Managing High Availability and Resiliency

This topic documents the basic approach used to support operational continuity, the ability to maintain availability during routine maintenance, and disaster recovery requirements for a Cisco Process Orchestrator. Using the production-level planning and operations described in the following topics, implementing these features should result in as much as 99.95% up time for your site:

- Understanding High Availability and Resiliency
- Advance Planning
- Ensuring Operational Continuity
- Maintaining Availability During Routine Maintenance
- Performing Disaster Recovery

Related Topics

- Handling Restarts and Failures
- Windows Performance Counters

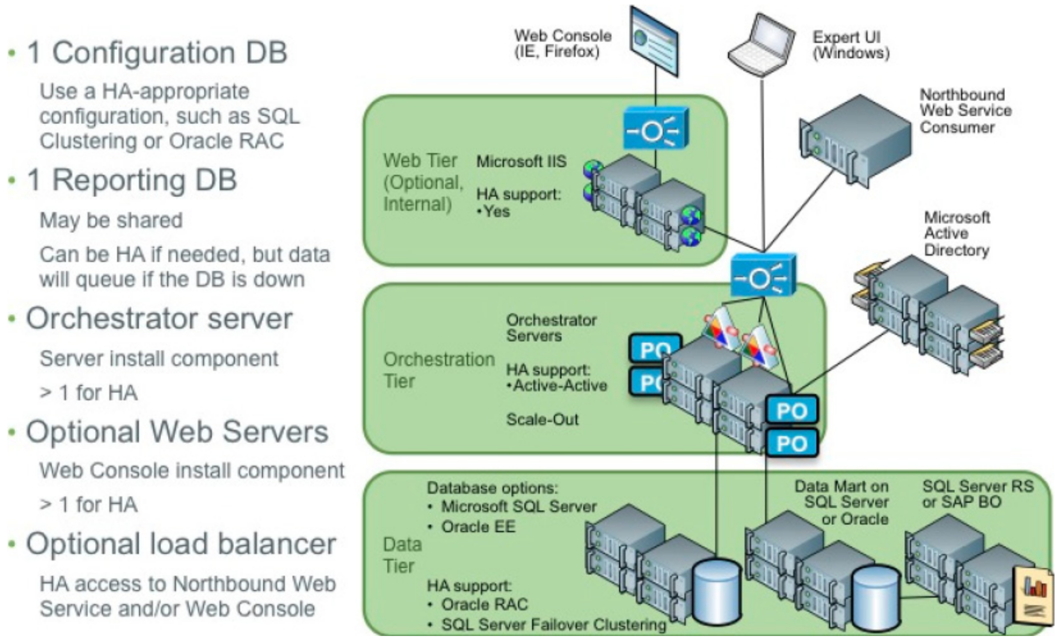
Understanding High Availability and Resiliency

The Cisco Process Orchestrator active-active server provides a highly available and scalable solution, the goal of which is to virtually eliminate down time due to hardware application failures. This solution, which protects critical pieces of the system from failure and excessive loads, includes these features:

- Rather than a single server performing all of the work (running processes, monitoring triggers, and so on), a Process Orchestrator environment can contain many servers. If one server fails, as long as another is available to take its place, useful work can continue.
- In a Process Orchestrator environment, all servers connected to the system share the workload, which helps increase scalability and performance. The more servers that are available in the system, the less work each of them must perform.
- A Process Orchestrator environment is available during routine maintenance, such as rebooting the operating system, applying OS security patches, and performing minor upgrades (even to the Process Orchestrator itself).

The following figure illustrates the recommended Process Orchestrator high availability environment:

Figure 1 High Availability Environment



- 1 Configuration DB
  - Use a HA-appropriate configuration, such as SQL Clustering or Oracle RAC
- 1 Reporting DB
  - May be shared
  - Can be HA if needed, but data will queue if the DB is down
- Orchestrator server
  - Server install component
  - > 1 for HA
- Optional Web Servers
  - Web Console install component
  - > 1 for HA
- Optional load balancer
  - HA access to Northbound Web Service and/or Web Console

Web Console (IE, Firefox)
Expert UI (Windows)
Northbound Web Service Consumer
Web Tier (Optional, Internal)
Microsoft IIS
HA support: •Yes
Microsoft Active Directory
Orchestration Tier
Orchestrator Servers
HA support: •Active-Active
Scale-Out
PO
Data Tier
Database options: • Microsoft SQL Server • Oracle EE
HA support: • Oracle RAC • SQL Server Failover Clustering
Data Mart on SQL Server or Oracle
SQL Server RS or SAP BO

The Process Orchestrator environment consists of:

- A reporting database. Although the environment requires a reporting database to enable reporting features, the reporting database can span environments and therefore can be thought of as external to the environment.
- A configuration database. The configuration database is specific to *one environment*, and therefore requires a high availability-appropriate solution.
- Multiple Process Orchestrator servers
- Optional: Multiple web servers
- A load balancer in front of the northbound web service and/or web servers (optional). For information about installing a load balancer, see the *Cisco Process Orchestrator Installation Guide.*

Because the Process Orchestrator environment supports high availability, the installation process:

- Requires an environment name (defaults to PRODUCTION).
- Differs for "subsequent" servers that are added to an existing high availability environment. During the installation of a secondary server, the installation process does *not* create the databases, configure security, import automation packs, or configure Windows accounts.

