and will not be added.
4	All dependant plug-ins added will be added to the template in the Inactive or Unselected state. The existing plug-ins within the template are contained as such and its values are preserved while adding the dependant plug-ins.

Chapter 4

Working with Automated Configuration Using the Command Line

Understanding Automated Configuration Using the Command Line

PeopleTools provides an automated configuration management framework that enables you to store your environment configuration settings in a persistent state in the form of a template file. After setting up the basic infrastructure of a PeopleSoft environment, including database, application server, Process Scheduler server, and PIA domain, you run an Application Engine program, PTEM_CONFIG. This Application Engine program reads your environment properties from the stored template file, and inserts the stored values into the database, saving you from updating the settings manually each and every time you create or refresh an environment.

The settings stored in the template file are those that you typically enter on a configuration page in PeopleTools and save to the database. For example, the value for your Integration Gateway URL, which you would normally add manually on the Gateways page, can be entered in your template file once, and then retrieved by the automated configuration Application Engine program and inserted into the database each time you refresh that environment.

The PTEM_CONFIG Application Engine program, fires configuration plug-ins that take the properties specified for a particular plug-in, as specified in the template file, and configures the features associated with that plug-in. You enable configuration plug-ins by referencing them in your properties file and specifying the settings for that configuration plug-in.

The automated configuration management framework enables you to create the template file. The file can be created either manually or through PIA.

Understanding the Template File

You store your environment settings in a template file that you create either manually or through PIA. You save this file to a location on your system, which is identified by the PS_FILEDIR environment variable. You enable the desired configuration plug-ins by including a reference to that plug-in in the template file, followed by the expected configuration properties for that plug-in. When you run the automated configuration Application Engine program, the program engages the specified configuration plug-ins and inserts the configuration values you've included into the database.

To manually create a template file, see Creating a Template File Manually.

To create a template file through PIA, see Creating a Template File in PIA.

Creating a Template File Manually

This section discusses:

- Working with the structure and general settings of a template file.
- Specifying general settings.
- Specifying template settings.
- Enabling configuration plug-ins.

Working with the Structure and General Settings of a Template File

This section discusses:

- Naming the template file.
- Setting the structure of the template file.

Naming the Template File

Using the text editor for your operating system, create a new file and save it using the following convention:

<file_name>.<file_extension>

The file extension can be .txt, .properties, or any extension that is readable in ASCII format.

Note: You need to specify the template name using the PTEM_PROP environment variable prior to running the PTEM_CONFIG Application Engine program.

Setting the Structure of the Template File

The content of the properties should be structured in the following order:

1. General settings.
2. Template settings.
3. Individual configuration plug-in settings.

To enable a plug-in, you include a reference to that plug-in, such as:

plugin.IB.1=PTEM_CONFIG:PTIBRenameNode

where

- IB - refers to the category of the plug-in.
- 1- represents the sequence number in which plug-ins are configured.

- PTEM_CONFIG:PTIBRenameNode - represents the plug-in name, which consists of application package name (PTEM_CONFIG) and class name (PTIBRenameNode).

Then, you include the properties for that plug-in immediately following the plug-in reference. The following example illustrates a sample template file:

```
configure=true
verify=true
decrypt_password=true

template.domain=@webhist.SESSIONCOOKIEDOM
template.host=@webhist.WEBSERVERNAME
template.httpport=@webhist.PORTALHTTPPORT
template.jslport=@webhist.jolt_port
template.localnode=%LocalNode
template.piasite=@webhist.ps.discovery.siteName
template.sslport=@webhist.PORTALHTTPSPORT
template.tools_release=%ToolsRelease
template.userid=%UserId

#Group:Integration Broker
#Description:Integration Broker Configurations

##### Rename the default local node#####

plugin.IB.1=PTEM_CONFIG:PTIBRenameNode

#Default local node
env.default_local_node=@localnode@

#Purge application server messages
env.app_msg_purge_all_dms=true
```

Note: All other configuration plug-ins you want to enable can be included in the file as needed.

Specifying General Settings

The general settings appear at the top of the template file. These settings apply to the overall configuration program run.

Property	***Description***
configure	Enables you to set the entire template file as active by setting it to true. If set to false, the PTEM_CONFIG Application Engine program ignores the plug-ins and settings stored within the template file.
verify	If set to true, the automated configuration management framework verifies the settings. The parameters provided in the template file are validated against the corresponding values inserted into the database to ensure they are identical.
encrypt_password	If set to true, any sensitive data stored in the template file will be encrypted, and it will generate a template file where the sensitive data is encrypted.

Property	*Description*
decrypt_password	Set to true if the template file contains any encrypted sensitive data.
generate_property_file	Set to true if the template file consists only plug-in names and the properties for each plug-in is to be generated. If set to True, the property generates a new file (new_config_file.txt) with the plug-in properties in the same path where the template file is stored.

Specifying Template Settings

Template settings are a group of settings that apply to almost all configuration plug-ins and can be shared across multiple property settings. For example, HTTP port is a template setting, and rather than specifying it for each and every configuration value where it is needed (PIA 'URL, URI URL, Integration Gateway URL, and so on), you can specify it once in the template settings, and it will be replicated as needed throughout the template file, where referenced. This avoids unnecessary repetition and errors.

You can use %SystemVariable to default the value to a system variable; and use

%ENV_VARIABLE_NAME% to set.

You can also retrieve stored values in the web profile web history, by using

@webhist.<property_name>.

The following example illustrates the template settings section of a sample template file:

```
template.domain=@webhist.SESSIONCOOKIEDOM
template.host=@webhist.WEBSERVERNAME
template.httpport=@webhist.PORTALHTTPPORT
template.jslport=@webhist.jolt_port
template.localnode=%LocalNode
template.piasite=@webhist.ps.discovery.siteName
template.sslport=@webhist.PORTALHTTPSPORT
template.tools_release=%ToolsRelease
template.userid=%UserId
```

Property	*Description*
template.domain	Name of the PIA domain, such as 'peoplesoft'.
template.host	Server host for the web server installation.
template.httpport	The HTTP port on which the PIA domain listens.
template.jslport	Jolt port.
template.localnode	The current node used by the system. %Dbname or %LocalNode can be used if applicable.

Property	*Description*
template.piasite	The name of the PeopleSoft site.
template.sslport	The HTTPS port on which the PIA domain listens.
template.tools_release	PeopleTools release.
template.userid	User ID for accessing and configuring the system.

Enabling Configuration Plug-ins

You can include the configuration plug-ins in this section within your template file.

