PI World 2019 Lab

Managing Time Series Data Quality for Business Data Owners and PI Administrators

OSIsoft, LLC
1600 Alvarado Street
San Leandro, CA 94577 USA
Tel: (01) 510-297-5800
Web: http://www.osisoft.com

Published: March 21, 2019

Table of Contents

1. Introduction

Real-time data must meet sufficient quality criteria to transform from data to business intelligence that delivers tangible value. This lab outlines the basic categorization of data quality; various methods to measure and monitor your data quality; and the organization framework required to institutionalize data governance. This lab should be of particular interest to data system administrators, IT management, and business owners depending on real-time data for daily operation.

In this lab, you will learn about:

- Data Quality (of course), including its value, its definition and how to achieve data quality
- Key concepts around Data Quality Monitoring
- Practical (hands-on) examples of
    - Monitoring System Health
    - Performing Simple Error Analysis
- Concepts around advanced methodologies of Data Quality Monitoring

2. Data Quality Concepts

In this section, you will learn about key concepts around data quality.

2.1 Why is data quality important for your organization?

Data can influence and drive all aspects of an organization to success, from plant-floor operations to future market-wide predictions. However, decisions based on faulty data can lead to poor planning and various different operational issues. Key affected areas include, but are not limited to:

Safety and Security

Safety is critical to effective operations. It is inherent in establishing optimal process control. Real-time data is instrumental in recognizing and mitigating process hazards before an incident occurs.

Regulatory Compliance

Safety, health, and environmental records are critical to sustaining a business's "license to operate". These are often subject to agency review, making accurate records a requirement. Improper reporting can lead to fines, enhanced surveillance, and even closure.

Process Monitoring and Optimization

By monitoring the process in real-time, operations and technical personnel are able to control and optimize material and energy conservation, ensure safe operating conditions, and meet production plans. Poor quality data could lead to inefficient operations or failure to ship.

Asset Health

Real-time data is used to gauge the current condition and predict future operability of critical equipment. Systematically tracking asset performance metrics empowers overall asset maintenance programs. Inadequate data may miss known warning signs, leading to premature failures. Unplanned failures lead to higher maintenance costs, missed production, or safety and environmental incidents.

Quality and Reputation

Accurate and timely operation of the process determines product quality. Data quality can impact product quality through poor control, inaccurate sampling, and erroneous reporting. Poor product quality or measurement may require rework, disposal, or other inefficiencies, which may impact a shipment and damage reputation.

Cost Control

Efficient operations enable cost effective operations. Deviations from plan, safety or environmental incidents, or product quality excursions may all incur a financial penalty. Quality decisions require quality data.

Innovation

Exploration and discovery require a keen understanding of the present and the past. Only high quality representative datasets are adequate for new advances in machine-learning, artificial intelligence, and business intelligence platforms. Failure to manage baseline data quality can prevent companies from discovering new business opportunities and breakthrough operational technologies.

2.2 What is Data Quality? Data fit for its intended purposes

The first step to managing data quality is to understand what defines "quality". In simple terms, it is data that is fit for its intended purpose. There are several lists of data quality metrics publicly available, depending on the industry and data type. These metrics can be both subjective and/or objectively measured indicators. OSIsoft recommends customers to consider the measurable qualitative metrics below and to review potential sources for data quality issues.

The following quantitative metrics are paramount to building a data quality reputation that leads to high user adoption and future value opportunity.

Accurate

Is the data correct and reliable? Can it be trusted?

Complete

Has all the necessary data been captured? Are there gaps in data or periods of static "flat-line" data collection?

Conforming

Does the data system (PLC, SCADA, Historian, etc.) configuration conform to the data source configuration? Is data captured at the correct frequency, within the right span, and at the appropriate level of accuracy? Is data properly compressed without compromising its integrity?

Consistent

Is the data consistent with normal functioning of the instrument and process?

Unique

Is each real-time data point unique at its time stamp?

Timely

Is data delivered when necessary for business needs?

Auditable

Can the data be traced through the system from inception to consumption for validation?

Accessible

Is data quickly and easily available to all appropriate consumers?

