



## PI to PI Interface

### User Guide

OSIsoft, LLC  
777 Davis St., Suite 250  
San Leandro, CA 94577 USA  
Tel: (01) 510-297-5800  
Fax: (01) 510-357-8136  
Web: <http://www.osisoft.com>

PI to PI Interface User guide

© 1997-2014 by OSIsoft, LLC. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, photocopying, recording, or otherwise, without the prior written permission of OSIsoft, LLC.

OSIsoft, the OSIsoft logo and logotype, PI Analytics, PI ProcessBook, PI DataLink, ProcessPoint, PI Asset Framework (PI AF), IT Monitor, MCN Health Monitor, PI System, PI ActiveView, PI ACE, PI AlarmView, PI BatchView, PI Coresight, PI Data Services, PI Event Frames, PI Manual Logger, PI ProfileView, PI WebParts, ProTRAQ, RLINK, RtAnalytics, RtBaseline, RtPortal, RtPM, RtReports and RtWebParts are all trademarks of OSIsoft, LLC. All other trademarks or trade names used herein are the property of their respective owners.

#### U.S. GOVERNMENT RIGHTS

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the OSIsoft, LLC license agreement and as provided in DFARS 227.7202, DFARS 252.227-7013, FAR 12.212, FAR 52.227, as applicable. OSIsoft, LLC.

Version: 3.8.9.x

Published: 11 April 2014

# Contents

---

<b>Introduction to the PI to PI Interface.....</b>	<b>1</b>
Related manuals.....	3
Supported operating systems.....	3
Interface limitations.....	4
<b>How the PI to PI Interface works.....</b>	<b>5</b>
Interface startup.....	5
History recovery.....	5
Real-time data collection.....	6
Performance considerations.....	7
Data type conversion.....	8
Error handling.....	8
Failover for the PI to PI Interface.....	8
Tracking interface status.....	9
<b>Installing and configuring the PI to PI Interface.....</b>	<b>11</b>
Installing the PI to PI Interface.....	11
Configuring the PI to PI Interface.....	12
Create trusts on the source and target PI Data Archives.....	12
Create and configure the interface instance.....	13
Configure the Windows service.....	13
Enable buffering.....	14
<b>Configuring PI points for the PI to PI Interface.....</b>	<b>15</b>
PI point attributes.....	15
Tag (PI point name).....	15
PointSource (point source).....	16
PointType (data type).....	16
Location1 (interface instance).....	16
Location2 (time stamp adjustment).....	16
Location3 (interface status events).....	17
Location4 (scan class).....	18
Location5 (write mode).....	18
InstrumentTag (source point name).....	19
UserInt1 (source point ID).....	19
Scan.....	19
Shutdown.....	19
Mapping source points to target points.....	20
Preventing data mismatches.....	20
<b>Features supported by the PI to PI Interface.....</b>	<b>23</b>
Additional details on interface features.....	23
<b>Installation checklist for the PI to PI Interface.....</b>	<b>27</b>
Configure data collection.....	27
Configure interface diagnostics.....	28
Configure advanced interface features.....	28

- PI ICU reference for the PI to PI Interface..... 29**
  - Required tab..... 30
  - History Recovery tab..... 30
  - Debug tab..... 31
  - Location tab..... 31
  - Optional tab..... 31
  - Opt Cont tab..... 32
  - Source PI Server Failover tab..... 33
  
- Error and informational messages from the PI to PI Interface..... 35**
  
- Command-line parameters for the PI to PI Interface..... 37**
  - General interface operation parameters..... 37
  - History recovery and archive data collection parameters..... 39
  - Exception data collection parameters..... 41
  - Point attribute override parameters..... 41
  - Source Data Archive-level failover parameters..... 42
  - Sample startup and configuration files..... 43
    - Sample PltoPI batch file..... 43
    - Sample PltoPI configuration file..... 44
  
- Technical support and other resources..... 45**

# Introduction to the PI to PI Interface

---

The PI to PI Interface copies PI point data from one PI Data Archive to another. Data is moved in one direction, meaning data is copied from the source to the target PI Data Archive. Typical applications for the interface include the following:

- Isolate users from the source PI Data Archive, for security, load distribution, or to resolve remote access issues.
- Maintain a copy of archive data on a secondary PI Data Archive, perhaps as a backup.
- Create a test copy of a production PI Data Archive.
- Aggregate data from many PI Data Archives to a single PI Data Archive.



**Note:**

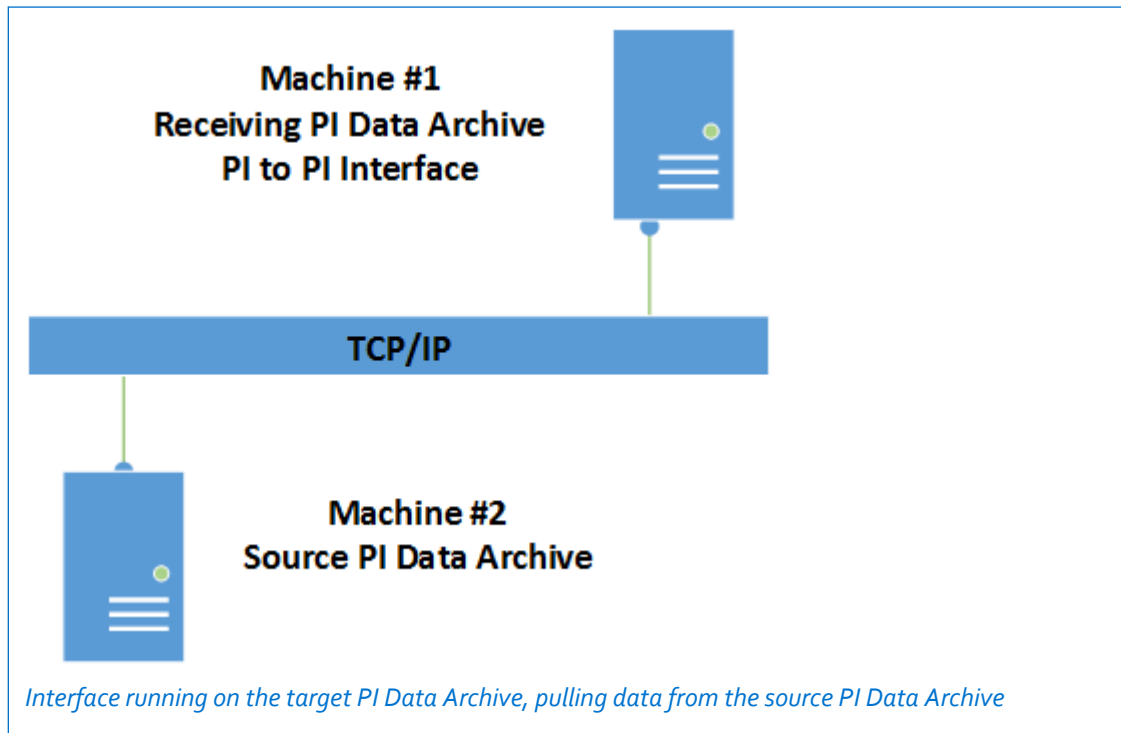
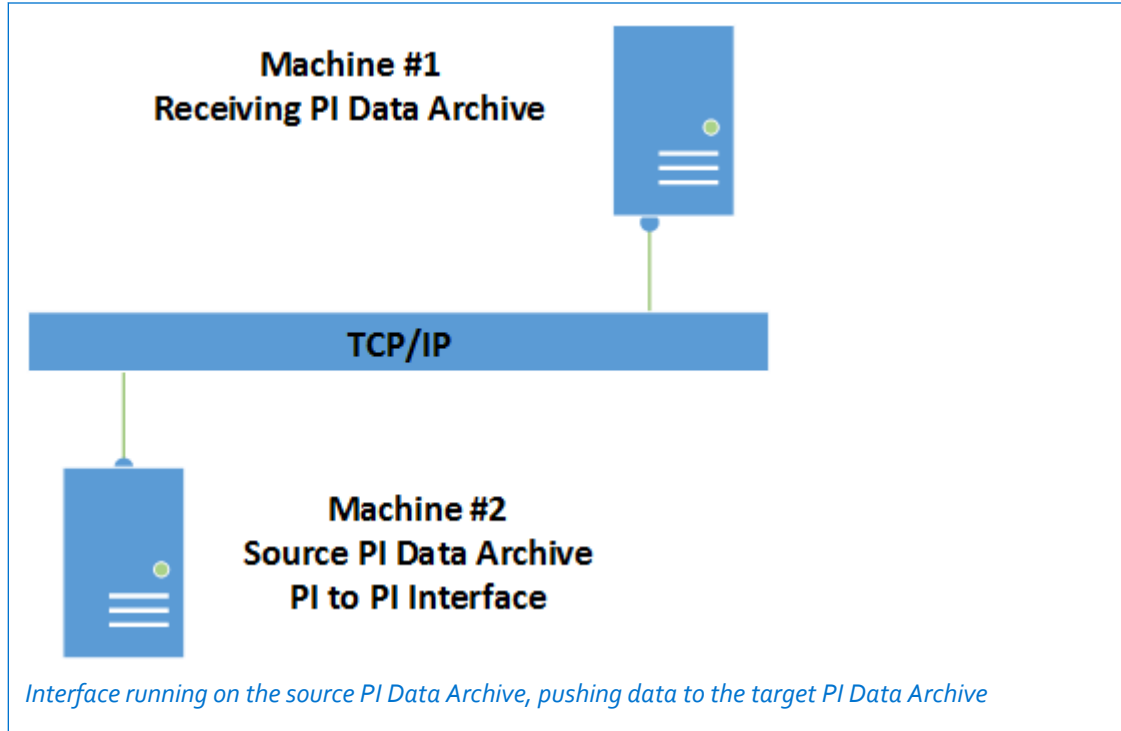
You can also use the interface for data replication in PI collectives, as an alternative to fanning data using buffering. Use this method only if it is required for your system configuration or security needs. For details about high-availability strategies, refer to the *High Availability Administration Guide*.