The following example illustrates the PTIBRenameNode configuration plug-in. In this example, the plug-in is not enabled because the plug-in is commented (#). To enable the plug-in, delete the (#) hash sign preceding the plug-in name. For example, plugin.IB.1=PTEM_CONFIG:PTIBRenameNode.

```
##### Rename the default local node#####

#plugin.IB.1=PTEM_CONFIG:PTIBRenameNode

#Default local node
env.default_local_node=@localnode@

#Purge application server messages
env.app_msg_purge_all_dms=true
```

Note: All other configuration plug-ins you want to enable can be included in the file as needed.

In PIA, you can enable a plug-in in a template by selecting the check box corresponding to the plug-in on the Configuration Template Definition page.

See Creating a Template.

Creating a Template File in PIA

The automated configuration management framework enables you to create a template file by exporting (downloading) a template from the PIA interface to a network location or computer. The template file is saved with a .txt extension and can be used as an input file when configuring systems using the command line.

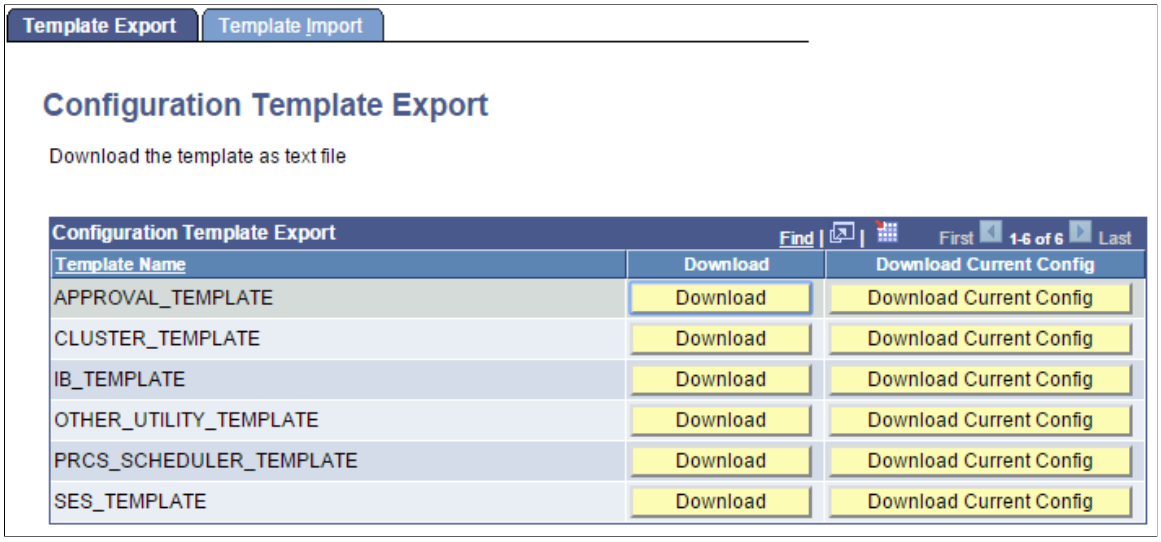
Exporting a Template

Use the Configuration Template Export page to create a text file of the settings and configuration plug-ins of the selected template.

Access the Configuration Template Export page using the following navigation path:

PeopleTools > Automated Config Manager > ACM Templates > Export/Import ACM Templates

This example illustrates the fields and controls on the Configuration Template Export page. You can find definitions for the fields and controls later on this page.



Field or Control	*Description*
Download	Click to create and save the template file as a text file. By default, the downloaded file name will be TEMPLATE_NAME.TXT.

Field or Control	*Description*
Download Current Config	Click to create a ”snapshot” of the current configuration. This option enables you to export current configuration values set in the system to a template file. This option enables you to export current configuration values set in the system to a template file. A product area contains various plug-ins that have properties with set values. This set of plug-ins and property values constitute the configuration of that product, such as Integration Broker, in the environment. Selecting **Download Current Config** allows you to view the configuration parameters of a plugin as defined within a template. The snapshot template can be used to: • Clone environments. Environment template variables, such as env.hostname= @hostname@, will not be resolved. However, specific configuration settings, not dependent on a specific host for example, will appear in the exported template for reuse. This reduces the need to modify an exported template manually. • Back up environments. In the case of an environment goes down or becomes corrupt, the snapshot backup file can be used to restore the environment to its original state. By default, the downloaded file name will be TEMPLATE_ NAME_CurrentConfiguration.TXT.

Note: If the template contains any passwords, the passwords are encrypted.

Running the PSRUNACM Script

In addition to the Automated Configuration Manager that is executed from PIA, the automated configuration management framework allows you to configure the products for your environment from the command line using the PSRUNACM script.

The PSRUNACM.bat and the PSRUNACM.sh files are available in the Utility folder of PS_HOME.

This section discusses:

- Running PSRUNACM in Windows.
- Running PSRUNACM in UNIX.

Running PSRUNACM in Windows

To run PSRUNACM in Windows:

1. Set the PS_HOME and PS_CFG_HOME environment variables.
2. Set the PS_SERVER_CFG environment variable to the Process Scheduler configuration file directory.

   For example:

   set PS_SERVER_CFG=$PS_CFG_HOME/appserv/prcs/$DOMAIN/psprcs.cfg
3. Set the PS_FILEDIR environment variable, if required.

   The template file that is used as input to configure the environment can be present either in database or as a text file.

   - If the template file is present in database, do not set the PS_FILEDIR environment variable.
   - A template file (text file) can exist in a remote path or you can copy the template file (text file) into the same directory as the psrunACM script.

     Note: If the template file (text file) exists in a remote path, you must set the PS_FILEDIR environment variable to specify the remote path of the template file. If the template file exists in a remote path and if the PS_FILEDIR variable is not set, the directory from where the psrunACM script is run is considered as the location of the template file. Hence, you need to copy the template file to the same directory as the psrunACM script.
   - If you copy the template file into the same directory as the psrunACM script, do not set the PS_FILEDIR environment variable.
4. Run the PSRUNACM.bat file.

   - If the template file is present in database, run the following command by replacing the values within <> with appropriate values.

     ```
     psrunACM.bat <SERVER> <DBTYPE> <DBNAME> <USERID> <PASSWORD> <TEMPLATE_NAM⇒
     E> <OPTION>
     ```

     where

Field or Control	***Description***
SERVER	The host machine name in which the script is executed.
DBTYPE	The type of database on which the configuration is run, for example, ORACLE.
DBNAME	The name of the hosted database.
USERID	Operator ID.
PASSWORD	Operator password.