  However, the secondary server *does* require connectivity to an existing server in the environment. In addition, a security check is performed to ensure that the person installing the new server has the permissions to:
  - Add the server to an existing environment
  - Access the processes database
- Configures a default Process Orchestrator Windows Runtime User for Orchestration Server targets.

- Recommended: Assign security principals to default roles (typically Active Directory groups). The install process prompts you to specify the groups to be assigned to default security roles, but allows this step to be skipped. However, you will need to do this eventually or the environment will only work for the account that installed it.

Note Do not use local security groups and users in a high availability environment. By their nature, local security groups and users are accessible to one Process Orchestrator server, but are not accessible to another server. This means that they will be very confusing to configure and work with because server A will have a different "view" of permissions than server B.

In addition, because the Process Orchestrator environment supports high availability:

- The license key is tied to a computer name. This association is checked only when the license is applied. Therefore, if you have a high availability environment with three servers, for example, you can obtain a license key tied to one of the servers and then apply that license using the UI connected to that server. After the license has been applied, the license loses its association with the first server's name and becomes available to all servers in the environment.
- Process execution and event monitoring are automatically "equally" divided between all Process Orchestrator servers, with the following exceptions:
  - Only the *current* load is taken into account.
  - CPU, disk space, and other performance factors are *not* taken into account.
  - Due to the technical constraints of adapter connections, some work must be performed on a specific node.
- There are several constructs in Process Orchestrator that relate to files that exist on the servers. These constructs must now store files on some network share, which is available to all servers, so that if one server writes the file and fails, or if work is shifted for load balancing, other servers can access it to continue the automation. Ideally the network share would not be a directory on some specific Windows server, but a highly available location with redundancy and fault tolerance, such as an HA NAS system. Examples of such elements that should now use a share in a multi-server Process Orchestrator environment are:
  - Automation summaries, which are stored as XML files. The server must log in to write automation summaries from multiple servers.
  - Files that are written by some process and later read by the same or another process such as email attachments or FTP files. Although Process Orchestrator provides backward compatibility so that processes that rely on the well-known Windows Computer target can run after the upgrade, processes that rely on file locations *local* to a Process Orchestrator server will not be high availability-ready.
- Windows Computer targets are created for each Process Orchestrator server. However, the singleton (well-known) Windows Computer target from prior Process Orchestrator versions still exists and represents the first server in the environment. Processes that rely on the well-known Windows computer target will not be high-availability ready.
- The Core Automation Pack contains a well-known target group that includes all Process Orchestration server targets.

Note When writing high-availability processes, use this group instead of the well-known Windows computer target ID.

Advance Planning

Ensuring both operational continuity and disaster recovery preparedness requires:

1. Creating a support team (see Creating the Cisco Process Orchestrator Support Team).
2. Planning for high availability by installing multiple Process Orchestrators on virtual machines (see Planning for High Availability).
3. Using one of the recommended databases (see Storing the Information).
4. Considering the dependencies Process Orchestrator has on other services (see Considering Dependencies on Other Services).
5. Develop a strategy for performing regular database and system backups.

Creating the Cisco Process Orchestrator Support Team

To maintain operational continuity and ensure that the people on your site are prepared for any emergency:

Step 1 Create your Cisco Process Orchestrator Support team *well in advance of any disaster.*

When a disaster occurs, the first step of any recovery process should be to contact this team. The members of this team should know the prescribed (and latest) list of procedures, and each member should have a pre-defined role in the recovery process.

This team should consist of:

- The team assigned to day-to-day administration and operations.

Note Although Process Orchestrator can be useful across a broad span of IT users, it needs an administrator/operator. This person will use the Operations Workspace (see Monitoring Processes) to monitor the health of processes and receive operational alerts, and will typically serve as the primary contact for customer support issues.

- The database administrator (DBA) responsible for the database if the disaster is related to the Process Orchestrator database.
- The manager of the VM farm on which the Process Orchestrator servers run.
- Personnel who can run the software that restores the Process Orchestrator server. This can be the VM farm operator.

Step 2 Because outages in individual upstream systems can affect processes that span technologies and services, maintain the contact information for the owners of any upstream systems (see Considering Dependencies on Other Services). This information should include current names and phone numbers.

Planning for High Availability

A single-server environment does not provide high availability, so you should plan to implement multiple Process Orchestrator active-active servers. When you are planning your Process Orchestrator environment, consider these recommendations:

- Be sure that all Process Orchestrator servers have similar hardware and software configurations; that is, the same (or similar) amounts of memory, number of CPUs, OS levels, and so on.

  Cisco Process Orchestrator's high availability solution does not take hardware or operating system differences into account when distributing load. When new work arrives to be executed by Cisco Process Orchestrator, only the current load (the number of process instances and monitored triggers) is considered. Over time, more work will be assigned to less loaded or more capable Process Orchestrator servers, but the amount of memory, CPUs, or storage is not directly taken into account when assigning work.
- Install the Process Orchestrator servers on virtual machines backed by networked storage. Using this approach, if a host fails, the VM can be migrated to a new host using a tool such as vCenter. Many customers find this method of fault tolerance preferable because the host can change to another site in disaster recovery situations.
- VM farms tend to be sets of machines connected to the same storage, and might be in the same data center. Consider replicating your snapshots to a secondary data center.

Moving to High Availability

Customers running multiple database and Process Orchestrator 2.x server installations running the *same content* against many targets should consider going to a single database and a multiple high availability server installation. To migrate to this environment:

Step 1 Upgrade one of the servers directly.