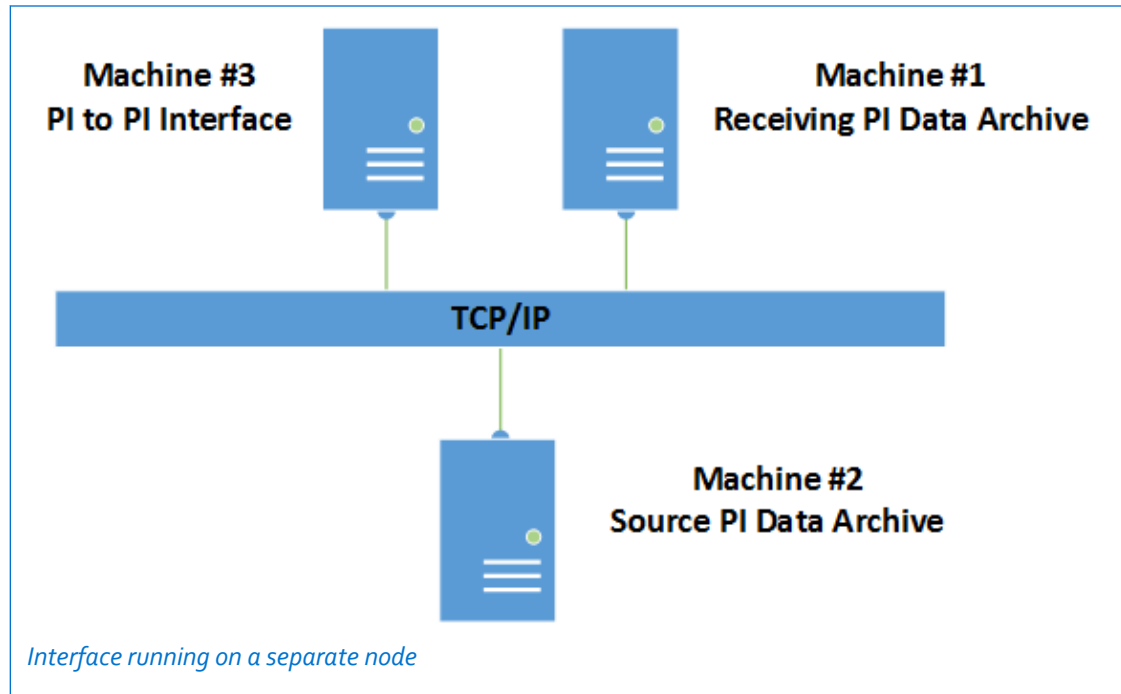
Interface PI points are created on the target PI Data Archive. Each interface point is configured to receive data for a unique source point. PI points receive archive or exception data from the source point. Exception data is data that has not yet been subjected to compression. By default, all points belonging to the scan class 1 receive exception data. PI points assigned to any other scan class receive archive data.

The interface supports history recovery, which recovers data for time periods when the interface was not running or unable to collect data. The history recovery period is configurable, and the default is eight hours.

Both source PI Data Archive-level failover and UniInt phase 2 interface-level failover are supported. When running in source PI Data Archive-level failover mode, the interface obtains data from one of two available source PI Data Archives. To ensure that the interface obtains the same data regardless of which source server is active, the source PI Data Archives must have identical point definitions and data streams for all interface source points. In UniInt failover mode, two copies of the interface are connected to the source PI Data Archive at the same time. If the primary interface stops collecting data, the backup interface assumes the primary role and continues data collection. Failover maximizes interface data availability on the target PI Data Archive(s). Source PI Data Archive-level failover and UniInt Phase 2 interface level failover modes can be run simultaneously.

The following diagrams illustrate three basic hardware configurations for configuring the interface.





#### Topics in this section

- [Related manuals](#)
- [Supported operating systems](#)
- [Interface limitations](#)

## Related manuals

- [PI Server manuals](#)
- [PI API Installation Manual](#)
- [PI Interface Configuration Utility User Manual](#)
- [UniInt Interface User's Guide](#)
- [PI Buffering User's Guide](#)

## Supported operating systems

Platforms: (32-bit or 64-bit in emulation mode)

- Windows 2003 Server
- Windows Vista
- Windows 2008 and Windows 2008 R2 Server
- Windows 7

- Windows 8
- Windows 2012 Server

No 64-bit builds of the interface are available.

## Interface limitations

Before deploying the interface, be sure it is suitable for your intended purpose. Note the following:

- **Not a true data replication tool**

The interface does not synchronize PI Data Archive data or validate data. It simply copies data from one PI Data Archive to another in an incremental, time-forward manner. The interface does not guarantee that the data in the source and target PI Data Archives matches exactly. If the goal is replication, use n-way buffering, which is supported with PI API v1.6.x and later. For details, see the PI API installation manual.

- **Potential data gaps**

The PI Archive Subsystem might temporarily queue data in memory prior to committing it to disk. This approach can lead to data gaps when using the interface for real-time data collection with history recovery enabled. To avoid data gaps, run the interface in history recovery mode without snapshot updates. Current real-time data from the source PI Data Archive will not be available on the target PI Data Archive.

- **No support for annotations or batch data**

The interface is a PI API based application. It does not currently support PI point annotations, which require the PI SDK. For this reason, it cannot be used to copy batch database data between PI Data Archives.

- **Performance dependencies**

The interface is a single threaded process. Its performance depends on the responsiveness of the source and target PI Data Archives and network quality. To monitor performance, use OSIsoft's PI Interface for Performance Monitor and PI Interface for Ping.

- **PI collectives**

The interface cannot be configured to communicate with a PI collective, only with one of the PI Data Archives in the collective.

History recovery can be compromised when the interface is configured for n-way buffering to a collective. If the last value for each target point is not the same in all PI Data Archives in the collective, data gaps or overlaps can result. To avoid this problem, initialize all PI Data Archives in the target collective with the history for the interface point list prior to implementing the PI to PI Interface.

- **Buffering**

The PI Buffer Subsystem v3.4.375.38 supports writing to the archive only in NO REPLACE mode, which prevents the interface from honoring any archive write options configured in the **Location5** attribute of target points, which can lead to data loss. To avoid this issue, use a later version of PI Buffer Subsystem or PI Bufserv.



# How the PI to PI Interface works

---

You can configure the interface to copy exception (snapshot) or archive data. For best performance, configure a separate instance of the interface for each source PI Data Archive. If you intend to copy both archive and exception data, configure separate instances of the interface for each type of data. Be advised that data throughput (events per second) can be limited by network quality or system resources.

## Topics in this section

- [Interface startup](#)
- [History recovery](#)
- [Real-time data collection](#)
- [Performance considerations](#)
- [Data type conversion](#)
- [Error handling](#)
- [Failover for the PI to PI Interface](#)
- [Tracking interface status](#)

## Interface startup

On startup, the interface reads site-specific operating parameters such as source and target PI Data Archive, data update rates, and history recovery settings, from its configuration file. Next, the interface connects to the source and target PI Data Archives. Each connection to a PI Data Archive is associated with a specific PI user. On the source server, this user must have permission to read PI point attributes and data. On the target server, the user must have read access for point attributes and read and write data access for its points.

Next, the interface builds a list of target PI points, grouping points internally by scan class. Each target point must have a source point. If an invalid point configuration is detected, the point is either rejected or added to the point list and marked as erroneous, and the interface logs the error.

After the interface builds its point lists, it calculates the time offset between PI Data Archives. Each participating node must be configured with the correct system time for its time zone. During operation, the offset is checked every two minutes. The offset is used for data time stamp adjustments (if enabled) and to time stamp interface status events.

Finally, the server performs history recovery and begins real-time data collection.

## History recovery

History recovery enables you to recover archive data for time periods when the interface was not running or otherwise unable to collect data. During history recovery, the interface updates target PI points with values from the source archive, according to the settings you configure. The interface performs history recovery after the following events:

- During interface startup (before starting real-time data collection)
- After restoring a lost PI Data Archive connection
- Update queue overflow error recovery
- For each new target point added to the interface

You can specify the recovery period, time increments within the total recovery period, a pause time between increments, and the maximum number of archive events returned per data request. These parameters enable you to manage the performance impact on the source and target PI Data Archives by throttling the data collection rate.

Time-range history recovery is configured through the interface startup configuration file, which you edit using PI ICU. When time-range recovery is enabled, the interface starts up, recovers archive data for the specified start and end time then exits. Specify the recovery start and end times in the time zone where the interface runs, even if the source PI Data Archive is in a different time zone than the machine where the interface runs. Ensure that the target archive has enough space for the data to be recovered.

The starting point for history recovery is determined on a point-by-point basis. If the current value for a point is more recent than the starting point of the recovery period, history recovery begins at the point's current value, to prevent data overlap, streamline data retrieval, and prevent out-of-order data. For newly-created points with no archive data, the interface performs history recovery for the total recovery period.

History recovery is performed independently for each scan class. Source data is written to the target archive in one of three modes: append, replace or no replace. This setting is configured on a point-by-point basis. The interface logs the progress of recovery.

By default, during history recovery, the interface also collects real-time exception data every five seconds. To configure this time interval using PI ICU, go to the **Optional** tab and select the check box **Set time interval between clearing exception queue during history recovery**. As it nears the end of its point list, more points are in the update list and the time to collect exception updates increases. This extra overhead can significantly increase the time required to complete history recovery.

By default, history recovery bypasses compression testing on the target server when its version is PI Data Archive 3.3 or later. Target PI Data Archives that are pre-3.3 always compression-test incoming data.