Field or Control	*Description*
TEMPLATE_NAME	The name of the template file, which is used as input for the configuration run.
OPTION	The option parameter is used to decide whether the template should be executed from database, imported to database, or exported from database. Valid values: • EXP - The template mentioned in TEMPLATE_NAME is exported from the database mentioned in DBNAME to the local machine. • IMP - The template mentioned in TEMPLATE_NAME is imported to the database mentioned in DBNAME from the current path from where the psrunACM.bat script is invoked. • EXEC - The template mentioned in TEMPLATE_NAME is used as input to configure the environment.

- If the template file is available as a text file, run the following command by replacing the values within <> with appropriate values.

  Before you run the command, ensure that the PS_FILEDIR environment variable is set if the template file exists in a remote path.

```
psrunACM.bat <SERVER> <DBTYPE> <DBNAME> <USERID> <PASSWORD> <TEMPLATE_NAM⇒
E>
```

Note: If the input is a text file, the OPTION parameter is not required.

Running PSRUNACM in UNIX

To run PSRUNACM in UNIX:

1. Set the PS_HOME and PS_CFG_HOME environment variables.
2. Set the PS_SERVER_CFG environment variable to the Process Scheduler configuration file directory.

   For example:

```
set PS_SERVER_CFG=$PS_CFG_HOME/appserv/prcs/$DOMAIN/psprcs.cfg
```

3. Set the PS_FILEDIR environment variable, if required.

   The template file that is used as input to configure the environment can be present either in database or as a text file.

   - If the template file is present in database, do not set the PS_FILEDIR environment variable.

- A template file (text file) can exist in a remote path or you can copy the template file (text file) into the same directory as the psrunACM script.

  Note: If the template file (text file) exists in a remote path, you must set the PS_FILEDIR environment variable to specify the remote path of the template file. If the template file exists in a remote path and if the PS_FILEDIR variable is not set, the directory from where the psrunACM script is run is considered as the location of the template file. Hence, you need to copy the template file to the same directory as the psrunACM script.

- If you copy the template file into the same directory as the psrunACM script, do not set the PS_FILEDIR environment variable.

4. Run the PSRUNACM.sh script.

   - If the template file is present in database, run the following command by replacing the values within <> with appropriate values. The parameters within [ ] are optional.

     ```
     psrunACM.sh <SERVER> <DBTYPE> <DBNAME> <USERID> <PASSWORD> <TEMPLATE_NAME⇒
     > [DOMAIN] <OPTION - EXEC/IMP/EXP> [LOG_LEVEL - ERROR]
     ```

     where

Field or Control	***Description***
SERVER	The host machine name in which the script is executed.
DBTYPE	The type of database on which the configuration is run, for example, ORACLE.
DBNAME	The name of the hosted database.
USERID	Operator ID.
PASSWORD	Operator password.
TEMPLATE_NAME	The name of the template file, which is used as input for the configuration run.
DOMAIN	The Process Scheduler server domain name is an optional parameter. **Note:** If the domain name is not provided, ensure that the PS_SERVER_CFG environment variable is set.

Field or Control	*Description*
OPTION	The option parameter is used to decide whether the template should be executed from database, imported to database, or exported from database. Valid values: • EXP - The template mentioned in TEMPLATE_NAME is exported from the database mentioned in DBNAME to the local machine. • IMP - The template mentioned in TEMPLATE_NAME is imported to the database mentioned in DBNAME from the current path from where the psrunACM.sh script is invoked. • EXEC - The template mentioned in TEMPLATE_NAME is used as input to configure the environment.
LOG_LEVEL	The log_level parameter is optional. This log_level parameter is used to limit the console output to show the details of only those plug-ins that failed. This parameter can take only a single value — Error.

- If the template file is available as a text file, run the following command by replacing the values within <> with appropriate values.

  Before you run the command, ensure that the PS_FILEDIR environment variable is set if the template file exists in a remote path.

```
psrunACM.sh <SERVER> <DBTYPE> <DBNAME> <USERID> <PASSWORD> <TEMPLATE_NAME⇒
> <DOMAIN>
```

Note: When the input is a text file, the OPTION parameter is not required.

Running the PTEM_CONFIG Application Engine Program

This section provides an overview and discusses:

- Ensuring sufficient security permissions.
- Setting the environment variables for the command line session.
- Running the PTEM_CONFIG Application Engine program.
- Verifying your configuration settings.

Understanding the PTEM_CONFIG Program Run

After you have created the template file and entered all the required property values for your environment for the configuration plug-ins you have referenced, you load the values into the database by running the PTEM_CONFIG Application Engine program. The PTEM_CONFIG Application Engine program, which you run from the command line, determines the location of your environment template file by the PS_FILEDIR environment variable setting.

The following instructions assume you are familiar with running Application Engine programs. For more information on running Application Engine programs from the command line, see “Using the Command Line to Invoke Application Engine Programs” (Application Engine).

Ensuring Sufficient Security Permissions

Make sure the user ID running the Application Engine program has the appropriate security access. To run the PTEM_CONFIG Application Engine program, the user you submit on the command line must be associated with the PeopleSoft Administrator or the ACM Administrator role in PeopleTools Security.

Likewise, the user needs to have any required security for any of the products being automatically configured. For example, if you are configuring the Search Framework, the user ID needs to be associated with the required roles for that product, such as Search Administrator, Search Server, and so on.

See Security Permissions Required for Automated Configuration Management.

Setting the Environment Variables for the Command Line Session

Open the command line utility for your operating system and set the following environment variables.

- Set the PS_HOME environment variable to reflect the path of PS_HOME directory.

  For example:

  D:\>set PS_HOME=D:\PT8.55

- Set the PS_FILEDIR environment variable to reflect the location of the template file for the current command line session.

  For example in Windows:

  D:\>set PS_FILEDIR=D:\QEDMO_AUTOCONFIG

  In this case, the template file is located in D:\qedmo_autoconfig.

  For example in Linux or AIX:

  export PS_FILEDIR=/tmp

- Set the PS_WRITE_ACM_LOG environment variable to specify whether the plugin execution should be logged and whether the status of each plugin should be generated in JSON format while running plugins through command line.

  Set the PS_WRITE_ACM_LOG to true to generate logs and JSON output.

  For example in Windows:

set PS_WRITE_ACM_LOG=true

For example in Linux or AIX:

export PS_WRITE_ACM_LOG=true

If PS_WRITE_ACM_LOG is set to true, the ACM log file is stored in the PS_FILEDIR location with the name ptem_config.log. Also, the JSON file containing status of each executed plugin is stored in the PS_FILEDIR location with the name <propfilename>_out.json, where <propfilename> is the name of the template file. For example, IB_config.properties_out.json. The property file name is stored in the PTEM_PROP environment variable.

- Specify the name of the template file for Application Engine program input using the PTEM_PROP environment variable.