Step 2 Add additional high availability servers (on separate hardware, unrelated to the old 2.x servers).

Step 3 Create automation packs to include your targets from the non-upgraded 2.x environments.

Step 4 One by one, shutdown your old 2.x servers and at the same time, import the automation packs with targets from those servers into the new high availability environment.

Customers running multiple database and Process Orchestrator 2.x server installations running *different* content against many targets might not want to use high availability, but it can be accomplished. In this case, to migrate to this environment:

Step 1 Perform same steps listed above.

Step 2 Export whatever content you need from each 2.x server (in addition to the targets), then import the content back into the new high availability environment.

Storing the Information

Virtually all Process Orchestrator state information is stored in the database. Although there are two databases, only the configuration database is a hard upstream dependency (see Considering Dependencies on Other Services) and must be made highly available to prevent a database-based Process Orchestrator outage. The reporting database is used for auditing and reporting purposes, but Process Orchestrator does not depend on its presence or on it being up and running.

Process Orchestrator supports the following high availability databases:

- SQL Server Failover Cluster

- Oracle Real Application Clusters (see the Oracle RAC configuration steps in the *Cisco Process Orchestrator Installation Guide*)

Although the bulk of the Process Orchestrator server's state is stored in the database, the following important pieces of data are stored in the Process Orchestrator server system:

Table 14-1 Information Held in the Process Orchestrator Server System

Type of Information	Description
The keys used to encrypt security credentials	This key is stored in a controlled space in the Windows security subsystem of the Windows server hosting the Process Orchestrator server. Process Orchestrator stores runtime user names, passwords, and other credentials in the database so that it can connect using those credentials when running an activity against a target. Credentials are encrypted with a key specific to the Process Orchestrator environment before being stored in the database. This environment-specific key, which is stored in the Windows security system of each Process Orchestrator server, can be exported as a file, retained for disaster recovery purposes, and moved to a standby/backup server for recovery (see Saving the Process Orchestrator Server Data in Preparation for a Disaster). In a disaster recovery situation, the Process Orchestrator database is useless without the corresponding and separately stored encryption keys.
The Process Orchestrator server installation	This is core software that can be recreated from the installation media.
The server configuration file	This file configures certain properties of the Process Orchestrator server. For example, it tells the Process Orchestrator server which ports to open to talk to the client. It also tells the server where the database is and which credentials to use to access it.
Persistent queue files for data to be written to the reporting database	These files store reporting data so that it is not lost across server restarts (see Handling Restarts and Failures).

Considering Dependencies on Other Services

Process Orchestrator interacts with many different IT technologies and processes (see Considering Dependencies on Other Services). It is critically dependent on some technologies, such as its backend database on an Oracle or SQL Server (a hard upstream dependency), and on other technologies such as a DB2 database or VMware vCenter connection (soft upstream dependencies), only to the extent that processes interact with those technologies.

Similarly, other IT services might rely on Process Orchestrator. For example, an IaaS private cloud service built on Cisco Intelligent Automation for Cloud (that includes Process Orchestrator) will be non-operational or will have to revert to all-manual workarounds without a functioning Process Orchestrator.

Other IT services could operate in various degraded states without Process Orchestrator. For example, Incident Management in Remedy will still function as IT's Incident Management without a functional Process Orchestrator, but capabilities such as automated troubleshooting (a hard downstream dependency) and ticket enrichment (a soft downstream dependency) will not be available.

To summarize, when you are planning your Process Orchestrator environment, take into consideration all of these types of dependencies:

Table 14-2 Dependency Types

Dependency	Description
Hard	A component or service is required by the dependent service to be fully operational.
Soft	A component or service is not required by the dependent service to be fully operational, but functionality directly involving the component might be degraded or broken.
Upstream	The IT services that Process Orchestrator relies on.
Downstream	The IT services that rely on Process Orchestrator.

For more information about the dependencies on Process Orchestrator and Process Orchestrator's dependencies on other services, see the *Cisco Process Orchestrator Compatibility Matrix*.

Ensuring Operational Continuity

Ensuring operational continuity of your Process Orchestrator environment includes these actions:

- Backing Up Your Data
- Monitoring Processes and Events
- Handling Errors, Exceptions, and Diagnostics

Backing Up Your Data

In a disaster recovery situation, you will need copies of the following information, so be sure to regularly:

- Back up your database. Follow the standard database backup procedures for data recovery and the best practice database resiliency strategies of your database vendors, which should be provided in the documentation for the applicable database platform.
- Save the Windows security credentials encryption key for the Process Orchestrator environment. This only needs to be done once; this key is shared by all servers in a single Process Orchestrator environment. For information about how to save the key, see Saving the Process Orchestrator Server Data in Preparation for a Disaster.
- Save the Process Orchestrator server configuration file. For more information about this file, see Storing the Information.
- Take snapshots of the Process Orchestrator server VMs and save them at an alternate site (see Planning for High Availability).
- Save any customizations to the Web.config file in the Web Console server.

Saving the Process Orchestrator Server Data in Preparation for a Disaster

When making preparations for a disaster, the most important piece of information to save is the security key that a Process Orchestrator server uses to encrypt sensitive data before storing it to the database. This environment-specific key is stored in the Windows security system of each Process Orchestrator server.

Note In a disaster recovery situation, the Process Orchestrator database is useless without the corresponding and separately stored encryption key for the Process Orchestrator environment.

Another important piece of data is the Process Orchestrator server configuration file. For more information about this file, see Storing the Information.