## Real-time data collection

During real-time data collection, the interface reads events from the source PI Data Archive and sends them to the target server. There are two options for determining where the source data comes from:

- **Archive data**

This data has passed exception testing by whatever interface accepted the data, and has passed compression testing on the source server. However, reading data from the PI Data Archive increases the workload on the source server, because the data is read from disk.

- **Exception data**

This data has passed exception testing by the interface that accepted it, but has not passed compression testing by the source server. Reading exception data imposes less workload on

the source server, because the data resides in memory. To ensure the closest correspondence between values on the source and target servers, configure identical compression settings on the source and target servers. To configure a PI point for exception data collection, assign it to scan class 1.

**Note:**

If you need to perform both archive and exception data collection, configure separate instances of the interface for each purpose, to ensure optimal performance.

The source PI Data Archive provides a time stamp for each data event. The interface might adjust time stamps to account for time zone differences and clock drift between PI Data Archives. Each PI Data Archive must be configured with the correct time for its time zone. For individual points, you can configure time stamp adjustment by setting the **Location2** point attribute.

Target point definitions can be modified while the interface is running. The interface checks with the target PI Data Archive for point configuration updates, by default every two minutes. These updates include adding, editing, and removing points. Processing point updates is a low priority for the interface. To prevent update processing from delaying data collection, the interface processes a maximum of 25 point updates at a time. If there are more than 25 point updates, the interface checks every 30 seconds until all updates have been processed. If a large number of point edits are made, consider stopping and restarting the interface, to force it to process all edits by rebuilding its point list.

## Performance considerations

### Minimize network latency

To minimize the effect of network latency on data throughput, run the PI to PI Interface on the source PI Data Archive computer with buffering enabled.

### Maximize archive cache reads

PI Data Archives version 3.4 and higher cache a small amount of recent archive data in memory (by default, four events). The size of this server cache is configurable, and you can adjust the frequency of scan classes so that scans occur frequently enough to maximize the reading of archive data from the cache rather than from disk.

### Reading archive data versus exception data

You can configure the interface to collect data from the source server's archive or from its snapshots (exception data). Reading snapshots is faster, because data is read from memory rather than disk. However, exception data has not been compressed. To read a point's exception data, assign the point to scan class 1.

### Managing memory consumption

The PI Data Archive maintains update manager queues that store events in memory for client applications such as PI Interfaces. The server creates separate queues for each client application that requests updates. The queues grow between scans as exceptions accumulate on the source PI Data Archive. To control the amount of server memory consumed by update queues, you can configure a maximum size for individual queues and a maximum for the total amount of memory consumed by all update queues. If a queue reaches its maximum size, the

server stops queuing updates for the interface, which causes the interface to delete its point list, log errors, and enter history recovery to reinitialize data collection.

To configure these limits, set **MaxUpdateQueue** and **TotalUpdateQueue** using PI System Management Tools or the **piconfig** utility. Default maximums for queue size are as follows (specified as exception events per client application):

- PI Data Archive 3.4.375.38 (PR1) and later: 50000
- PI Data Archive 3.3 to PI Data Archive 3.4.370: 4096
- PI Data Archive 3.2: 2000

To calculate the minimum queue size required, use the following formula:

$$\text{Minimum Limit} = \text{Interface Point Count} * \text{Scan Rate (Sec)}$$

For example, if the interface is maintaining 10,000 target points with a scan rate of 10 seconds, set the pending update limit and global update limit to at least 100,000.

You can also manage the workload by careful configuration of scan rates for target points. Choose a small initial scan rate (for example 10 seconds), then use a performance point to determine the average time required to complete the exception data scan and adjust the scan rate accordingly.

## Data type conversion

If the data types of the source and target PI points differ, the interface can convert source data to compatible target data types as follows:

Source point data type	Compatible target point data type
Float 16/32/64	Int16/32, Digital
Int16/32	Float, Digital
Digital	Int16/32
String	Digital

## Error handling

When the interface encounters an error, it logs a message to the log file. If an error is not PI point-specific, the interface determines whether it is still connected to the PI Data Archive. If the error is PI point-specific, the point is then marked by the interface as erroneous and is removed from the update list. The interface attempts to read erroneous points every ten minutes. If an attempt succeeds, the point is restored to active data collection.

## Failover for the PI to PI Interface

### Source PI Data Archive-level failover support

Source PI Data Archive-level failover maximizes interface data availability on the target PI Data Archive(s). It enables the interface to obtain data from one of two source PI Data Archives. Each source server must have identical PI point definitions and data streams for interface source points.

The interface initiates failover if the active source data becomes stale or is not available due to network issues. The PI Interface Status utility is required to monitor data quality for each source PI Data Archive. The interface uses the utility status point output to verify that source data has not become stale.

### UniInt interface failover support

UniInt Phase 2 Failover provides support for cold, warm, or hot failover configurations. Phase 2 hot failover results in a no data loss solution for bi-directional data transfer between the PI Data Archive and the data source, given a single point of failure in the system architecture similar to Phase 1. However, in warm and cold failover configurations, you can expect a small period of data loss during a single point of failure transition. This failover solution requires two copies of the interface to be installed on different interface nodes collecting data simultaneously from a single data source. Phase 2 Failover requires each interface to have access to a shared data file. Failover operation is automatic and operates with no user interaction. Each interface participating in failover has the ability to monitor and determine liveness and failover status. To assist in administering system operations, the ability to manually trigger failover to a desired interface is also supported by the failover scheme.

The failover scheme is described in detail in the *UniInt Interface User Manual*, which is a supplement to this manual.

## Tracking interface status

To track interface status, you can create the following digital PI points on the target PI Data Archive:

- **Internal state PI point**

Indicates whether the interface is in startup, performing history recovery, scanning for data, experiencing network communication errors, denied data or point access (security permissions) or performing shutdown operations. To enable this point using PI ICU, go to the **PItoPI > Debug** tab, check **Interface Status Tag on Receiving PI Server**, and specify the name of the PI point.

- **Failover status PI point**

If you enable source server-level failover, you can configure a PI point that tracks failover status. To configure the point using PI ICU, go to the **PItoPI > Source PI Server Failover** tab, check **Enable failover status logging**, and specify the name of the PI point.



# Installing and configuring the PI to PI Interface

---

## Installing the PI to PI Interface

This section provides detailed instructions for installing and configuring the interface. A minimum functional configuration is described, which you can adjust according to your requirements. For details about features that are common to all UniInt interfaces, refer to the *UniInt Interface User Manual*.

Before installing and configuring, ensure that the following prerequisites are met:

- Verify that the source and target PI Data Archives are up and running.
- Verify that the interface node time zone is set correctly.
- On the interface node, install the following:
  - OSIssoft Prerequisites
  - PI Interface Configuration Utility (PI ICU)
  - PI API version 1.6.1.5 or greater, which is installed with the interface.

For optimal security, run the interface on the same computer as the most sensitive source PI Data Archive. This configuration does not require you to create an opening in the firewall that protects the source server.

PI points on the target PI Data Archive must be configured to permit the interface to read and write data and attributes. Points on the source PI Data Archive must be configured to permit the interface to read data and attributes. The **DataAccess** and **PtAccess** point attributes control access to the point data and attributes.

Buffering is temporary storage of the data that the interface collects and forwards to the target PI Data Archive. To ensure that you do not lose any data if the interface loses its connection with the target PI Data Archive, enable buffering. There are two buffering applications: the PI Buffer Subsystem (PIBufss) and the PI API Buffer Server (BufServ). PIBufss and BufServ are mutually exclusive; that is, on a particular computer, you can run only one at a time. For details about configuring buffering, refer to the *PI Buffering User Guide*.

To install the interface, download its setup kit from the OSIssoft downloads web page, and run the kit. By default, the interface is installed in the following location:

`%PIHOME% \Interfaces\PItoPI\`



**Note:**

The `%PIHOME%` directory, which is the root directory under which OSIssoft products are installed, is defined by the `PIHOME` entry in the `pipc.ini` configuration file in the `%windir%` directory. To override the default locations, edit the `pipc.ini` configuration file. Reserve the `C:` drive for the operating system and install the interface on another drive.

## Configuring the PI to PI Interface

Use the information in this section to perform the necessary tasks to configure the interface.

### Procedure

1. [Create trusts on the source and target PI Data Archives.](#)
2. [Create and configure the interface instance.](#)
3. [Configure the Windows service.](#)
4. [Enable buffering.](#)

## Create trusts on the source and target PI Data Archives

When creating trusts, you have many options. Following is a simple and secure approach, creating a trust for the following applications:

- PI to PI Interface (define on both the source and target servers)
- PI Interface Configuration Utility (ICU)
- Buffering

To create these trusts using PI System Management Tools, connect to the PI Data Archive and perform the following steps:

### Procedure

1. Click **Security** and choose **Mappings & Trusts**.
2. On the **Trusts** tab, right-click and choose **New Trust**.  
The Add Trust wizard appears.
3. Specify a meaningful name and description.
4. Configure settings as follows:

Trust	Type	Application Name	Network Path	PI User
PI to PI Interface	PI-API application	PIToE	Name of the interface node or IP address plus netmask 255.255.255.255	PI points' data owner
PI ICU	PI-API application	PI-ICU.exe	Name of the interface node or IP address plus netmask 255.255.255.255	Dedicated PI identity with the precise permissions required (database read access, read ptsecurity and read-write permission for OPC points)



Trust	Type	Application Name	Network Path	PI User
Buffering	PI-API application	BufServ: "APIBE" PIBufss: Pibufss.exe	Name of the interface node or IP address, plus netmask 255.255.255.255	PI points' data owner

## Create and configure the interface instance

To create an instance of the interface, launch PI ICU and perform the following steps:


### Procedure

1. Choose **Interface > New from BAT file**.
2. Browse to the directory where the interface is installed (default is %PIPC%\Interfaces\PItoPI, select PItoPI.bat\_new and click **Open**. The Select PI Host Server window appears.
3. Specify the target PI Data Archive and click **OK**. ICU displays the settings of the new instance of the interface.
4. Click on the **General** tab and edit the settings as follows.
  - a. **Point source**: Configure the point sources for the source interfaces that you want to duplicate.
  - b. **Interface ID**: Enter 1 or a numeric ID not already in use.
  - c. **Scan class**: Set to desired scan frequencies. (Scan class 1 is reserved for exception data collection.)  
Note that, when defining scan classes, you can spread the server workload using offsets.
  - d. Click **Apply**.
5. Click on the **PItoPI** tab.
  - a. Configure the source PI Data Archive (the server from which the interface is to read point data).
  - b. Click **Apply**.
6. Use the message log to verify the configuration:
  - a. To display the message log, launch PI SMT and choose **Operation > Message Logs**.
  - b. Using PI ICU, start the interface interactively by choosing **Interface > Start Interactive** (or type Ctrl-T).
  - c. Watch the message log for messages that indicate success or errors.