  For example:

  D:\>set PTEM_PROP=IB_TEMPLATE.txt

- Specify whether to execute, import, or export a template in database using the PTACM_OPTION environment variable.

  Valid values:

  - Execute - This option will override the template file given in PTEM_PROP and take the template from database (template name will be provided in the PTACM_TEMPLATE environment variable).
  - Import - this option will import a template file given in PTEM_PROP to template provided in PTACM_TEMPLATE.
  - Export - this option will export the template in PTACM_TEMPLATE to template file given in PTEM_PROP.

  For example:

  D:\>set PTACM_OPTION=execute

- If the PTACM_OPTION is set, specify the template name using the PTACM_TEMPLATE environment variable.

Running the PTEM_CONFIG Application Engine Program

From PeopleTools 8.59, the DPK deployment process executes the utility script, psrunACM.bat or psrunACM.sh, in order to control the execution of ACM plug-ins in cases where it may encounter failures thereby ensuring that the DPK deployment process continues smoothly or is aborted when certain conditions are not fulfilled. The PS_WRITE_ACM_LOG environment variable enables logging of the ACM plug-in execution and generates a JSON file containing the status of each executed plug-in, thus giving you greater control on running the DPK deployment process. The utility script provides the following return codes:

Return Code	*Description*
0	Successful completion.
10	The template under consideration does not exist in the database.
11	Web history is not found. Login to PIA, and re-run the program.
12	User does not have permission to execute any of the plug-ins.
13	Exiting configuration as per post-configuration condition.
14	Failure when running any plug-in in the template.
15	Warning when running any plug-in in the template.

In the same command line session where you set PS_FILEDIR to point to your environment variables template file, launch the PTEM_CONFIG Application Engine program.

Change directories to PS_HOME\bin\client\winx86 and submit the following command line parameters to the Application Engine executable (psae.exe) replacing the values in <> with appropriate values.

For example:

```
psae.exe -CT <DB type> -CS <server name> -CD <DB name>  -CO <user> -CP <password> -⇒
R <runid> -AI PTEM_CONFIG  -I <instance number>
```

Parameter	*Description*
-CT	Database type (connection type).
-CS	Server name (used only for specific database types).
-CD	Database name.
-CO	User ID connecting to the database.
-CP	Password associated with the user ID.
-R	Run control ID.
-AI	Application Engine program name (Application ID)
-I	Instance number.

See the Application Engine documentation for more information on other optional command line parameters.

After the PTEM_CONFIG program processes all the configuration properties for the configuration plug-ins contained in the environment variables template file, you should see a message indicating the successful program run.

Verifying Your Configuration Settings

Depending on the configuration plug-ins you included, test the appropriate settings in your environment to confirm that the values you entered in the environment variables template file are correct.

For example, if you included Integration Broker configuration plug-ins, make sure the elements related to Integration Broker are configured correctly.

Applying Plug-in Property Updates Selectively

When configuring environments with Automated Configuration Manager, you can apply dozens of configuration properties per template, depending on the amount of plug-ins the template invokes. In some scenarios, you may find that you need to update a small subset of the plug-in properties due to preferences or changes in the overall environment. Rather than running an entire template script again and updating configuration settings that are fine as they are, you can choose to apply only that subset of properties that need to be updated for a particular plug-in.

Creating a Selective Template

To apply plug-in properties selectively:

1. Create a separate template.txt file.
2. Add to the template file the plug-in section that controls the values of the property values you need updated.
3. Identify the required properties for the plug-in and make sure they remain in the template file.

   Note: Required properties appear with an asterisk (*) next to them in template definition in the PIA interface. If you are missing a required property, an error will be shown on the command line, listing the missing properties.

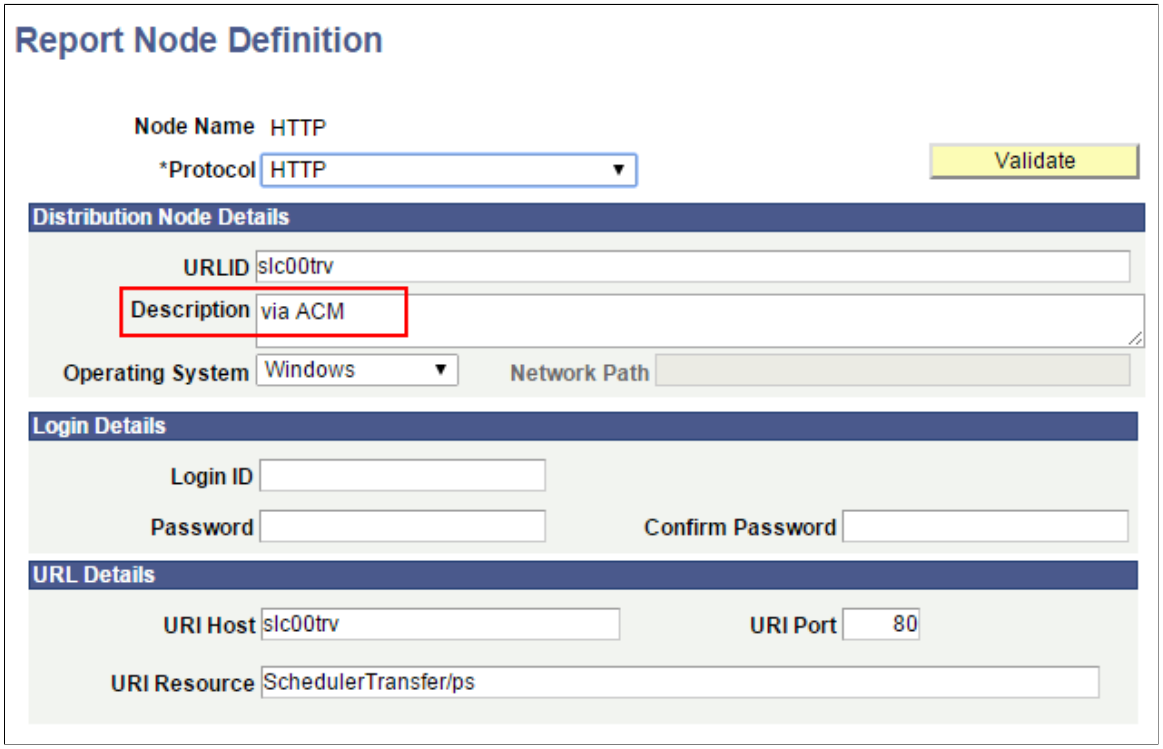
4. Identify the property (or properties) that you need to update, and make sure to provide the updated values.
5. Remove any unnecessary property settings, and save the file.
6. Apply the updated configuration settings from the command line.