To save this information:

Step 1 Once for the Process Orchestrator environment:

a. Export the security credentials encryption key to a file:

```
aspnet_regiis -px "Tidal Intelligent Automation Server" Keys.xml -pri
```

Aspnet_regiis.exe is a Microsoft.NET framework utility that resides in the .Net Framework folder. This folder is typically located at this location:

```
C:\Windows\Microsoft.Net\Framework64\{Version}\aspnet_regiis.exe
```

You might need to specify the full path to this utility if you have not defined the path in your Path environment variable

b. Save the Keys.xml file, then move it to a secure location (typically a standby or backup server) separate from the Process Orchestrator server.

Step 2 For each Process Orchestrator server, copy the server configuration file to a standby or backup server. You can find this file in the installation folder.

```
%INSTALLDIR%\Tidal.Automation.Server.exe.config
```

Monitoring Processes and Events

Each server in a Process Orchestrator environment publishes performance and events. A Process Orchestrator server can also provide data about its own health to operational support systems, and can interact with operational support systems with regard to the health of services for which it provides automation.

If you want to monitor the entire high availability environment, however, you must monitor *all* of the servers. Most application management systems include the ability to monitor the data that Process Orchestrator publishes, such as Windows event logs and performance counters.

Process Orchestrator can integrate with IT management or network management tools such as the Cisco Prime products. Process Orchestrator also provides direct out-of-the box integrations with other management tools, including:

- Microsoft System Center Operations Manager
- SAP CCMS and Solution Manager
- Remedy Service Desk

Process Orchestrator also provides numerous features to integrate in a generic manner with other management systems. For example, the triggered Process Orchestrator process can invoke a web service or run a script and pass in the incident, so the information can be published to another service assurance system.

Monitoring Processes

Process Orchestrator provides full visibility into the processes running on the system, including the status of all running processes, which activities have executed, succeeded, failed, and so on.

Use the Operations view to monitor the processes that are scheduled to execute, view processes that are currently running, and verify that processes have successfully completed. You can also start processes or interact with human steps in processes called Tasks. For more information about monitoring processes, see Monitoring Operations.

Managing Events

Process Orchestrator provides IT process records such as alerts and incidents, and uses these records to:

- Supply incidents and events for external tools it might be monitoring or managing.
- Manage the Process Orchestrator itself. For example, the Process Orchestrator Core automation pack performs some self-monitoring of the Process Orchestrator, and can raise an alert or incident if there is something that requires the Process Orchestrator administrator's attention.

Use the **Operations > Auditing > System** view to review the list of internal Process Orchestrator events, including notifications of warnings and errors, and information events about the general functioning of the system.

Note Events from all Process Orchestrator servers in a high availability environment appear in this view.

For more information about managing events, see Working with Events and Triggers.

Handling Errors, Exceptions, and Diagnostics

Use Windows event logs and Windows performance counters to monitor the Process Orchestrator platform, either of which can be monitored through Microsoft Windows directly or fed into the service assurance platform of choice. Because event logs and performance counters are the standard for monitoring Windows-based applications, most service assurance tools include the ability to monitor these elements.

- System event logs

  The Process Orchestrator server logs errors that can be useful when you need to diagnose failures to the Windows Event Logs. For example, if there is an error connecting to a database, this will be logged as an event.

  These logs can also be picked up by most enterprise systems management tools if you want a view of Process Orchestrator health within your systems or application management console.

  For more information about viewing system events, see Working with Events and Triggers.

- Windows performance counters

  The Process Orchestrator server publishes a number of Windows performance counters concerning its operations. For a list of Windows performance counters, see Windows Performance Counters.

Note Be sure to monitor the system event logs and Windows performance counters from *all* Process Orchestrator servers in an environment. Monitoring one server is not sufficient to monitor the health of all servers.

Maintaining Availability During Routine Maintenance

This section discusses how to maintain the availability of the servers in the Process Orchestrator environment during routine maintenance:

- Handling Routine Maintenance Outages
- Adding More Capacity
- Relocating a Process Orchestrator Server
- Maintaining the Database

Handling Routine Maintenance Outages

To maintain availability of the overall high availability environment during routine outages, such as Microsoft security patches and minor Process Orchestrator upgrades that do not affect the database schema:

Step 1 Shut down one Process Orchestrator service (while keeping the others running). Wait for the Process Orchestrator service to completely stop via a friendly shutdown.

Step 2 Install the Microsoft patches or Process Orchestrator patches on that server.

Step 3 Reboot as necessary, then restart the Process Orchestrator service.

Step 4 Repeat steps 1-3 on all Process Orchestrator servers, one at a time, to maintain availability of the overall high availability environment.

For more information about shutting down and restarting Process Orchestrator, see the *Cisco Process Orchestrator Installation Guide*.

Adding More Capacity

To add more capacity to your Process Orchestrator environment, just add another Process Orchestrator server. Before you can do that, you must:

- Be able to connect to an existing Process Orchestrator server
- Have permission to add a new server
- Acquire the credentials for connecting to the Process Orchestrator process database

For information about how to add a new server, see the *Cisco Process Orchestrator Installation Guide*.

Relocating a Process Orchestrator Server

To relocate a Process Orchestrator server:

Step 1 Add a new server to the environment (see Adding More Capacity).

Step 2 Uninstall Process Orchestrator from the old server (see the *Cisco Process Orchestrator Installation Guide)*.