## Configure the Windows service

Launch PI ICU and perform the following steps:

### Procedure


1. Click **Service**.
2. To ensure that buffering is running when the interface starts, click **bufsrv** (for PI Data Archive (PI Server) before 3.4) or **pibufss** (for PI Data Archive (PI Server) 3.4 and later) in the **Installed services** list, then click the left-arrow button to add it to the **Dependencies** list. If prompted, update service dependencies.
3. Set **Startup Type** to **Auto**.
4. Click **Create**.
5. To start the service, click .

### After you finish

To verify that the service is created and is running, use the Windows **Services** control panel.

## Enable buffering

### Procedure

1. In ICU, choose **Tools > Buffering**. The Buffering window appears.
2. Click **Enable buffering with PI Buffer Subsystem**.
3. To start the buffering service, click **PI Buffer Subsystem Service**, then click .

### After you finish

To verify that buffering starts successfully, check the message log for messages indicating that the buffering application connected to the PI Server. To verify that the configuration is working as intended, reboot the interface node and confirm that the interface service and the buffering application restart.

# Configuring PI points for the PI to PI Interface

The PI point (also called a PI tag) is a time-stamped record of a single set of measurements (for example, the temperature of tank 100 every five minutes). If you mis-configure tags, the interface cannot correctly transmit the data from the source server to the target server.

## Topics in this section

- [PI point attributes](#)
- [Mapping source points to target points](#)
- [Preventing data mismatches](#)

## PI point attributes

To define PI points (PI tags), you configure, at a minimum, the following information:

- PI point name
- Point source
- Data type
- Interface instance
- Point data type
- Scan class

Depending on the type of point you are creating, a few other settings might be required. The following sections describe basic point settings in more detail.



### Note:

For information about other point attributes that are relevant to data collection and archiving, see the *UniInt Interface User Manual* and PI Server documentation.

## Tag (PI point name)

When assigning names to PI points, follow these rules:

- The point name must be unique.
- The first character must be alphanumeric, underscore (`_`), or percent sign (`%`).
- Control characters such as linefeeds or tabs are illegal, as are the following characters:  
\* ' ? ; { } [ ] | \ ` ' "

The following table indicates the maximum length of the length attribute, which depends on the combination of PI API and PI Server that you have installed.

PI API	PI Server	Maximum Length
1.6.0.2 or higher	3.4.370.x or higher	1023
1.6.0.2 or higher	Below 3.4.370.x	255
Below 1.6.0.2	3.4.370.x or higher	255

PI API	PI Server	Maximum Length
Below 1.6.0.2	Below 3.4.370.x	255

If your PI Server is earlier than version 3.4.370.x or the PI API version is earlier than 1.6.0.2 and you want to create points with names that exceed 255 characters, you must enable the PI SDK.



**Note:**

If the source point name length exceeds 80 characters, you must use the **UserInt1** attribute for source point mapping, due to a limitation of the PI API.

### PointSource (point source)

**PointSource** is an identifier that associates a PI point with a PI interface instance, enabling the interface to query the PI Server for the points that it updates. This field is not case-sensitive. In the interface batch startup file, the point source is specified using the **/PS** command-line parameter.

The following point sources are reserved. Do not configure them for interface instances.

Point Source	Reserved By
T	Totalizer Subsystem
G and @	Alarm subsystem
R	Random interface
9	RampSoak interface
C	Performance equations subsystem

### PointType (data type)

The interface supports all PI point data types. To prevent data corruption due to type mismatches, ensure that the data types of source and target points are compatible (ideally, identical). For details about point data types, see the *PI System Management Guide*.

### Location1 (interface instance)

**Location1** specifies the instance of the interface to which the PI point belongs. The value of this attribute must match the ID configured for the interface instance. This setting plus **PointSource** identify the interface instance that writes to a particular point. (**/ID**)

### Location2 (time stamp adjustment)

The source PI Data Archive supplies a time stamp for each data event. You can configure the interface to adjust these time stamps to account for time differences between PI Data Archives. Each PI Data Archive must be configured with the correct system time for its configured time zone.

Valid settings are as follows:

Location2 Value	Adjust for Time Zone Differences	Do Not Adjust for Time Zone Differences (PI2 servers only)	Adjust for Clock Drift**	Truncate Sub-second Time stamps***
0	Yes	--	--	--
1	--	Yes	--	--
2	Yes	--	Yes	--
3	--	Yes	Yes	--
4	Yes	--	--	Yes
5	--	Yes	--	Yes
6	Yes	--	Yes	Yes
7	--	Yes	Yes	Yes

\*\* Target PI Data Archive is the time master. Time stamps are adjusted to target PI Data Archive time. An offset of 30 minutes or less is considered clock drift.

\*\*\*Truncating sub-second time stamps can lead to data loss during history recovery and archive data scanning if multiple events are stored within the same second. Use with caution.

### Location3 (interface status events)

**Location3** configures which interface status events are to be written to a point. If a PI point is configured for archive data collection, this attribute also specifies whether the snapshot value is included with each scan update. Note that IO Timeout events result in a data gap for periods when the interface is disconnected from the source PI Data Archive.

Interface status events are configured on a point-by-point basis through the Location3 point attribute. Interface shutdown events are enabled through the interface startup script. By default, shutdown events are not written by the interface.

Location3 Value	Write I/O Timeout	Write Access Denied	Include Snapshot	Point-level Debugging
0	Yes	Yes		
1		Yes		
2	Yes			
3				
4	Yes	Yes	Yes	
5		Yes	Yes	
6	Yes		Yes	
7			Yes	
8	Yes	Yes		Yes
9		Yes		Yes
10	Yes			Yes
11				Yes
12	Yes	Yes	Yes	Yes
13		Yes	Yes	Yes
14	Yes		Yes	Yes

Location3 Value	Write I/O Timeout	Write Access Denied	Include Snapshot	Point-level Debugging
15			Yes	Yes



**Note:**

Set **Location3** to ensure that the following status events are not enabled:

- I/O Timeout status events are triggered when the interface loses a PI Data Archive connection. This status indicates that data is not being collected. An event is written to indicate the time of disconnection and another event is written to indicate the time of re-connection.
- Access Denied status events are written whenever the interface is denied access to a point's data or attribute information.
- Intf Shut status events are enabled through **/stopstat** startup parameter. When enabled, an event is written when the interface is stopped and again on startup. These events indicate that no data is being collected because the interface is not running.

If written to target points, the preceding events create a data gap that cannot be filled using history recovery.

### Location4 (scan class)

**Location4** is used to configure the scan class for the PI point. The scan class determines the frequency at which input points are scanned for new values. **Location4** must be a positive number (or 0 for triggered and event-based points). Specify scan frequency and optional offset using the following format:

*HH:MM:SS.##, HH:MM:SS.##*

Examples:

*00:01:00,00:00:05*

or, equivalently:

*60,5*

If you omit *HH* and *MM*, the scan period is assumed to be in seconds. Subsecond scans are specified as hundredths of a second (.01 to .99).

To configure a time of day at which a single scan is performed, specify the time using a 24-hour clock and append an "L" following the time, as follows: *HH:MM:SS.##,offset,L*. For example:

*12:00:00,0,L*

### Location5 (write mode)

Specifies the write mode for sending archive data to the target PI Data Archive.

The following table lists the supported modes.

Locations	Write Mode	Description
0	ARCAPPEND	Archive and snapshot events. Add archive event regardless of existing events. Default snapshot behavior is to append data if event at time stamp exists.
2	ARCREPLACE	Archive events only. Add archive event, replace if event at same time. Default snapshot behavior is to append data if event at time stamp exists.
4	ARCREPLACE	Archive and snapshot events. Add archive or snapshot event, replace if event at same time.

### InstrumentTag (source point name)

In target PI points, the **InstrumentTag** attribute specifies the name of the source point. Alternately, the source point can be specified using the **ExDesc** or **UserInt1** attributes. If you do not configure a source point, the source and target point names must be identical. To configure the mapping method used by the interface using PI ICU, go to the **PItoPI > Optional** tab, select the **Source tag definition** attribute, and enable the desired option.

### UserInt1 (source point ID)

As an alternative to mapping PI point names, you can specify the unique PointID of the source point in this attribute of the target point. The advantage of this approach is that the mapping remains valid if the source point is renamed. Due to a limitation of the PI API, if the source point name is long than 80 characters, you must use this approach to mapping.