Example: Applying Configuration Property Updates Selectively

This section provides a simple example to illustrate the concept of applying configuration properties selectively.

Assume that you have previously configured Process Scheduler using the PRCS_SCHEDULER_TEMPLATE.txt file. However you realize that you need to the change the Description field of the report node definition.

This example illustrates a field value that can be updated using a configuration plug-in. In this case, the value needs to be updated, yet the other values on the page do not require updating.



After locating the plug-in and the property associated with the field that needs updating, you may find that within the current template file currently contains dozens of properties and multiple plug-ins. However, you only want to update a single field in your test environments.

This example illustrates how you need to identify the property that controls the automatic updates of a field and make sure you know which plug-in controls the property.

```
########################################
#Configure Process Scheduler Report Node configuration
########################################

plugin.Process Scheduler.1=PTEM_CONFIG:PTProcessSchedulerReportNode

#Distribution Node Name
env.distnodename=HTTP
#Protocol; Possible Values: 0 => HTTP:; Possible Values: 1 => HTTPS:; Possible Valu
env.cdm_proto=0
#SSL Mode; Possible Values: 0 => EXPLICIT:; Possible Values: 1 => IMPLICIT:
env.cdm_ssl_mode=0
#Operating System; Possible Values: 0 => DOS:; Possible Values: 1 => NT/Win95 Client
env.opsys=2
#File Transfer ID
env.ftpid=
#File Transfer Protocol Address
env.ftpaddress=
#Directory for FTP
env.ftpdirectory=
#URLID
env.url=@host@
#Windows Network Path
env.winnetworkpath=
#Description
env.descrlong=via ACM
#URI Host
env.uri_host=@host@
#URI Port
env.uri_port=@httpport@
#URI Resource
env.uri_resource=SchedulerTransfer/ps
```

Rather than invoking multiple plug-ins unnecessarily and updating dozens of configuration properties, you can create a separate template file containing only the required plug-ins and properties for your selective update.

In this case, only the PTProcessSchedulerReportNode plug-in needs to be invoked.

The required properties are:

- Distribution Node Name
- Environment Protocol
- Operating System

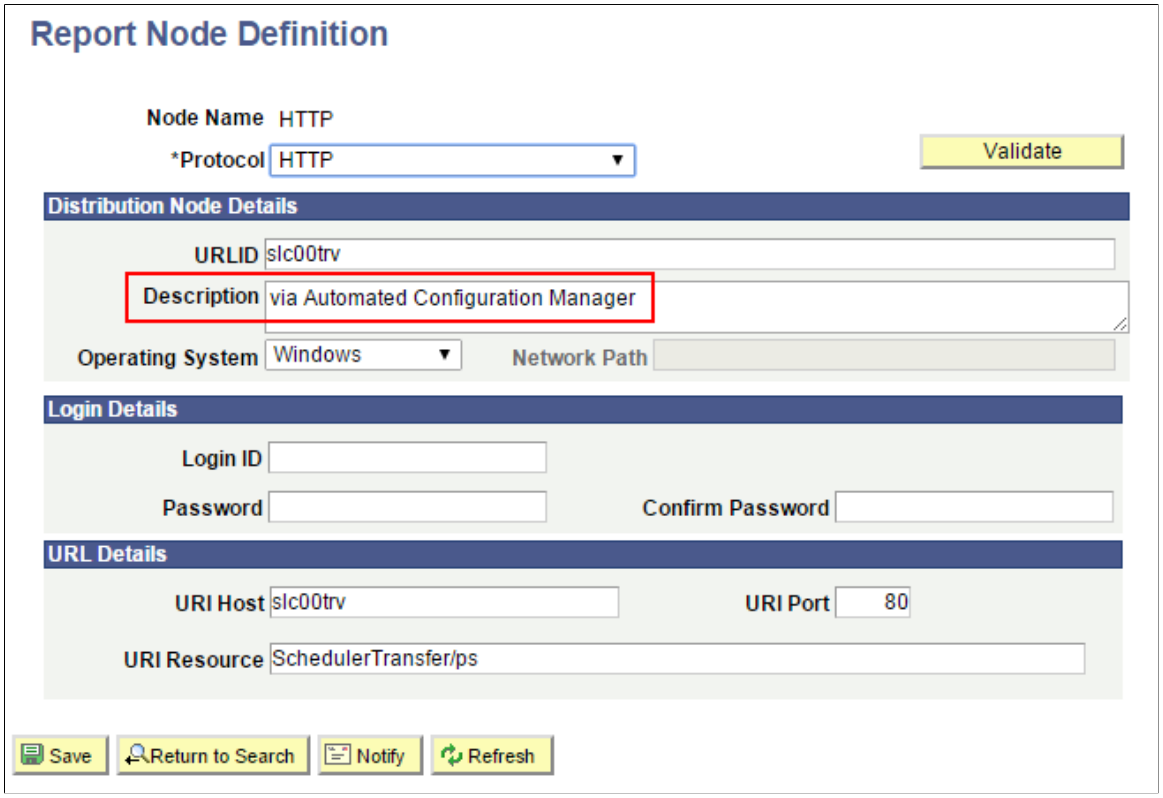
You need to include all mandatory properties in addition to the selected properties you need updated.

This example illustrates that with selective updates, you only need to include required entries in the selective template file.

```
configure=true
verify=true

################################################
#Configure Process Scheduler Report Node configuration
################################################

plugin.Process Scheduler.1=PTEM_CONFIG:PTProcessSchedulerReportNode

#Distribution Node Name
env.distnodename=HTTP

#Environment Protocol
env.cdm_proto=0

#Operating System
env.opsys=2

#Description
env.descrlong=via Automated Configuration Manager
```

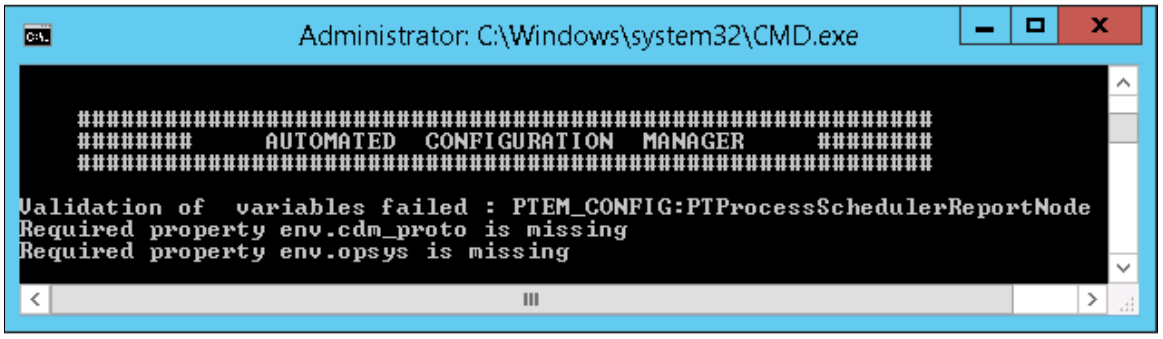
After running the template file, you update the configuration with only the new values you supplied with your selected property value(s).