2.3 Who is responsible? The Data Governance Roles

To achieve and, more importantly, sustain high quality data requires that a support framework be built with the roles below properly staffed as your data profile grows

Data Governance Council

The data governance council (DGC) provides the structure and oversight necessary for maintaining a data quality program. Members of the DGC have the organizational authority to establish and implement policy. The DGC should include members from both IT and business groups. Tasks for the DGC include:

- Establish data governance policies
- Monitor compliance and manage change
- Resolve conflicts in policy
- Allocate resources

Data Owner

The data owner has overall accountability for the governance of data within a defined area. Data owners work together to provide consistent execution of data governance policy. Tasks for each data owner include:

- Define and assign process elements conforming to the data governance policy
- Implement the data governance policy
- Train consumers and other contacts on the need and application of the data governance policy
- Identify and report data risks to the DGC and consumer

- Resolve exceptions to the policy with the data administrator
- Manage change to both the data and governance process

Data Administrator

The data administrator works on the frontline of data governance. The tasks of the data administrator include:

- Monitor and manage data quality checks
- Identify and report data risks to the data owner
- Identify data inconsistencies and work with Data Owner(s) to resolve
- Report incidents impacting data quality
- Implement approved changes to data quality systems

Consumer

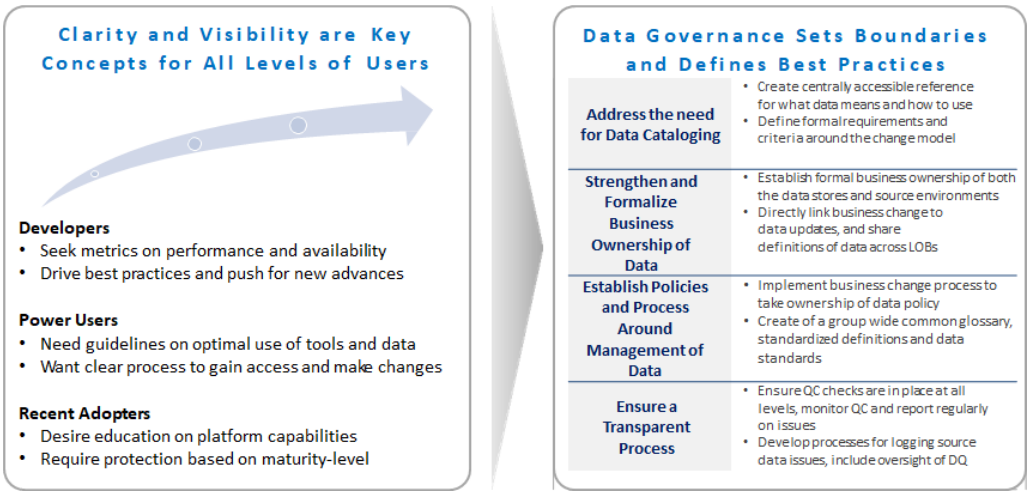
The data consumer is the end customer of quality data. The consumer group uses data to develop operational and business intelligence to achieve business priorities. The consumer is often the first to recognize data quality errors, and should work with the data governance team to mitigate these issues.

2.4 How can I achieve this? The Data Governance Process

System Standards

The DGC will define and document standards used to govern the collection and use of real-time data. These may include but are not limited to:

- Instrumentation / hardware standards
- Data retention and backup procedures
- Naming conventions
- Data profiles with corresponding data quality requirements
- Data monitoring protocols
- Change management



Data Profiles

Data may be classified by the business need (i.e. safety, environmental, control, monitoring, etc.) profiled for proper configuration, data quality rules, and overall prioritization.

Prioritization

Not all data is business critical. Data used for monitoring water tank temperatures may not require as rigorous governance as emissions sampling for example. Once data is classified, it should be prioritized based on business need. Analysis of the business drivers satisfied by the data point will identify critical points.

Configuration

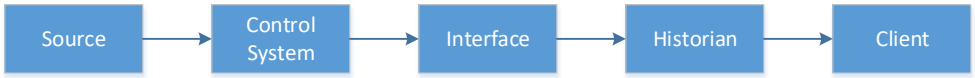
Types of instruments, business drivers all may require different configuration standards as they transmit data from source to consumer. Issues of redundancy, latency, scan rates, and retention may be covered by the DGC.

Quality Rules

Classification and prioritization of data determine the amount of validation required of a given data point. Higher priority classes and data points will require more thorough testing for data quality.