### Scan

This attribute enables or disables data collection for the PI point. By default, data collection is enabled (**Scan** is set to 1). To disable data collection, set **Scan** to 0. If the **Scan** attribute is 0 when the interface starts, the interface does not load or update the point. If you enable scanning while the interface is running, the time required for data collection to start depends on how many points you enable, because they are processed in batches. For efficiency, if you need to enable scanning for a large number of points, stop and restart the interface. If a point that is loaded by the interface is subsequently edited so that the point is no longer valid, the point is removed from the interface and **SCAN OFF** is written to the point.

### Shutdown

By default, the PI Shutdown subsystem writes the SHUTDOWN digital state to all PI points when PI Server is started. The time stamp that is used for the SHUTDOWN events is retrieved from a file that is updated by the snapshot subsystem. The time stamp is usually updated every 15 minutes, which means that the time stamp for the SHUTDOWN events is accurate to within 15

minutes in the event of a power failure. For additional information on shutdown events, refer to PI Server manuals.



**Note:**

The SHUTDOWN events that are written by the PI shutdown subsystem are independent of the SHUTDOWN events that are written by the interface.

To prevent SHUTDOWN events from being written when PI Server is restarted, set the **Shutdown** attribute to 0. To configure the PI shutdown subsystem to write SHUTDOWN events only for PI points that have their **Shutdown** attribute set to 1, edit the `\\PI\dat\Shutdown.dat` file, as described in PI buffering documentation.

## Mapping source points to target points

To load the target server with points from the source server, you can use OSIsoft’s Tag Configurator plug-in for Microsoft Excel to import the points from the source PI Data Archive, make any changes you need, then export them to the target PI Data Archive.

When the interface loads a target point, it must identify the source point from which data is to be collected. If the name of the source and target points are identical, no explicit mapping is required. If the names differ, you must explicitly map the source points to the target points. There are three options for mapping source points to target points:

- Set the target point’s **InstrumentTag** attribute to the name of the source point.
- Specify the name of the source point in the target point’s **ExDesc** attribute, using the **/STAG** flag.
- Set the target point’s **UserInt1** attribute to the source point’s PointID (the unique numeric identifier of the source point, displayed on the **System** tab of PI SMT’s Point Builder tool). This approach ensures that the mapping remains valid if the source point is renamed.

To configure the method that the interface uses to map points using PI ICU, go to the **PItoPI > Optional** tab, select the **Source tag definition attribute** check box, and choose the desired method.



**Note:**

Points in the source server might be added, changed or deleted. To replicate such changes to the target server automatically, you can use OSIsoft’s PI AutoPointSync (APS) with the PI to PI APS Connector. For details, consult the user manuals for PI APS and the PI to PI APS Connector.

## Preventing data mismatches

Unlike interfaces that receive unprocessed data, the data that is read by the PI to PI Interface has already passed exception handling and compression. To prevent mismatches between the data in source and target PI points, disable interface exception filtering for the points on the target server.

To ensure that no values read from the source server are filtered out by the target, configure target points as follows:

Point Attribute	Value
CompDev	Same value as source PI point



Point Attribute	Value
CompDevPercent	Source Tag Value
CompMax	32767
CompMin	0
ExcDev	0
ExcDevPercent	0
ExcMax	32767 or higher
ExcMin	0

**Note:**

If a point configured for archive data updates has its **Location3** attribute set so that it receives snapshot updates, data compression must be enabled to prevent data mismatches between PI Data Archives.

The **Zero** and **Span** point attributes must be identical in the source and target points. For Float16 points, the **Zero** and **Span** attributes must be set, because they affect how the data is stored in the archive. For digital PI points, do not set the **Zero** and **Span** attributes, because they are set by the PI Data Archive when a digital point is created.



## Features supported by the PI to PI Interface

The following table shows the features that the interface supports.

Feature	Support
Auto creates PI points	APS Connector
Point Builder utility	No
ICU Control	Yes
PI point types	All data types are supported
Sub-second time stamps	Yes
Sub-second scan classes	Yes
Automatically incorporates PI point attribute changes	Yes
Exception reporting	Yes
Outputs from PI Data Archive	No
Inputs to PI Data Archive: Scan-based / Unsolicited / Event Tags	Scan-based only
Supports questionable bit	No
Supports multi-character PointSource	Yes
Maximum point count	Unlimited
Uses PI SDK *	No
PINet string support	No
Source of time stamps *	Source PI Data Archive
History recovery	Yes
Disconnected Startup *	Yes
SetDeviceStatus *	Yes
Failover	Source PI Data Archive-Level UniInt Phase 2 Interface Level (Warm)
Vendor Software Required on PI Interface Node / PINet Node *	No
Vendor Software Required on Foreign Device	No
Vendor Hardware Required	No
Additional PI Software Included with Interface	No
Device Point Types	Real, Integer and Digital.
Serial-based Interface	No

## Additional details on interface features

### APS Connector

AutoPointSync is an OSIsoft product that keeps the point configuration of PI Data Archives in sync with each other. The PI to PI APS Connector is required to use APS with this interface.



**Note:**

The PI to PI APS Connector contains a separate implementation of the procedure to identify the source point for an interface point. If an interface instance is registered with APS, consult the PI to PI APS Connector manual to confirm that the PI to PI APS Connector version implements the same procedure as the interface, which is required to ensure that the PI to PI APS Connector synchronizes with the same source point as the interface.

### Supports questionable bit

The interface copies questionable bit data for a give source point from one PI Data Archive to another.

### Uses PI SDK

The PI SDK and the PI API are bundled together and must be installed on each PI Interface node. This interface does not require the PI SDK.

### Maximum point count

The interface does not enforce any hard-coded maximum for the number of points it can maintain, but it is a single-threaded process and its performance is affected by hardware, network conditions, and workload. For details about monitoring interface performance, refer to the *UniInt Interface User Manual*.

### Source of time stamps

The source PI Data Archive provides a time stamp for each data event. Time stamps are automatically adjusted to account for time zone differences. The interface can also adjust time stamps for clock drift, which is the time offset between PI Data Archives after accounting for time zone differences. An offset of 30 minutes or less is considered clock drift. Adjusting for clock drift means the time offset is added to the source time stamp, adjusting it to target PI Data Archive time. Time stamp adjustment for individual points is configured using the Location2 point attribute. Interface nodes and the source and target PI Data Archives must have the correct system time for their time zone.

### History recovery

History recovery enables you to recover archive data for time periods when the interface was not running or otherwise unable to collect data. History recovery is performed on startup, after restoring a lost PI Data Archive connection and after a disruption in exception data collection. In addition, when a new point is added to the interface, history recovery is performed on that point. The history recovery period is configurable. The default is to recover data for the previous 8 hours. To disable history recovery, set the time period to 0. You can also recover data for a specified time period. If you specify a start and end time, the interface recovers data for the specified period, then exits.

### UniInt-based

UniInt stands for Universal Interface, the framework on which this interface is based. The framework speeds interface development and provides a common set of basic features. The *UniInt Interface User Manual* is a supplement to this manual.

### Disconnected startup

The PI to PI interface is built with a version of UniInt that supports disconnected start-up. Disconnected start-up is the ability to start the interface without a connection to the PI Data Archive. This functionality is enabled by adding **/cachemode** to the list of start-up parameters or by enabling disconnected startup using the ICU. Refer to the *UniInt Interface User Manual* for more details on UniInt disconnected startup. The PI to PI Interface supports disconnected start-up for both source and target PI Data Archives.

### SetDeviceStatus

The health point with the point attribute **ExDesc = [UI\_DEVSTAT]** represents the status of the source device. The following events can be written into this point:

- 1 | Starting: The interface is starting.
- Good: The interface is properly communicating and reading data from the server.

The following events indicate a failure to communicate with the server:

- 3 | 1 device(s) in error | Network communication error to source PI server
- 3 | 1 device(s) in error | Unable to get archive data from source PI server
- 3 | 1 device(s) in error | Unable to get snapshot data from source PI server
- 3 | 1 device(s) in error | Unable to write data to target PI server
- 3 | 1 device(s) in error | Unable to obtain current data with source PI server failover enabled
- 4 | Intf Shutdown: The interface is stopped.

Refer to the *UniInt Interface User Manual* for more information on how to configure health points.

### Source PI Data Archive-level failover support

Source PI Data Archive-level failover maximizes interface data availability on the target PI Data Archive(s). It enables the interface to obtain data from one of two source PI Data Archives. Each source server must have identical point definitions and data streams for interface source points.

The interface initiates failover if the active source data becomes stale or is not available due to network issues. The PI Interface Status Utility is required to monitor data quality for each source PI Data Archive. The interface uses the utility status point output to verify that source data has not become stale.

### UniInt interface failover support

UniInt Phase 2 Failover provides support for cold, warm, or hot failover configurations. Phase 2 hot failover results in a no data loss solution for bi-directional data transfer between the PI Data Archive and the data source, given a single point of failure in the system architecture similar to Phase 1. However, in warm and cold failover configurations, you can expect a small period of data loss during a single point of failure transition. This failover solution requires two copies of the interface to be installed on different interface nodes collecting data

simultaneously from a single data source. Phase 2 Failover requires each interface to have access to a shared data file. Failover operation is automatic and operates with no user interaction. Each interface participating in failover has the ability to monitor and determine liveness and failover status. To assist in administering system operations, the ability to manually trigger failover to a desired interface is also supported by the failover scheme.

The failover scheme is described in detail in the *UniInt Interface User Manual*, which is a supplement to this manual.

# Installation checklist for the PI to PI Interface

---

If you are not familiar with PI Interfaces, refer to the detailed installation procedure in this guide. If you are familiar with installing and configuring PI Interfaces, this checklist covers the basic steps required to get the interface running. [Configure data collection](#) is required. [Configure interface diagnostics](#) and [Configure advanced interface features](#) are optional.