This example illustrates the selected field updated by the property included in the selective template file.



If you have failed to include any properties required by the include plug-in(s), the system alerts you to which properties are missing.

This example illustrates the error displayed during the PSRUNACM script run if a required property is missing.



Handling Execution Errors and Status on the Command Line

This topic describes how to interpret the output of a configuration template execution and how you can use the status display and pre and post condition options to aid in creating your environments.

Understanding the Console Output

The console output displays a standard set of data to communicate states of each configuration plug-in invoked during a template run, such as status, description, severity and so on. There is also an overall indication of status at the end of the output, which applies to the template, not just the individual plug-ins. You can use scripts to interpret the console output and perform relative actions accordingly.

The following is an example of the command line console output:

```
############################################################
  ######     AUTOMATED  CONFIGURATION  MANAGER     ######
############################################################

PTEM_CONFIG:PTIBRenameNode: Rename the default local node
Configuring plug in : PTEM_CONFIG:PTIBRenameNode
STATUS: SUCCESS
DESCRIPTION:
SEVERITY:
Configuration completed : PTEM_CONFIG:PTIBRenameNode

PTEM_CONFIG:PTIBConfigureDBNode: Configure the DB node
Configuring plug in : PTEM_CONFIG:PTIBConfigureDBNode
STATUS: SUCCESS
DESCRIPTION:
SEVERITY:
Configuration completed : PTEM_CONFIG:PTIBConfigureDBNode

PTEM_CONFIG:PTIBConfigureGatewayNodes: Configure gateway URL, load connectors, defi⇒

ne node in the gateway
Configuring plug in : PTEM_CONFIG:PTIBConfigureGatewayNodes
STATUS: WARNING
DESCRIPTION:
Properties env.ib_appserver_host, env.ib_jolt_port are not configured correctly
Properties env.ib_appserver_host, env.ib_jolt_port are not configured correctly
SEVERITY: MAJOR
Configuration completed : PTEM_CONFIG:PTIBConfigureGatewayNodes

PTEM_CONFIG:PTIBConfigureGatewayProperties: Set keystore password, proxy host, p
roxy port, non proxy host in gateway properties file
Configuring plug in : PTEM_CONFIG:PTIBConfigureGatewayProperties
STATUS: ERROR
DESCRIPTION: Class PTIBConfigureGatewayProperties not found
SEVERITY: CRITICAL
Configuration failed : PTEM_CONFIG:PTIBConfigureGatewayProperties

****Successfully finished configuring the environment****
Configured environment in 9 seconds

OVERALL STATUS: ERROR
Application Engine program PTEM_CONFIG ended normally
```

Interpreting the Console Output

Status Value	*Description*
STATUS	• SUCCESS • WARNING • ERROR

Status Value	*Description*
DESCRIPTION	Various messages can be displayed, depending on the nature of the issue or error. In the case of warnings or errors, make sure you check all the details provided.
SEVERITY	• WARNING: Verification of configuration failures. • CRITICAL: Configuration failure. • MEDIUM: Pre/Post condition failure. • SEVERE: Exceptions.

PSRUNACM Error Codes

To access error codes:

- UNIX: `$ echo $?`
- Windows: `C:\>echo %ERRORLEVEL%`

Error Code	*Description*
0	Successful completion.
10	The specified template does not exist in the database.
11	Web history not found. Login to PIA, and re-run the program.
12	User ID specified does not have permission to execute any of the plug-ins.
13	Exiting configuration execution due to post-configuration condition.
14	Failure in running any plug-in in the template.
15	Warning in running any plug-in in the template.

Working with Pre and Post-Conditions with Template Files and the Console

This section provides some sample pre and post-conditions set in the template file and illustrates sample output for such conditions.

The feature related to setting pre and post-conditions with your configuration template executions is described in another topic in this product documentation. See Handling Errors During Plug-in Execution.

Sample Precondition

Property file:

```
plugin.prcs.1=PTEM_CONFIG:PTProcessSchedulerReportNode
PreConditionOnError=PTIBConfigureGatewayNodes,
PTIBConfigureGatewayProperties
env.wrkoperpswd=beaweb
env.url=http://example.com:8000/psreports/ps
```

Console output:

```
PTEM_CONFIG:PTIBConfigureGatewayNodes: Configure gateway URL, load connectors,
define node in the gateway
Configuring plug in : PTEM_CONFIG:PTIBConfigureGatewayNodes
STATUS: ERROR
DESCRIPTION:
Invalid Gateway User/Password. (158,171) PTEM_CONFIG.PTIBConfigureGatewayNodes.OnEx⇒

ecute
Name:getIntegrationGatewayProperties  PCPC:21728  Statement:254
Called from:PTEM_CONFIG.PTIBConfigureGatewayNodes.OnExecute
Name:configureNodeStatement:160
Called from:PTEM_CONFIG.PTIBConfigureGatewayNodes.OnExecute
Name:configureEnvironment  Statement:73
Called from:PTEM_CONFIG.MAIN.GBL.default.1900-01-01.Step01.OnExecute  Statement:904
SEVERITY: SEVERE
Configuration failed : PTEM_CONFIG:PTIBConfigureGatewayNodes

PTEM_CONFIG:PTIBConfigureGatewayProperties: Set keystore password, proxy host,
proxy port, non proxy host in gateway properties file

Configuring plug in : PTEM_CONFIG:PTIBConfigureGatewayProperties
STATUS: ERROR
DESCRIPTION:
Invalid Gateway User/Password. (158,171) PTEM_CONFIG.PTIBConfigureGatewayNodes.OnEx⇒

ecute
Name:getIntegrationGatewayProperties  PCPC:21728  Statement:254
Called from:PTEM_CONFIG.PTIBConfigureGatewayProperties.OnExecute
Name:configureEnvironment  Statement:58
Called from:PTEM_CONFIG.MAIN.GBL.default.1900-01-01.Step01.OnExecute  Statement:904
SEVERITY: SEVERE
Configuration failed : PTEM_CONFIG:PTIBConfigureGatewayProperties

PTEM_CONFIG:PTProcessSchedulerReportNode: Configure Process Scheduler Report Node
configuration
Configuring plug in : PTEM_CONFIG:PTProcessSchedulerReportNode
STATUS: ERROR
DESCRIPTION:
Skipping Configuration for: PTEM_CONFIG:PTProcessSchedulerReportNode
since Pre-Condition Check failed
SEVERITY: MEDIUM
Configuration failed : PTEM_CONFIG:PTProcessSchedulerReportNode

****Completed environment configuration****
Configured environment in 37 seconds
Application Engine program PTEM_CONFIG ended normally
```