Maintaining the Database

The following sections describe how to maintain the Process Orchestrator database:

- Grooming the Database
- Database Performance Best Practices

Grooming the Database

Process Orchestrator provides settings to control grooming of the following types of objects:

- Various sections of its Process and Reporting databases.
  - The Process database instances are primarily useful for viewing prior instances in the main expert UI, understanding the specific activities which were executed, querying the activity instance inputs, outputs, and other execution details typically useful in troubleshooting and development scenarios.

    Grooming the Process database can help optimize performance. Reducing the database size can improve database insert speeds, but at a cost of being able to view older process instances in the UI. If you perform complex views including large numbers of historic processes, the database and UI must deal with the larger data volumes. Larger views might also mean larger data payloads coming through the server to the UI. This factor therefore not only affects the database layer, but also the server and UI performance.

    Grooming the Process database as tightly as business scenarios allow optimizes performance of both the Process Orchestrator Server and UIs. For example, if no business requirement exists to monitor and troubleshoot failed processes past the end of a shift, setting grooming for one day might be appropriate.
  - Data about completed process instances is available on a long-term basis in the Process Orchestrator Reporting database. The Reporting database provides information on which processes have run, when they started and ended, whether they were successful or failed, how they were started, and who started the processes in long term storage. The Reporting database has information about processes only, not activities within the process.
- Task instances, such as alerts, incidents, and change requests. These objects are groomed *only* upon their completion, which means that an open or active task can remain in the database forever. By default, the tasks can stay in the database for a long time.
- Audit data, which is groomed less aggressively because it is more important to keep around longer.

Using the default settings, grooming is automatic, but you can tune these settings to your needs. For example, there are options to:

- Control how much data is retained for completed instances.

- Mark expired tasks as completed.
- Configure a process to archive on failure. This allows low data and performance load during normal/successful executions of the process, while archiving failures so that they can be debugged and diagnosed.
- Start grooming immediately (rather than to wait for the scheduled time).

To change the grooming settings for the Process database, see Managing the Process Database.

Database Performance Best Practices

Proper database server hardware and routine database maintenance can have substantial effects on performance. Although the Process Orchestrator ships with performance-optimized schemas, including the relevant indices, you must install and operate these databases.

Ideally, to optimize performance, a database administrator (DBA) familiar with best practices should prescribe and configure the server hosting the database platform containing the Process Orchestrator databases. The DBA should also be involved in the installation of the database and should perform routine maintenance.

In high performance scenarios, the following best practices can dramatically affect performance:

- Run regular backups of the Process Orchestrator database. Without backups, the transactional logs of the database will grow fairly substantially, resulting in large files and poor database performance, and ultimately affecting Process Orchestrator's performance.
- Always run the database server on physical hardware, not on a VM.
- Use a separate host server for the database instead of running the database server on a Process Orchestrator server.
- Disk throughput (I/O) is also an important metric for large scale deployments. Use a separate high speed disk for the database, operating system, program files, and swap files.
- Provide sufficient memory to avoid paging. In very large installations, Process Orchestrator has benefited from databases running 16 or even 32GB of RAM.
- Use a high-speed network connection. Typically this means the database is "close" to the Process Orchestrator server, and certainly in same data center.

For best practices, refer to the documentation associated with your chosen database platform.

Performing Disaster Recovery

This section discusses the following topics:

- Recovering from a Server Failure
- Recovering from a Reporting Database Failure
- Recovering from an Entire Environment Failure

Recovering from a Server Failure

In a multi-server high availability environment, there is very little you need to do to recover from a single server failure. When a Process Orchestrator server fails (due to, for example, a network outage, disk failure, or software failure), the remaining Process Orchestrator servers within the environment will:

1. Recognize the failure (within a few seconds).
2. Report the failure in event logs.
3. Redistribute the work that was performed by the failed Process Orchestrator server among the remaining healthy servers.

If the server failure is recoverable, simply bringing the server or service back up will put it back into the high availability environment and operations will resume.

If the server failure is not recoverable (for example, a hardware failure), you can add a new Process Orchestrator server to the environment. There is very little need to recover the failed server; it will not be considered healthy, and no work will be assigned to it. To delete the old server from the environment, choose **Administration > Orchestration Servers > Remove**.

Recovering from a Reporting Database Failure

If only your Reporting database has failed, there are several options:

- If possible, recover the database server. Operations will resume as before, and no changes are needed.
- If the database server is not recoverable but the database is (because there is a backup), restore the database to a new database server (see Restoring the Database to a New Database Server).
- If both the database server and the database are not recoverable, recreate the database on a new database server (see Recreating a New Database on a New Database Server).

Restoring the Database to a New Database Server

To restore the database to a new database server:

Step 1 Open a console connecting to *any* Process Orchestrator server in the environment.

Step 2 Remove the existing Reporting Database connection and connect to the new database server (see Managing the Report Database).

The changes will propagate throughout the entire high availability environment.

Recreating a New Database on a New Database Server

To create a new database on a new database server:

Step 1 Open a console connecting to *any* Process Orchestrator server in the environment.

Step 2 Choose **Administration > Database Settings**, right-click **Report Database** and choose **Remove Cisco Process Orchestrator Reporting Database Connection**.

Step 3 Create a new Reporting database on the new database server (see Creating a Reporting Database Connection).

The changes will propagate throughout the entire high availability environment.