## Topics in this section

- [Configure data collection](#)
- [Configure interface diagnostics](#)
- [Configure advanced interface features](#)

## Configure data collection

### Procedure

1. Confirm that you can connect to the target PI Data Archive using PI System Management Tools (SMT). You do not need to run PI SMT on the same computer as this interface.
2. If you are running the interface on an dedicated node, use PI SMT to define a trust that permits the interface to write data to the target PI Data Archive and a trust that permits the interface to read data from the source server.
3. Install PI Interface Configuration Utility (ICU) on the interface node.
4. Install the interface.
5. Verify that the system time and time zone on the interface node are set correctly.
6. Run the ICU and configure a new instance of this interface, specifying the point sources for all the interfaces for which you want to duplicate point data.
7. If the interface maintains digital points, ensure that the digital state sets on the source and target servers are identical.
8. Start the interface interactively and confirm its successful connection to the PI Data Archive without buffering.
9. Confirm that the interface collects data successfully.
10. Stop the interface and configure buffering. To optimize buffering throughput, choose the **Tools > Buffering > Buffering Settings** ICU menu item and set the **Primary and Secondary Memory Buffer Size (Bytes)** to 2000000.
11. Start the buffering application and the interface. Confirm that the interface works together with the buffering application by either physically removing the connection between the interface node and the PI Data Archive node or by stopping the PI Data Archive.
12. Configure the interface to run as an automatic Windows service. Confirm that the service starts properly.
13. Restart the interface node and confirm that the interface and the buffering application restart.

## Configure interface diagnostics

### Procedure

1. Configure scan class performance points.
2. Install the full version of the PI Performance Monitor interface on the interface node.
3. Configure Performance Counter points.
4. Configure UniInt Health Monitoring points.
5. Configure the I/O Rate point.
6. On the PI Data Archive node, install and configure the Interface Status Utility (ISU).
7. Configure the Interface Status point.

## Configure advanced interface features

### Procedure

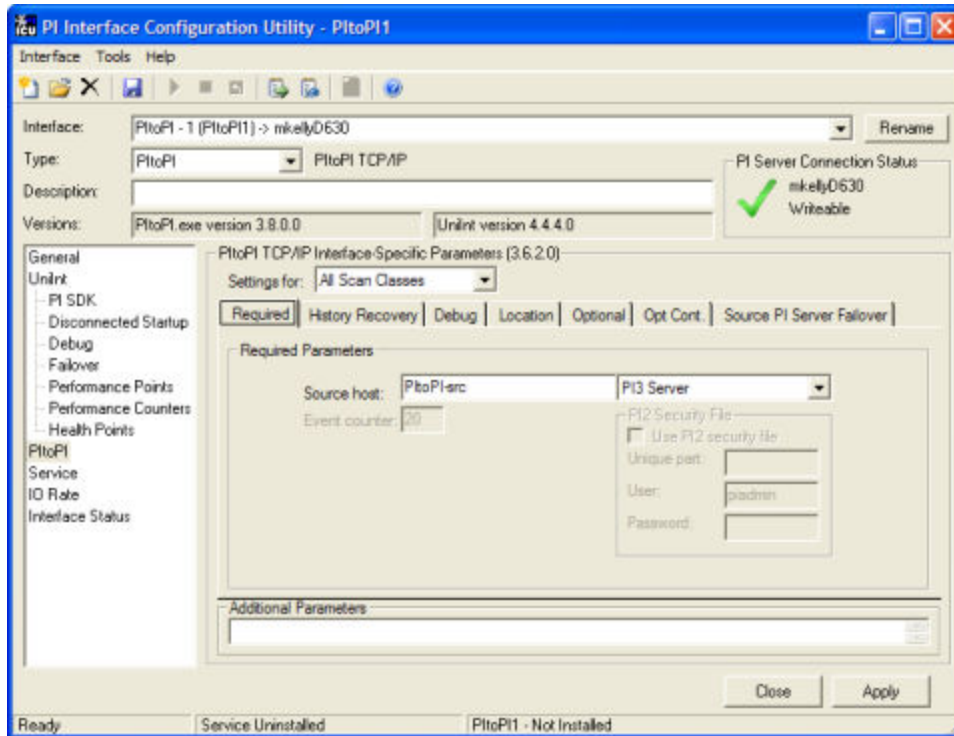
1. Configure the interface for disconnected startup. Refer to the *UniInt Interface User Manual* for more details on UniInt disconnected startup.
2. Configure UniInt failover; see [Features supported by the PI to PI Interface](#) in this document for details related to configuring the interface for failover.
3. To ensure uninterrupted data collection, configure a secondary source PI Data Archive. Using PI ICU, enable source server failover.



# PI ICU reference for the PI to PI Interface

The PI Interface Configuration Utility (PI ICU) provides a graphical user interface for configuring PI Interfaces. The interface is launched using a Windows batch (.bat) file, which invokes the interface executable, specifying run-time options as command-line parameters. To ensure a correctly-formatted startup batch file, use PI ICU to configure the interface, rather than manually editing the batch file.

To configure the interface-specific settings, go to the PtoPI page, as shown in the following figure.



The following sections describe the interface settings, noting the corresponding command-line parameter to help you read the resulting batch startup file for debugging purposes. For details about settings that are common to all interfaces, consult the *Unint Interface User Manual*.

## Topics in this section

- [Required tab](#)
- [History Recovery tab](#)
- [Debug tab](#)
- [Location tab](#)
- [Optional tab](#)
- [Opt Cont tab](#)
- [Source PI Server Failover tab](#)

## Required tab

- **Source host**

The name of the PI Server from which this interface instance reads data. (**/SRC\_HOST=hostname**). Select the server type (PI3 or PI2) from the drop-down list. For PI2 servers, specify security settings as required. For details about PI2 security, refer to the PI2 Server documentation.

- **Event counter**

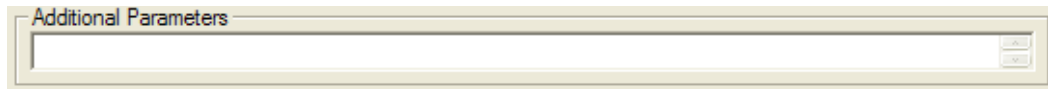
The event counter is a unique value used to correlate an I/O rate point specified in the `IORates.dat` file with this copy of the interface. (**/EC=x**).

I/O rate points are used to track interface performance. This field is enabled when a scan class is selected in the **Settings for** field.

For detailed information about event counters, refer to the *UniInt Interface User's Guide*.

- **Additional Parameters**

This field enables you to specify any settings for the interface that are not supported by PI ICU.



## History Recovery tab

The following settings configure how much history is recovered and enables you to tune performance of recovery.

- **Maximum hours of history to recover**

Number of hours to recover history for all points. Setting the value to 0 disables history recovery for all points. (**/RH=#**)

- **Hours of history to recover per cycle**

For load balancing, this setting enables you to configure the number of hours of history (and therefore, the size of the chunk) that the interface recovers each time it cycles through its point list. (**/RH\_INC=#**)

To recover all history in a single chunk, specify a value greater than the value configured for **Maximum hours of history to recover**. If this field is set to zero, the default is 24 hours. If this field is greater than **/RH**, it is set to **/RH** and the interface retrieves the whole history range in one chunk.

- **Millisecond pause between history calls**

Specifies the number of milliseconds to pause between history recovery calls, to reduce workload on the PI Data Archive. (**/HRPAUSE=milliseconds**)

- **Use history recovery only (no snapshot data collection)**

If this check box is selected, the interface does not read the source server snapshot. (*/HROONLY*)

For each scan class, history is recovered from the last snapshot value to the current time. This box must be checked if you want to specify a history time range.

- **Time Range History Recovery**

If **Use history recovery only** is enabled, specifies the time range of history to recover before exiting. (*/HROONLY=dd-mmm-yy:hh:mm:ss,dd-mmm-yy:hh:mm:ss*)

The times must be specified using PI time string formats. The following example recovers two hours of data and then exits:

```
10-dec-98:10:00:00,10-dec-98:12:00:00
```

If the source server resides in another time zone, specify the corresponding time in the interface node's time zone.

This setting overrides the normal check for the most recent snapshot in the target database, so out-of-order data might result. When time-range history recovery is enabled, any value specified for the **Maximum hours of history to recover** setting is ignored.

- **Start history recovery beginning with the first value prior to the start time**

Retrieve history for all points, starting from the value immediately prior to the specified history recovery start time. By default, recovery begins with the first value after start time. Requires **Use history recovery only** to be enabled.

## Debug tab

- **Debug Parameters**

Check the types of debug messages that you want logged. (*/DB=#,#,#,#...*)

- **Interface Status Tag on Target PI Server**

Specify the name of an interface status point that is configured on the target PI Data Archive server. (*/IST=tagname*)

## Location tab

The settings on this tab enable you to apply specific settings to the location codes of target PI points, overriding the value they contain in the source server. (*/C1* through */C5*)

## Optional tab

- **Apply tag's compression specifications to data retrieved during history recovery**

Use compression specifications in point configurations to send data retrieved during history recovery with compression. Usually data is retrieved from source server and sent to target server without compression during history recovery. (/DC)

- **Source tag definition attribute**

Specifies how the interface determines the source-to-target point mapping. By default, the interface seeks a target point with the same name (**Tag** attribute) as the source point.