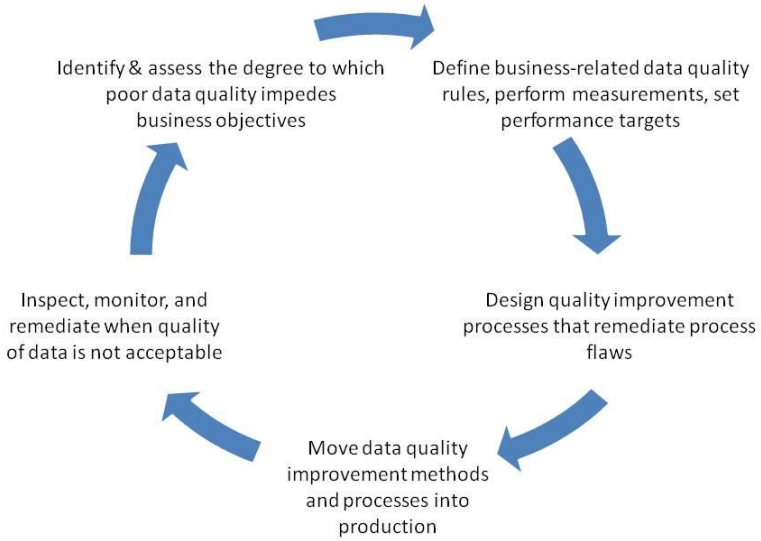
Data Map

Understanding the flow of data from source to consumer is critical to understanding and mitigating the potential for error in the data stream. System health is the foundation to overall data quality. Map the data flow to identify implementation points for data quality and system monitoring (i.e. calibration standards, interface and network monitoring, data checks, etc.).



Sustain

Data governance requires maintenance. Monitor for policy adherence as well as for data quality. Continuously improve.



Change Management

All systems grow over time. Change is constant but should be managed constructively so that impacts are understood and new policies or activities are communicated. Where possible, data governance should utilize existing enterprise change management structures.

What is Change?

Changes to policy or to the data stream should warrant some level of scrutiny before implementation. The level of scrutiny depends on the type of change and the criticality of its impact. Changes that may impact data quality or the overall data governance may include:

- Configuration changes to hardware, software, or the data stream
- Tasks and role alignment within the data governance policy
- Instrumentation changes
- Changes to the contextual layer and templates for data consumption
- Changes to controlled visualization or business critical data consumers

Impact Assessment

The impact of the change depends on business drivers and policies effected by the change. Evaluate changes against the uses cases for the data, the roles and workflows touched, and effort to implement.

Approval

The criticality and impact of the change may be used to determine the level of approval for required for the change. In many cases, small changes may require the approval of an impacted individual. In others, larger changes to either policy or the data stream may warrant review by the DGC. This should be documented per the data governance policy.

Communication

Once the change has been approved, the change should be communicated. The type of change and the impact of the change determine the level of communication and training required. Communication plans may range from email notices to face-to-face training.

Follow Up

In order to maintain the system and account for change, change management requires documentation and validation. For this reason, document the cycle of change from request, design, approval, and communication. In many cases, the change should be documented in the effected policy, configuration lists, or hardware libraries. Validate that the change has been implemented correctly and is in service.

2.5 Data Monitoring

Configuration Checks

Configuration standards are a prerequisite for data quality monitoring. The base configurations of data classes should be known and validated before proceeding with more extensive monitoring. Validate that the data has been setup correctly. Basic data configuration should include consistent:

- Unit of measure
- Scan rates
- Compression and exception ratios
- Naming conventions

Once validated, data configuration should be validated periodically against standards to monitor for change and adherence to policy.

System Health

Instrumentation, interfaces, control systems, and networks have to be functioning correctly in order to maintain data quality. These devices may have system flags and watchdogs to determine service status. In addition, transmission rates, buffer queues, and other data stream checks should be monitored.

Simple Error Analysis

Data can be checked for simple errors based on base data quality rules. These include:

- Boundary Limits: Has the value violated a maximum or minimum measurement limit?
- Rate of Change: Has the rate of change of the value violated the process or measurement capability?
- Stale Data: Has the data ceased to update or flat-lined?
- Run Status: Is the equipment out of service with no value measured?
- Bad Data: Has the data been listed as “bad” as defined by the control system or is the data set missing over tested time ranges?

Advanced Methodologies

Once system health and simple errors have been monitored, more advanced systems can be used to validate data quality. For example:

- SQC / SPC: Statistical methods to determine data quality deviations
- Pattern Recognition / AI: Artificial intelligence and pattern recognition algorithms
- Process Modeling: First principle and other models to calculate process values for comparison

Frequency

Not all quality checks can or should be monitored in real-time. In most cases, the availability of testing methods and the criticality of the data profiles will dictate the timing. Many tools exist for online system health monitoring. Configuration may be evaluated at some frequency as required by policy and change management. Methods exist for real-time simple error detection. More advanced methodologies provide additional options. Evaluate and tailor quality testing based on need and capability.