Sample Post-Condition

Property file:

```
plugin.ib.2=PTEM_CONFIG:PTIBConfigureGatewayNodes
PostCondition=EXIT
env.gateway_user=administrator
env.gateway_password=password
env.ib_appserver_host=SLC00FEQ
```

```
env.ib_jolt_port=9032
env.ib_node_proxy_userid=PTDMO
env.ib_node_proxy_password=password
env.ib_appserver_domain_password=password
```

Console output:

```
Configuring plug in : PTEM_CONFIG:PTIBConfigureGatewayNodes
STATUS: ERROR
DESCRIPTION:
Invalid Gateway User/Password. (158,171) PTEM_CONFIG.PTIBConfigureGatewayNodes.OnEx⇒

ecute
Name:getIntegrationGatewayProperties  PCPC:21728  Statement:254
Called from:PTEM_CONFIG.PTIBConfigureGatewayNodes.OnExecute
Name:configureNodeStatement:160
Called from:PTEM_CONFIG.PTIBConfigureGatewayNodes.OnExecute
Name:configureEnvironment  Statement:73
Called from:PTEM_CONFIG.MAIN.GBL.default.1900-01-01.Step01.OnExecute  Statement:904
SEVERITY: SEVERE
Configuration failed : PTEM_CONFIG:PTIBConfigureGatewayNodes
EXITING CONFIGURATION AS PER POST-CONFIGURATION CONDITION:
PTEM_CONFIG:PTIBConfigureGatewayNodes
Configured environment in 34 seconds
Application Engine program PTEM_CONFIG ended normally
```

Running the Console in Quiet Mode

When creating scripts to run configuration templates, you may want to reduce the output to only the essential information required for building your error handling. You can pass the QUIET parameter when calling Application Engine to: This feature helps which helps in scripting based on the console output.

- Hide the console headers and footers.
- Displaying only essential details of a configuration template execution.

For example, using the QUIET parameter:

```
%PS_HOME%\bin\client\winx86\psae.exe -CT %DBTYPE% -CS %SERVER% -CD %DBNAME% -CO %US⇒

ER% -CP
%PASS%  -R RUN023 -AI PTEM_CONFIG -I 2 -QUIET Y
```

Produces output similar to:

```
############################################################
########     AUTOMATED  CONFIGURATION  MANAGER     ########
############################################################
PTEM_CONFIG:PTIBRenameNode: Rename the default local node
Configuring plug in : PTEM_CONFIG:PTIBRenameNode
STATUS: SUCCESS
DESCRIPTION: NONE
SEVERITY: NONE
Configuration completed : PTEM_CONFIG:PTIBRenameNode
PTEM_CONFIG:PTIBConfigureDBNode: Configure the DB node
Configuring plug in : PTEM_CONFIG:PTIBConfigureDBNode
STATUS: SUCCESS
DESCRIPTION: NONE
SEVERITY: NONE
Configuration completed : PTEM_CONFIG:PTIBConfigureDBNode
****Completed environment configuration****
Configured environment in 7 seconds
```


Chapter 5

Working with Sensitive Data

Understanding Password Encryption in Template File

Sensitive data, like passwords, can be encrypted so that these properties are not exposed if you share the template file among multiple environments and multiple development or testing teams.

The following properties are currently identified as being sensitive data:

- userpass
- env.nodex_search_admin_password
- env.default_local_node_pass
- env.gateway_password
- env.ib_appserver_domain_password
- env.gateway_keystore_password
- env.nodex_search_read_password
- env.search_call_back_password

To encrypt sensitive data, run the PTEM_CONFIG Application Engine program with the property encrypt_password set to True in the template file. This will generate a new template file named <*original template file name*>*_encrypted* in the location of the original template file.

The encrypted template file can be used as input to run the PTEM_CONFIG Application Engine program by providing the property decrypt_password=true in the template file, which internally decrypts the sensitive properties for configuring the environment.

Chapter 6

Migrating Configuration Data

Migrating Configuration Data Using ADS

PeopleSoft provides ADS definitions to migrate configuration data from one environment to another as a project file.

For information on using ADS, see “Understanding ADS Projects” (Lifecycle Management Guide).

Tables for Configuration Data Migration

The following tables store configuration data that can be used for migrating data.

ADS Definition	*Description*
PTACM_PLUGIN	Table for configuration plug-ins. This table contains the plug-in application class name, its description and category.
PTACM_TEMPLATE	Table for configuration templates. Templates contain plug-ins and their properties for configuration.
PTACM_TEMP_VAR	Table for configuration template variables. This table contains the template variable, which can be used to assign the values to the plug-in properties.

ADS Definition for Template

Use the PTACM_TEMPLATE ADS definition to export templates and related plug-ins.

The PTACM_TEMPLATE ADS definition uses the following tables.

Table	*Description*
PTACM_TEMPLATE	Stores template data.
PTACM_TEMP_VAR	Stores template variable data.
PTACM_ADS_RELN	Stores data pertaining to relationship between templates and plug-ins.

ADS Definition for Plug-in

Use the PTACM_PLUGIN ADS definition to export only plug-ins.

Exporting Configuration Data

In addition to the ADS definitions, you can also export configuration data using the export functionality available within Automated Configuration Manager. The exported template file can be used as input for command line when you run a product configuration on another system.

See Creating a Template File in PIA.