Recovering from a Total Database Failure

If your entire database has failed:

Step 1 Contact your Cisco Process Orchestrator Support team (see Creating the Cisco Process Orchestrator Support Team). To ensure that the latest procedures are followed and to minimize downtime, the support team should be contacted as early as possible and involved throughout the process. Do not wait to call them until the disaster recovery process goes astray.

Step 2 Stop the Process Orchestrator service on all of the server machines.

Step 3 Restore the database from the latest backup to a new database server (or new database schema). The restore will depend on the database version; follow the manufacturer's instructions for your database server.

Note If you do *not* have a backup, you *cannot* recover the database. In that case, you must reinstall the entire environment.

Step 4 Log in to the first Process Orchestrator server, then use the Database User utility (Tidal.Automation.Server.DatabaseUserConfigurationUtility.exe) to point the server to the new database and/or new credentials.

Step 5 Start the Process Orchestrator service on this server.

Step 6 Launch the Process Orchestrator console, then verify and correct Reporting database configuration if needed. This step is required if the Reporting database connection or the reporting credentials have changed.

Step 7 Log in, reconfigure, and restart (Step 4 and Step 5) all other Process Orchestrator servers in the environment.

Recovering from an Entire Environment Failure

If your environment fails but you have snapshots of the Process Orchestrator server VMs saved at an alternate site, use the procedure described in Recovering from a Total Database Failure instead.

Follow these steps when your environment fails and you *do not* have snapshots of the Process Orchestrator server VMs saved at an alternate site:

Step 1 Contact your Cisco Process Orchestrator Support team (see Creating the Cisco Process Orchestrator Support Team). To ensure that the latest procedures are followed and to minimize downtime, contact the support team as early as possible and involve them throughout the process. Do not wait to call them until the disaster recovery process goes astray.

Step 2 Restore the database from the latest backup to a new database server (or new database schema). The restore will depend on the database version; follow the manufacturer's instructions for your database server.

Note If you do not have a backup, you *cannot* recover the database. In that case, you must reinstall the entire environment.

Step 3 To import the Windows security credentials encryption key from the Keys.xml file created in Saving the Process Orchestrator Server Data in Preparation for a Disaster to the new (first) Process Orchestrator server in the environment, enter the following command (*do not cut and paste*):

```
aspnet_regiis -pi "Tidal Intelligent Automation Server" Keys.xml -exp
```

Aspnet_regiis.exe is a Microsoft.NET framework utility that resides in the .Net Framework folder. For example, this folder is typically located at this location:

```
C:\Windows\Microsoft.Net\Framework64\{Version}\aspnet_regiis.exe
```

You might need to specify the full path to this utility if you have not defined the path in your Path environment variable.

Note It is not enough to just bring the Process Orchestrator environment up; you also need the security credentials encryption key to apply to the server at the new site to make this database copy usable. If you do not have a backup of this key, you *cannot* recover the database. In that case, you must reinstall the entire environment.

Step 4 Reinstall the core Process Orchestrator software from the installation media. Be sure to reinstall the *same version* and all of the updates/hotfixes.

Step 5 Replace the server configuration file on the new machine with the file copied in Saving the Process Orchestrator Server Data in Preparation for a Disaster.

Note If the file contents contain any path (such as the log file path), make sure that the path is still valid in the new server machine.

Step 6 Use the Database User utility (Tidal.Automation.Server.DatabaseUserConfigurationUtility.exe) to point the server to the restored database and/or credentials.

Step 7 Start the Process Orchestrator service on this server.

Step 8 Launch the Process Orchestrator console, then verify and correct the Reporting database configuration if needed. This step is required if the Reporting database connection or the Reporting credentials have changed.

Step 9 Install any additional Process Orchestrator servers as *new* High Availability servers (see the *Cisco Process Orchestrator Installation Guide*). The additional servers will receive the database settings and encryption keys from the first server.

Step 10 Because the Process Orchestrator server is now installed on a different server:

a. Review the list of Windows targets in the Process Orchestrator, remove the Windows computer targets that point to "old" Process Orchestrator servers that are no longer available, and disable the old server target in the target definition.

b. If the Web Server is on a different computer or virtual machine from the Process Orchestrator server computer, and if the Web Server survived the outage but the Process Orchestrator servers did not, choose **File > Environment Properties** to verify and update the Web Console configuration file location.

c. Use the Core Functions Adapter properties to review and update the Automation Summary configuration.

Step 11 Choose **Administration > Orchestration Servers** and remove the old servers.

Handling Restarts and Failures

This section discusses the following topics:

- Handling Running Processes During Restarts and Failures
- Handling Reporting Data Queues During Restarts and Failures
- Handling Scheduled Triggers During Restarts and Failures
- Handling Events During Restarts and Failures

Handling Running Processes During Restarts and Failures

The rules described in this section apply to moving a process from one Process Orchestrator server to another server when a single server fails. If server A goes down, then if the process hasn't stopped on a non-restartable activity, it will move to another active server.

By default, Process Orchestrator persists all data about running processes so that state is preserved across service restarts; whether the restarts are planned or unexpected, process and activity states are persisted to the Process Orchestrator database. When the Process Orchestrator server shuts down in a friendly manner, it allows in-flight activities time to complete, but blocks new activities from running. This typically allows activities that might not be marked as restartable to complete so that processes do not fail during friendly shutdowns.

Generally, processes will pick up and start running again after a restart or failure; in some activities, this is configurable by the user. Many activities, if they were running when the server shut down, can be restarted without any side effects. Some examples:

- If an activity is passive (such a configuration query), Process Orchestrator can simply run the query again.