2.6 Additional Resources

System Monitoring

Interface Health Monitoring
https://customers.osisoft.com/s/knowledgearticle?knowledgeArticleUrl=KB00830

System Health Monitoring (PI Interface for Performance Monitor)

PI Server
https://livelibrary.osisoft.com/LiveLibrary/content/en/server-v11/GUID-F7CF8D1A-B941-48E1-93E7-046B5B24DC8B

Interfaces
https://livelibrary.osisoft.com/LiveLibrary/content/en/int-perf-mon-v2/GUID-FC0E918E-18D9-40B8-89A0-B8193BA1677C

Analysis Service
https://livelibrary.osisoft.com/LiveLibrary/content/en/server-v11/GUID-7BBDE09C-E055-4920-BA7B-7CE24437CFF8

SQL Server
https://customers.osisoft.com/s/knowledgearticle?knowledgeArticleUrl=KB00992

PI World Presentations of interest

Better Data Quality for Better Data Science with the PI System (PI World 2018 Barcelona)
https://www.osisoft.com/Presentations/Better-Data-Quality-for-Better-Data-Science-with-the-PI-System/

Monitoring Data Quality with Asset Analytics
https://www.osisoft.com/Presentations/Monitoring-Data-Quality-with-Asset-Analytics/

Partner Information

Seeq - Improving Data Quality using Seeq and PI AF (PI World 2018 Barcelona)
https://www.osisoft.com/Presentations/Seeq---Improving-Data-Quality-using-Seeq-and-PI-AF/

Element Analytics
https://www.elementanalytics.com/blog/for-data-quality-sensor-trust-is-key

3. Directed Activity – Interface Health Points

3.1 Adding Health Points to Monitor Interfaces

UniInt health points are supported by interfaces built with UniInt 4.3.0.0 and later. Health Points gather diagnostic information about the status of different aspects of a PI Interface. This is useful for understanding when a problem occurs what is the root cause. Further information about Interface Health Points is available in:

https://customers.osisoft.com/s/knowledgearticle?knowledgeArticleUrl=KB00830.

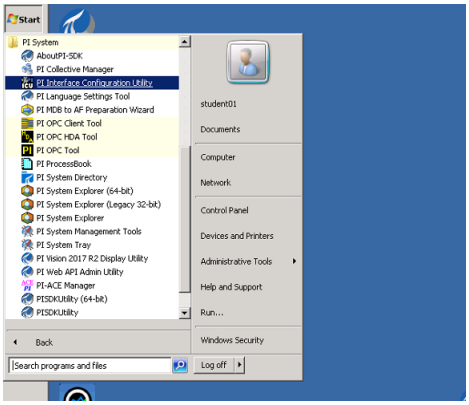
Three key general health points are Device Status, Heartbeat and IO Rate.

3.2 Creating Interface Health Points

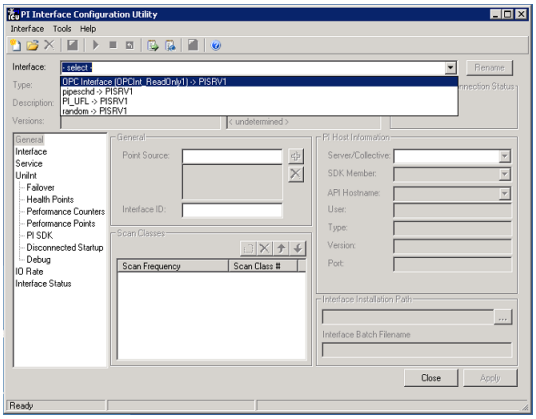
- Open Interface Configuration Utility
- Identify and Create PI Health Points

3.3 Creating Interface Health Points – Step by Step

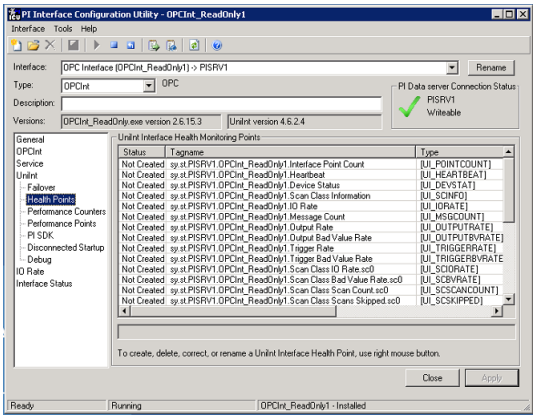
- Step 1, open PI Interface Configuration Utility