- **Specify maximum events to retrieve for a single point in each call to get history**

For target servers that are version 3.3 or higher, sets the maximum number of events to retrieve for a single point with each attempt to retrieve history. For performance tuning, the optimal value is one that retrieves as much history as possible without consuming excessive memory. (/MH=x, default: 1000)

- **Specify maximum number of exception events retrieved per data request**

The maximum number of exception events retrieved per data request. A large count reduces the number of calls required for acquiring exception updates. A small count reduces the time to complete each request (for troubleshooting network timeout issues). (/ME=#, default: 5000)

- **Set time interval between clearing exception queue during history recovery**

Sets the time interval between clearing the exception queue on the source PI Data Archive for exception data scan classes. By default the interface collects exceptions from the source PI Data Archive every five seconds during history recovery to prevent overflowing the queue. To tune history recovery performance, adjust this time interval. (/RH\_QCCKECK=#, default: 5)

- **Specify the freq. that the interface calculates time offset between PI Servers**

Sets the frequency in seconds at which the interface calculates time offsets between PI Data Archives. By default the interface calculates time offsets every 30 seconds. (/OC=#, default: 30)

## Opt Cont tab

- **Source Host reconnection delay**

Specifies the amount of time that the interface waits before attempting to reconnect to the source PI Data Archive after being inadvertently disconnected by network problems. Specify time in seconds (minimum one second, maximum 28800 seconds, which is eight hours). The time is converted to milliseconds before being saved in the batch file. (/DELAYS=x, default: 0 seconds)

- **Target Host reconnection delay**

Specifies the amount of time that the interface waits before attempting to reconnect to the target PI Data Archive after being inadvertently disconnected by network problems. By default the interface does not wait before trying to reconnect. Specify time in seconds

(minimum one second, maximum 28800 seconds, which is eight hours). The time is converted to milliseconds before being saved in the batch file. (*/DELAYR=x*)

- **Suppress writing I/O Timeout to tags upon reestablishment of a lost connection to the source PI Server**

When setting Location3 to write I/O Timeout for any tags, enable this setting to suppress the I/O Timeout state that is normally written to tags when the interface reconnects to the PI Data Archive. If this parameter is not set, the I/O Timeout state that is written at re-connection prevents history from being recovered for the period of disconnection. (*/TS*)

## Source PI Server Failover tab

The settings on this tab enable you to configure failover for source PI Data Archives, so that if one source server fails, the interface can collect data from the other.

- **Enable PItoPI Failover**

Check to enable server failover.

- **Source Server Interface Status Utility Tag**

The PI Interface Status Utility point on the source server that tracks the status of the primary server. Used by failover to determine whether the server is up and running. (*/SSU1=tagname*)

- **Secondary Source Server Node Name**

The port number of 5450 is appended to the name when PI ICU saves the setting in the batch file. (*/SEC\_SRC=nodename:5450*)

- **Secondary Source Int Status Utility Tag**

Specifies the PI Interface Status Utility point on the secondary source server that tracks the status of the server. Used by failover to determine whether the server is up and running. (*/SSU2=tagname*)

- **Number of connection attempts to source server**

Configures the number of connection retries that the interface attempts before failing over to the secondary source server. Default is 1. (*/NT= x*)

- **Enable failover status logging**

Enable failover status logging. If you enable this setting, you must configure the failover status point on target server. (*/FST=tagname*)



# Error and informational messages from the PI to PI Interface

---

Messages are written to %PIHOME%\dat\pipc.log during interface startup and during data collection. Additional messages are logged if you enable debugging. To display details about an error number, issue the following command:

```
\PI\adm\pidiag -e error_number
```

For detailed information about interface logging, see the following OSISOFT Knowledge Base topic:

<http://techsupport.osisoft.com/techsupport/nontemplates/KB/article.aspx?id=KB00401>  
(<http://techsupport.osisoft.com/techsupport/nontemplates/KB/article.aspx?id=KB00401>)

The following tables list interface-specific messages. For details about the messages logged by the UniInt framework, refer to the *UniInt Interface User Guide*.

Message	16-May-06 17:29:06 PItoPI 1> Error -77 returned from pisen_evexceptions call to source PI server.
Cause	Update manager queue limit has been reached on the source PI Data Archive.
Resolution	Increase PI Update Manager queue size limits on source PI Data Archive.





# Command-line parameters for the PI to PI Interface

In addition to the interface-specific parameters, the interface supports standard UniInt command-line parameters. For details, refer to the *UniInt Interface User Guide*.

## Topics in this section

- [General interface operation parameters](#)
- [History recovery and archive data collection parameters](#)
- [Exception data collection parameters](#)
- [Point attribute override parameters](#)
- [Source Data Archive-level failover parameters](#)
- [Sample startup and configuration files](#)

## General interface operation parameters

.BAT	Description
<b>/db=#</b> Optional INI File Setting: DebugFlags	<b>/db=1</b> : Max debug <b>/db=2</b> : Startup processing <b>/db=3</b> : PI Data Archive connections <b>/db=4</b> : PI Data Archive 2.x security validation (obsolete) <b>/db=5</b> : PI point additions, edits, deletions <b>/db=6</b> : Data read & writes <b>/db=7</b> : Failover Example: /db=2,4,5
<b>/delayr=#</b> Optional Default: /delayr=0	Millisecond time delay between reconnection attempts to the target PI Data Archive. Units are in milliseconds. Valid values are between 0 and 28800000ms (8 hours).
<b>/delays=#</b> Optional Default: /delays=0	Millisecond time delay between re-connection attempts to the source PI Data Archive. Valid values are between 0 and 28800000ms (8 hours).
<b>/ist=tagname</b> Optional	Name of interface status point. <b>/ist=&lt;tagname&gt;</b> <i>tagname</i> is a digital point on the target PI Data Archive.
<b>/oc=#</b> Optional Default: /oc=30	Number of seconds between calculating time offset between the interface and PI Data Archive nodes.

Command-line parameters for the PI to PI Interface

.BAT	Description
<p><b>/ps=x</b> Required</p>	<p>The <b>/ps</b> parameter specifies the point source for the interface. <i>X</i> is not case sensitive and can be any single or multiple character string. For example, <b>/ps=P</b> and <b>/ps=p</b> are equivalent.</p> <p>The point source that is assigned with the <b>/ps</b> parameter corresponds to the PointSource attribute of individual PI points. The interface will attempt to load only those PI points with the appropriate point source.</p> <p>If the PI API version being used is prior to 1.6.x or the PI Data Archive version is prior to 3.4.370.x, the PointSource is limited to a single character unless the SDK is being used.</p>
<p><b>/pw=password</b> Obsolete: Required for PI 2 source Optional INI File Setting: SourcePassword</p>	<p>Login password of PI user on PI Data Archive 2 node. Required when source server is PI Data Archive 2.x.</p>
<p><b>/sf=name</b> Obsolete: Required for PI 2 source INI File Setting: SecurityFile</p>	<p>Used for locating the security file on a PI Data Archive 2.x source server.</p> <p>This switch specifies the <i>name</i> part of the file name. Note that the complete file name must have this format:</p> <p><b>PItoPIname . SEC</b></p> <p>where <i>name</i> is the portion specified by <b>/sf</b>.</p>
<p><b>/src_host=name:5450</b> Required INI File Setting: SourceHost</p>	<p>Name or IP address of source PI Data Archive.</p> <p><b>/src_host=node_name:tcpip_port</b></p> <p>The port number is always 5450.</p>

.BAT	Description
<p><b>/stopstat= <i>digstate</i></b></p> <p>or</p> <p><b>/stopstat</b></p> <p><b>/stopstat</b> only is equivalent to <b>/stopstat="Intf Shut"</b></p> <p>Optional</p> <p>Default: no digital state written at shutdown.</p>	<p>If <b>/stopstat=</b><i>digstate</i> is present on the command line, then the digital state, <i>digstate</i>, will be written to each PI point when the interface is stopped. For a PI Data Archive 3.x server, <i>digstate</i> must be in the system digital state table. UniInt will use the first occurrence of <i>digstate</i> found in the table.</p> <p>If the <b>/stopstat</b> parameter is present on the startup command line without a digital state, then the digital state <b>Intf Shut</b> will be written to each PI point when the interface is stopped.</p> <p>If neither <b>/stopstat</b> nor <b>/stopstat=</b><i>digstate</i> is specified on the command line, then no digital state will be written when the interface is shut down.</p> <p>The <b>/stopstat</b> parameter is disabled if the interface is running in a UniInt failover configuration. The digital state, <i>digstate</i>, is not written to each PI point when the interface is stopped, to prevent the digital state being written to PI points while a redundant system is also writing data to the same PI points. The <b>/stopstat</b> parameter is disabled even if there is only one interface active in the failover configuration.</p> <p>Examples:</p> <pre>/stopstat=shutdown /stopstat="Intf Shut"</pre> <p>The entire <i>digstate</i> value must be enclosed within double quotes when there is a space in <i>digstate</i>.</p>
<p><b>/ts</b></p> <p>Optional</p>	<p>Suppress IO Timeout events when reconnecting to source PI server. These events are configured through the Location3 attribute.</p> <p>If this switch is not set, the event written at reconnection will prevent history from being recovered for the period of the disconnection.</p>

## History recovery and archive data collection parameters

Parameter	Description
<p><b>/dc</b></p> <p>Optional</p>	<p>Apply data compression to history recovery and archive scan updates. The default behavior is for this data to bypass compression.</p> <p>This switch must be specified to prevent data mismatches if points are configured to include snapshot value with archive scan updates.</p>