- Other activities can have side effects and cannot be rerun without consequence. For example, if a CLI adds some entry for a device to a file, running the CLI again will cause two entries for the device. In these cases, Process Orchestrator shows these processes as failed after the restart; the operator must examine the failures, then rerun the failed processes.
- The Web HTTP Request has a “Restart when interrupted” check box. Click this option if the HTTP Request is performing a read operation, in which case it should be restartable, but do not click it when performing a write operation because you could accidentally create multiple entries.

Process Orchestrator offers process Start Points that process authors can use to give operators restart points when a process needs to be restarted. If a volatile activity is run in Process Orchestrator and has not returned a status and Process Orchestrator shuts down, when Process Orchestrator restarts, this activity will not resume and the results of what the activity was supposed to do are lost.

Adapters determine whether to declare activities as restartable. The Process Orchestrator server also allows activities to save their state as they run, which allows long-running activities to be restarted. For example, a Create Approval activity might save a state that the approval request task has been created but the activity is blocked waiting on an approval.

Process Orchestrator's writing of state is transactional. Process Orchestrator will not proceed to the next step until the state change is committed to the database. There should be no time gap where process state can be lost. This persistence has a cost. With every step in every process, Process Orchestrator writes to the database to update states such as when the activity starts, stops, etc. Where this performance hit is inappropriate, Process Orchestrator provides a setting to turn restartability off for a process. Many processes do not need to restart after a shutdown or failure; it might often be acceptable to rerun the process at the next scheduled interval or on the next instance of the trigger. For example, many SAP diagnostic processes fall into this category. In the case of SAP diagnostics, Process Orchestrator's SAP automation pack process definitions have restartability turned off on analysis processes. If the server fails, the state of the SAP server with respect to alerts might not be the same after restart anyway. It is often best to not restart these processes but to instead to relearn the state of the SAP system. Also, by disabling restartability, the performance impact of maintaining state is reduced.

Handling Reporting Data Queues During Restarts and Failures

To improve performance, reporting data is held in queues and is occasionally written to the database in bulk rather than an item at a time. During a friendly shutdown, the queue is flushed so that the reporting data is fully synchronized to the reporting database. In the event of a failure, the data in the queue will be present on restart and will be sent whenever the Process Orchestrator Server starts up; no data will be lost. However, if an unexpected failure occurs for which there is a need to reinstall the Process Orchestrator server on a different computer or restore the computer hosting the Process Orchestrator server from a backup, reporting data from these queues is lost.

Handling Scheduled Triggers During Restarts and Failures

In a Process Orchestrator high availability environment, schedules that are monitored on server A will move to Server B when server A fails. There is a small window (of a few seconds) during which a schedule could be missed while server B is detecting server A’s failure.

Handling Events During Restarts and Failures

Process Orchestrator adapters resume sending events after the Process Orchestrator server restarts. While the adapter is still responsible for the implementation in this area, existing Process Orchestrator adapter implementations attempt to be consistent with this guaranteed delivery design. For example, when the SAP adapter reads CCMS alerts or the Remedy adapter polls for Incident state, the adapter stores the last-read record so that it can resume reading records starting with the next entry when the Process Orchestrator server resumes. For non guaranteed-delivery, network-initiated event technologies such as stop traps, the Process Orchestrator server cannot know about events that occurred while it was down. If required, many of these technologies can use highly-available intermediaries to persist the transient events. For example, there are tools to listen for stop traps and convert them to persistent stores such as log files or Windows events.

There are two types of event systems in the Process Orchestrator server:

- Event-based triggers. Because of the adapter implementations (see Handling Running Processes During Restarts and Failures), event-based triggers are not lost across server restarts or failures. Like state management in other areas of the product, trigger submissions are transactional. When Process Orchestrator adapters send a trigger to the Process Orchestrator server, processes depending on that trigger are initiated as a part of the submission. As with any other processes, these process instances are persisted to the database so that after a restart, the triggered processes are running. With the exception of transient, non-persisted events, such as stop traps or performance thresholds, there should be no time gap where events should be lost such that triggers fail to launch.
- Correlation. The Correlation feature allows a server to tie together a related series of events. This is achieved through a caching mechanism to ensure that the event data is available to the Correlate activities in processes when needed. This cache of events is not retained across server restarts, which can cause issues with some Correlate activities. Correlate activities can be used to make a process wait for a certain condition, or branch based on how many events with particular properties have been received in a particular time frame. Because the cache is not persisted across server restarts, events received before the restart will not be matched by Correlate activities. These effects can affect accuracy of a process “decision” made based on the data from a Correlate activity.

  In practice this has been found to not be an issue for the following reasons:

  - First, correlation is a fairly advanced feature in the product and is very rarely used in a majority of scenarios.
  - Second, best practices also dictate that a correlation time frame is configured to be fairly small, which will minimize both the likelihood and impact of this compound condition. The time required to restart a server will take up some or all of this time frame if the correlation time frame coincides with a server restart.
  - Third, these correlations are typically used in diagnostic situations, and often the events repeat if the problem recurs. Where a problem is still occurring, Process Orchestrator will typically pick up the condition the next time the diagnostic process launches.
  - Finally, Process Orchestrator automation packs tend to set these processes as non-persistent anyway so they are not restarted. If the Process Orchestrator server is down, it is usually best to get a fresh view of the health of the component being monitored rather than depend on the status before the Process Orchestrator server restart. In many cases, the failure is resolved and it just creates noise to record the old diagnosis.