Chapter 7

Sample Template File

Sample Template File

The following example shows a sample template file:

```
template_name=IB_TEMPLATE

configure=true
verify=true
decrypt_password=true

template.domain=@webhist.SESSIONCOOKIEDOM
template.host=@webhist.WEBSERVERNAME
template.httpport=@webhist.PORTALHTTPPORT
template.jslport=@webhist.jolt_port
template.localnode=%LocalNode
template.piasite=@webhist.ps.discovery.siteName
template.sslport=@webhist.PORTALHTTPSPORT
template.tools_release=%ToolsRelease
template.userid=%UserId

#Group:Integration Broker
#Description:Integration Broker Configurations

##### Rename the default local node#####

#plugin.IB.1=PTEM_CONFIG:PTIBRenameNode

#Default local node
env.default_local_node=@localnode@

#Purge application server messages
env.app_msg_purge_all_dms=true

##### Configure the DB node#####

plugin.IB.2=PTEM_CONFIG:PTIBConfigureDBNode

#PIA web server host
env.pia_webserver_host=@host@@domain@

#PIA web server port
env.pia_webserver_port=@httpport@

#PIA web server SSL port
env.pia_webserver_ssl_port=@sslport@

#PIA site name
env.pia_site_name=@piasite@

#Gateway host
env.gateway_host=@host@@domain@

#Gateway port
env.gateway_port=@httpport@

#Gateway SSL port
env.gateway_ssl_port=@sslport@
```

```
#Flag determines if the configuration is secure(https) / non secure (http); True-> ⇒

https, false->http
env.use_ssl_gateway=false

#Flag determines if the configuration is secure(https) / non secure (http); True-> ⇒

https, false->http
env.use_ssl_webserver=false

#Default user name
env.default_user_id=@userid@

#Default local node password
env.default_local_node_pass=

#Anonymous default user name
env.anonymous_default_user_id=@userid@

#Default local node
env.default_local_node=@localnode@

#External user id for wsdl_node
env.wsdl_external_user_id=@userid@

#External user password for wsdl_node
env.wsdl_external_pass=

#Flag to configure WSDL node
env.configure_wsdl_node=false

#Authentication TokenType for WSDL; Possible Values: NONE=none,STSD=SAML Token,USRT⇒

=Username Token
env.wsdl_node_tokentype=NONE

#Encryption for WSDL node
env.wsdl_node_tokenencrypted=0

#Digital Signature for WSDL node
env.wsdl_node_tokensigned=0

#Encryption Level for WSDL node; Possible Values: A=All, B=body and H=Header level ⇒

encryption
env.wsdl_node_ibencryptionlevel=A

##### Configure gateway URL, load connectors, define node in the gateway#####

plugin.IB.3=PTEM_CONFIG:PTIBConfigureGatewayNodes

#Gateway host
env.gateway_host=@host@@domain@

#Gateway port
env.gateway_port=@httpport@

#Gateway SSL port
env.gateway_ssl_port=@sslport@

#Flag determines if the configuration is secure(https) / non secure (http); True-> ⇒

https, false->http
env.use_ssl_gateway=false

#Default local node
env.default_local_node=@localnode@

#Gateway user name
env.gateway_user=administrator
```

```
#Gateway user password
env.gateway_password=

#Application server host name
env.ib_appserver_host=@host@@domain@

#Jolt port
env.ib_jolt_port=@jslport@

#Node proxy user name
env.ib_node_proxy_userid=@userid@

#Node proxy user password
env.ib_node_proxy_password=

#Tools Release
env.tools_release=@tools_release@

#Application server domain password
env.ib_appserver_domain_password=

#Virtual node
env.ib_virtual_node=

#Set as default node if this is set to true
env.ib_set_as_default_node=true

#Execute MAP File Synchronization
env.ib_synchronize_map_files=Y

##### Set keystore password, proxy host, proxy port, non proxy host in gateway prop⇒

erties file#####

plugin.IB.4=PTEM_CONFIG:PTIBConfigureGatewayProperties

#Gateway host
env.gateway_host=@host@@domain@

#Gateway port
env.gateway_port=@httpport@

#Gateway SSL port
env.gateway_ssl_port=@sslport@

#Flag determines if the configuration is secure(https) / non secure (http); True-> ⇒

https, false->http
env.use_ssl_gateway=false

#Gateway user name
env.gateway_user=administrator

#Gateway user password
env.gateway_password=

#Gateway keystore password
env.gateway_keystore_password=

#Gateway proxy host
env.gateway_proxy_host=

#Gateway proxy port
env.gateway_proxy_port=

#Gateway non proxy hosts
env.gateway_non_proxy_hosts=@host@@domain@|localhost|*.oracle.com

##### Active IB Pub/Sub Domain.#####

#plugin.IB.5=PTEM_CONFIG:PTIBActivateDomain
```



```
#Active Domain Retry Count
domain.activate_retry_count=10

#Active Domain Wait Time
domain.activate_wait_time=10

##### Activate IB Queues.#####

#plugin.IB.6=PTEM_CONFIG:PTIBActivateQueues

#Possible Values: PS_ALL to activate all queues, Else provide comma separated value⇒

s for queue names
queue.activate_queue_list=PS_ALL

#Active Queue Status; Default Value of queue status is 1
queue.activate_queue_status=1


##### Installing a RootCA certificate to the database.#####

configure=true
plugin.1=PTEM_CONFIG:PTImportCertificatestoDB

env.cert_type=ROOT
env.cert_alias=LBTest1
env.cert_issuer_alias=LBTest1
env.cert_string=----BEGIN CERTIFICATE----
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tDQpNSUlFS1RDQ0F4R2dBd0lCQWdJVVpqZlRpYzRWeEc2TGt⇒

xd09rcVhoRjliRlVld3dEUVlKS29aSW
----END CERTIFICATE----
env.cert_file_path=/tmp/rootCert.cer

##### Installing a Remote Node certificate whose RootCA is already added to the dat⇒

abase.#####

configure=true
plugin.1=PTEM_CONFIG:PTImportCertificatestoDB

env.cert_type=NODE
env.cert_alias=ServerCert
env.cert_issuer_alias=PeopleTools RootCA
env.cert_string=<Certificate PEM data> -→ Can be a single certificate / chain of ce⇒

rtificates
env.cert_file_path=/tmp/remoteCert.cer

##### Installing a RootCA for Remote Node and then Remote Node Certificate.#####
#In this case, we will need two template files. One contains the RootCA certificate⇒

 properties and the other contains Remote Node certificate details.

configure=true
plugin.1=PTEM_CONFIG:PTImportCertificatestoDB

env.cert_type=ROOT
env.cert_alias=LBTest1
env.cert_issuer_alias=LBTest1
env.cert_string=----BEGIN CERTIFICATE----
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tDQpNSUlFS1RDQ0F4R2dBd0lCQWdJVVpqZlRpYzRWeEc2TGt⇒

xd09rcVhoRjliRlVld3dEUVlKS29aSW
----END CERTIFICATE----
env.cert_file_path=/tmp/rootCert.cer

configure=true
decrypt_password=true
```

```
plugin.1=PTEM_CONFIG:PTImportCertificatestoDB

env.cert_type=NODE
env.cert_alias=LBServerCert
env.cert_issuer_alias=LBTest1
env.cert_string=<Certificate PEM data> -→ Can be a single certificate / chain of ce⇒

rtificates
env.cert_file_path=/tmp/remoteCert.cer
```