Parameter	Description
<p><b>/hronly</b>[=&lt;start,end&gt;]</p> <p>Optional</p> <p>INI File Setting: HistOnly</p>	<p>Specifies the time range to be recovered. The interface recovers data for the specified period, then exits. To disable exception data collection, omit time period.</p> <p>Specify the times using PI time string formats with a colon or underscore separating the date and the time:</p> <p><b>/hronly</b>=dd-mmm-yy:hh:mm:ss,dd-mmm-yy:hh:mm:ss</p> <p>or</p> <p><b>/hronly</b>=dd-mmm-yy_hh:mm:ss,dd-mmm-yy_hh:mm:ss.</p> <p>For example: /hronly=10-dec-98:10:00,10-dec-98:12:00</p> <p>or</p> <p>/hronly=10-dec-98_10:00,10-dec-98_12:00</p> <p>If the source and target PI Data Archives are in different time zones, time stamps are local to the node runs.</p>
<p><b>/hrpause</b>=#</p> <p>Optional</p> <p>INI File Setting: HistPause</p> <p>Default: /hrpause=0</p>	<p>Milliseconds to pause between points during history recovery. Used to throttle archive data retrieval during history recovery.</p>
<p><b>/mh</b>=x</p> <p>Optional</p> <p>Default: /mh=1000</p>	<p>Supported for target PI Data Archive version 3.3 or later.</p> <p>Sets the maximum number of archive events retrieved per data request. If more than the maximum exist, the interface makes multiple calls until all events are retrieved for the time period. Increasing the maximum can increase data throughput for archive data retrieval.</p>
<p><b>/ns</b></p> <p>Optional</p>	<p>Start history recovery beginning with the first value prior to the start time. By default, recovery begins with the first value after the start time.</p>
<p><b>/rh</b>=#</p> <p>Optional INI File Setting: HistRecovery</p> <p>Default: /rh=8</p>	<p>Hours of history recovery to perform. No maximum is enforced, but be sure you have sufficient disk space for the archive files on the target server computer. For information on backfilling data, see the <i>PI Server System Management Guide</i>.</p>


Parameter	Description
<p><b>/rh_inc=#</b></p> <p>Optional</p> <p>INI File Setting: MaxArcTimespan</p> <p>Default: /rh_inc=24</p>	<p>Time increments within the total <b>/rh</b> recovery period.</p> <p>For example, if <b>/rh=48</b> and <b>/rh_inc=24</b>, the interface cycles through the point list twice. On the first cycle, the first 24-hour period is recovered. On the second cycle, the second 24-hour period is recovered.</p>
<p><b>/rh_qcheck=#</b></p> <p>Optional</p> <p>Default: /rh_qcheck=5</p>	<p>For exception data scan classes, specify how often to clear the exception queue on the source PI Data Archive during history recovery.</p> <p>To prevent queue overflow, the interface periodically collects exceptions from the source PI Data Archive during history recovery. You can adjust this time interval to tune history recovery performance.</p>

## Exception data collection parameters

Parameter	Description
<p><b>/me=#</b></p> <p>Optional</p> <p>Default: /me=5000</p>	<p>Sets the maximum number of exception events retrieved per data request. A large count reduces the number of calls required to read exception updates. A small count reduces the time to complete each request (for troubleshooting network timeout issues).</p>
<p><b>/sn</b></p> <p>Optional</p>	<p>Disable exception filtering for data collected from the source PI Data Archive. This data has already passed exception for the source tag. Additional data filtering can lead to data mismatches between PI Data Archives.</p>

## Point attribute override parameters

Parameter	Description
<p><b>/c1</b></p> <p>Optional</p>	<p><b>Location1</b> point attribute override.</p> <p>Load all points configured for the specified point source regardless of <b>Location1</b> and interface ID values.</p>
<p><b>/c2=#</b></p> <p>Optional</p>	<p><b>Location2</b> point attribute override.</p> <p>Ignore <b>Location2</b> for each point and use the specified value. Valid values are 0-7.</p>
<p><b>/c3=#</b></p> <p>Optional</p>	<p><b>Location3</b> point attribute override.</p> <p>Ignore <b>Location3</b> for each point and use the specified value. Valid values are 0-15.</p>

Parameter	Description
<b>/c4=#</b> Optional	<b>Location4</b> point attribute override.  Ignore <b>Location4</b> for each point and use the specified value. Valid values are 1 (exception data collection) or 2 (archive data collection).
<b>/c5=#</b> Optional	<b>Location5</b> point attribute override.  Ignore <b>Location5</b> for each point and use the specified value. Valid values are 0-3.
<b>/ptid</b> Optional	Point ID of source PI point is specified in <b>UserInt1</b> attribute. Ignore <b>InstrumentTag</b> , <b>ExDesc</b> and <b>Tag</b> (PI point name) attributes.  <div style="background-color: #e6f2ff; padding: 5px;">  <b>Note:</b>                          This switch is not compatible with source PI Data Archive failover because point IDs will not necessarily match between source PI Data Archives. If the source PI Data Archives are part of a PI collective use <b>/tn</b> instead.                     </div>
<b>/tn</b> Optional	Source point name is same as interface point name. Ignore <b>InstrumentTag</b> , <b>ExDesc</b> and <b>UserInt1</b> attributes.
<b>/tnex</b> Optional	Source point name is located in the <b>ExDesc</b> or <b>Tag</b> attribute and the <b>InstrumentTag</b> and <b>UserInt1</b> attributes are ignored for identifying source point.

## Source Data Archive-level failover parameters

Parameter	Description
<b>/fst=tagname</b> Optional	Name of failover status point.  <b>/fst=&lt;tagname&gt;</b>  where <i>tagname</i> is a digital point on the target PI Data Archive.
<b>/nt=#</b> Optional Default: /nt=1	Number of re-connection attempts to source PI server before initiating a failover. Valid values are 0 and greater.  Prevents failover flip-flop when experiencing intermittent network updates.
<b>/sec_src=node:5450</b> Required	Name or IP address of the secondary source PI Data Archive.  <b>/sec_src=node_name:5450.</b>  The port number is 5450 for PI Data Archive 3.x. PI Data Archive 2.x is not supported for source PI Data Archive failover.

Parameter	Description
<p><i>/ssu1=tagname</i></p> <p>Required</p>	<p>PI Interface Status Utility point name for the / <b>src_host</b> source PI Data Archive.</p> <p><i>/ssu1=&lt;tagname&gt;</i></p> <p>Required for monitoring source data quality (current or stale data).</p>
<p><i>/ssu2=tagname</i></p> <p>Required</p>	<p>PI Interface Status Utility point name for the / <b>sec_src</b> source PI Data Archive.</p> <p><i>/ssu2=&lt;tagname&gt;</i></p> <p>Required for monitoring source data quality (current or stale data).</p>

## Sample startup and configuration files

The startup files for the interface reside in the directory %PIHOME%\Interfaces\PItoPI. %PIHOME% is defined in %WINDIR%\pipc.ini by the installation program. By default, %PIHOME% is c:\pipc. The startup files are:

- **PItoPI.bat**: Invokes the interface, specifying run-time settings are command-line parameters.
- **PItoPI.ini**: Stores settings required for collecting data from multiple sources.

Do not edit these files manually. Use PI ICU to configure the interface.

### Topics in this section

- [Sample PItoPI batch file](#)
- [Sample PItoPI configuration file](#)

## Sample PItoPI batch file

```

REM=====
REM
REM   PIttoPI.bat
REM
REM Sample startup file for the PI to PI Interface
REM
REM=====
REM
REM OSISOFT strongly recommends using PI ICU to modify startup files.
REM
REM Sample command line
REM
REM   .\PItoPI.exe ^
REM   /host=XXXXXX:5450 ^
REM   /src_host=XXXXXX:5450 ^
REM   /ps=PItoPI ^
REM   /id=1 ^
REM   /f=10
REM End of PIttoPI.bat File

```

## Sample PItOPI configuration file

```

;-----
; Purpose:
;   This file is used in conjunction with PItOPI.bat. It is only
;   required to collect data from multiple source
;   servers using a single copy of the interface.
;
;   the headings read [PItoPI-x.y] where;
;   x = interface id
;   y = scan class (if specified)
;-----
;
; [PItoPI-1]
; EventCounter=1
; MaxArcTimespan=24
;
; [PItoPI-1.1]
; SourceHost=XXXXXX:5450
; HistRecovery=48
;
; [PItoPI-1.2]
; SourceHost=XXXXXX:5450
; HistRecovery=72
;
;-----
; List of possible parameters
;
; SourceHost=XXXXXX:5450      Name of source PI Data Archive,
;                             port=5450 for PI Data Archive 3.x and 545 for PI
Data Archive 2.x
; TargetHost=XXXXXX:5450    Name of the target PI Data Archive,
;                             port=5450 for PI Data Archive 3.x and 545 for PI
Data Archive 2.x
; SecurityFile=securityfile  Required if PI Data Archive 2.x, <name> part of
security file
;                             PItOPI<name>.SEC
; SourceLogin=userid        PI Data Archive 2.x PI user name
; SourcePassword=password   PI Data Archive 2.x PI user password
; EventCounter=#           Number of event counter defined in
;                             \dat\iorates.dat file
; HistRecovery=#           total hours of history recovery, default=8hrs
; MaxArcTimespan=#         history recovery increment, divided into total
;                             hours of history recovery (HistRecovery),
;                             default=24hrs
;
; HistPause=#              pause between history recovery increments in
;                             milliseconds
; HistOnly=#               flag to disable exception data collection
;                             (0=off, 1=on)
; DebugFlags=#,#,#,#...    Generates additional messages for troubleshooting
;                             comma separated list: 1,2,3,4,5,6,7
;-----
; end of sample PItOPI.ini

```



## Technical support and other resources

---

For technical assistance, contact OSIsoft Technical Support at +1 510-297-5828 or through the [OSIsoft Tech Support website \(https://techsupport.osisoft.com\)](https://techsupport.osisoft.com). The website offers additional contact options for customers outside of the United States.

When you contact OSIsoft Technical Support, be prepared to provide this information:

- Product name, version, and build numbers
- Details about your computer platform (CPU type, operating system, and version number)
- Time that the difficulty started
- Log files at that time
- Details of any environment changes prior to the start of the issue
- Summary of the issue, including any relevant log files during the time the issue occurred

The [OSIsoft Virtual Campus \(vCampus\) website \(http://vcampus.osisoft.com\)](http://vcampus.osisoft.com) has subscription-based resources to help you with the programming and integration of OSIsoft products.