Windows Performance Counters

A Cisco Process Orchestrator server publishes performance counters that can be used to monitor its health and diagnose performance issues. These counters are grouped into several performance objects, which are discussed in the following sections.

The following standard Windows performance counters are also of interest:

- Memory—Pages/sec
- Processor—% Processor Time
- Process—Private Bytes
- Process—Handle Counts
- Physical Disk

Cisco Process Orchestrator Object

Table 14-3 Windows Performance Counters - Cisco Process Orchestrator

Counter	Description
Running Process Instances	The number of running process instances
Running Activity Instances	The number of running activity instances
In-Memory Activity Instances	The number of in-memory activity instances. An in-memory activity instance might be running now or have run within a process that is running now. For example, if you have a single 10-step sequential process, there will always be only one active activity instance. But as the workflow continues to run, the number of "in-memory" activities will grow as the execution progresses. When the process stops running, the number drops to zero.
In-Memory Process Instances	The number of in-memory process instances. In-memory activity instance might be running now, or have run within a process which is running now. For example, if you have a single 10-step sequential process, there will always be only one active activity instance. But as workflow continues to run the number of "in-memory" activities will grow as the execution progresses. When the process stops running, the number drops to zero.
Schedules Monitored	The number of schedules being monitored in Process Orchestrator.
Schedules Met	The number of monitored schedules met in Process Orchestrator. If there is only one process definition that has an "Every day, every 5 minutes" schedule trigger, then "Schedules Monitored" will be 1 for as long as the server runs. "Schedules Met" will grow by one every five minutes as the schedule fires process instances.
Events Monitored	The number of events being monitored in Process Orchestrator.

Table 14-3 Windows Performance Counters - Cisco Process Orchestrator

Counter	Description
Events Fired	The number of monitored events fired in Process Orchestrator. If there is only one process definition that has an "Every day, every 5 minutes" schedule trigger, then "Events Monitored" will be 1 for as long as the server runs. "Events Fired" will grow by one every five minutes as the schedule fires process instances.
Web Service Calls	The number of calls to the Process Orchestrator Northbound Web Service.
Completed Activity Instances /sec	The rate (per second) at which activities are being completed (with success or failure).
Completed Process Instances /sec	The rate (per second) at which running process instances are being completed (with success or failure).
Started Activity Instances /sec	The rate (per second) at which activity instances are being started.
Started Process Instances /sec	The rate (per second) at which process instances are being started.

Cisco Process Orchestrator Adapter Object

There are separate instances for each adapter.

Table 14-4 Windows Performance Counters - Cisco Process Orchestrator Adapters Object

Counter	Description
Running Activities	The number of activities running under this adapter.
Activities Run	The number of activities this adapter has run
Disappearances	The number of times this adapter has disappeared unexpectedly.
Event Fired	The number of monitored events fired by this adapter.
Events Monitored	The number of events being monitored by this adapter.
Excessive Memory Restarts	The number of times this adapter has been restarted due to excessive memory usage.
Host Process ID	The ID of the process that hosts this adapter.
Idleness Shutdowns	The number of times this adapter has shut down due to idleness.
Requested Restarts	The number of times this adapter has requested to be restarted.
Starts	The number of times this adapter has started
Targets Managed	The number of targets this adapter is managing.

Cisco Process Orchestrator Persistent Queue Object

There are separate instances for each queue.

Table 14-5 Windows Performance Counters - Cisco Process Orchestrator Persistent Queue Object

Counter	Description
%Used Space	The maximum space for the queue is preallocated; this counter measures how much of the queue is used.
Queue Item Submits/Sec	Number of items submitted to the persistent queue per second.
Queue Items	This is the current number of items in the queue. This number will vary as new items are added by the process execution and removed as the engine records them into the reporting database.

Cisco Process Orchestrator Data Access Layer Operations Object

There are separate instances for each type of DAL (Data Access Layer) operation.

Table 14-6 Windows Performance Counters - Cisco Process Orchestrator Data Access Layer Operations Object

Counter	Description
# of Data Access Layer Calls	Number of times a particular DAL (Data Access Layer) call was made.
# of Errors in Data Access Layer Calls	Number of times a DAL call completed with an error.
Average Data Access Layer Call Duration (milliseconds)	Average time it took to complete this type of DAL operation.
Cumulative Data Access Layer Call Duration (milliseconds)	Total time spent in all DAL operations of this type.
Longest Data Access Layer Call Duration (milliseconds)	The time it took for the longest of DAL operations of this type to complete.

Cisco Process Orchestrator SAP Connections

There are separate SAP requests in each queue.

Table 14-7 Windows Performance Counters - Cisco Process Orchestrator SAP Connections

Counter	Description
RFC calls in waiting	The number of RFC calls waiting in the queue. The maximum number of concurrent calls allowed is 1 by default.
RFC calls in working	The maximum number of RFC calls that are concurrently working.

Table 14-7 Windows Performance Counters - Cisco Process Orchestrator SAP Connections

Counter	Description
Total RFC Calls	The total number of RFC calls made in the SAP connection
SAP alerts	The total number of SAP alerts monitored on the SAP or SAP solution manager connection.
Total SAP Requests	The total number of SAP requests made on the SAP or SAP solution manager connection.
Queued SAP Requests	The total number of SAP requests waiting in the queue. Each SAP or SAP solution manager connection processes one SAP request at a time.