- Step 2, select the PI Interface **OPC Interface**

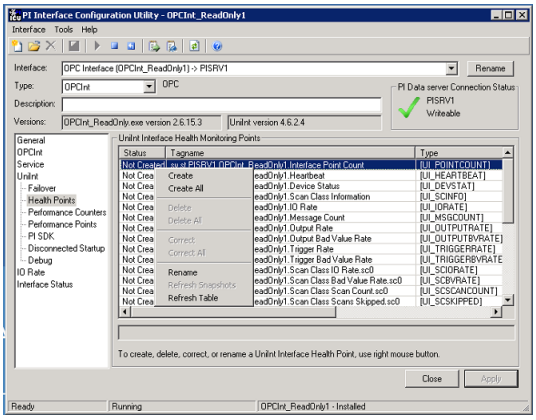


- Step 3, select the **UniInt | Health Points** section



- Step 4, on each of the Interface Health Monitoring Points required **right-click** and select **Create**

Create three Health Points: **UI_HEARTBEAT**, **UI_DEVSTAT** and **UI_IORATE**



The status of the three Health Points should show as Created, while the remaining Health Points should be Not Created.

4. Directed Activity – Device Status

4.1 Capturing the Status of a Data Source

The Device Status Health Point stores communication information about the interface and the data source. The information is provided as string values with a code and message. To make use of this information it is useful to break out the code and the message into separate attributes. For more information, see:

https://customers.osisoft.com/s/knowledgearticle?knowledgeArticleUrl=KB00932

4.2 Analyzing Device Status Health

- Create attributes to show Device Status Code, Message and Health
- Create expressions to decode Device Status and Health
- Test Device Status Health

4.3 Analyzing Device Status Health – Step by Step

The Device Status value will be split into its Code and Message. Additionally the health will be evaluated.

- Step 1, in the **PI Interface Element Template** create three new Attributes to contain the derived Device Status information.

Name	Description	Categories	Default UOM	Value Type	Data Reference	Display Digits	Settings
Device Status – Code	**Interface Device Status Code**	**Device Status**	**<None>**	**Int32**	**PI Point**	**0**	**\\%Server%\%Element%.%Attribute%;pointtype=Int32**
Device Status – Message	**Interface Device Status Message**	**Device Status**	**<None>**	**String**	**PI Point**	**0**	**\\%Server%\%Element%.%Attribute%;pointtype=String**
Device Status – Health	**Data Source Health**	**Device Status, Health Status**	**<None>**	**Health**	**PI Point**	**0**	**\\%Server%\%Element%.%Attribute%;pointtype=Int32**

Tip

Using the Tag Creation functionality in AF Element Templates is a great way to automate and standardize PI Point building

- Step 2, in the PI Interface Element Template create a new Analysis Template

The following are suggested expressions to isolate and return the code, message and health

Name	Expression	Value at Evaluat	Value at Last Tri	Output Attribute	
DevStat	`String('Device Status')`			Map	⊗
Break	`// Device Status message is usually formed %code% | %message%, find the pipe symbol location` `InStr(DevStat, "|")`			Map	⊗
HasBreak	`// Some Device Status reports without the pipe symbol, does this message have the break?` `Break > 0`			Map	⊗

DScode	`If HasBreak Then`
	`  // return the first section of the status with the code`
	`  Int(Left(DevStat, Break - 1))`
	`Else`
	`  // return an inidcation that there is no code`
	`  -1`
				Device Status - Code	⊗

DSmessage	`If HasBreak Then`
	`  // return the second section of the status with the message`
	`  Mid(DevStat, Break +1)`
	`Else`
	`  // return the whole status if there is no sectioning`
	`  DevStat`
				Device Status - Message	⊗

DSgood	`// Healthly when the code is zero, or if there is no code and the message is "Good"`
	`If DScode = -1 Then`
	`  If UCase(Trim(DSmessage)) = "GOOD" Then "Healthy" Else "Unhealthy"`
	`Else`
	`  If DScode = 0 Then "Healthy" Else "Unhealthy"`
				Device Status - Health	⊗

Remember to map the Output Attributes

✓ Best Practice	When creating an analysis it is good practice to use informative variable names, comments and use indentation to increase readability

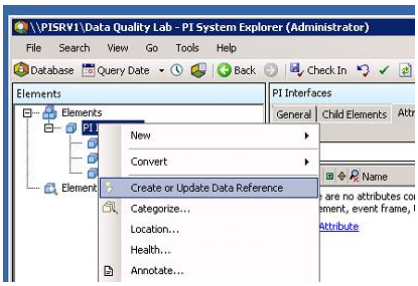
- Step 3, **evaluate** the expression to ensure it works as expected. To do this you will need to click the **Select an example** hyperlink and select an element to perform the evaluation against

- Step 4, to schedule this analysis **Event-Triggered** is appropriate as the result needs to be calculated for every Device Status value

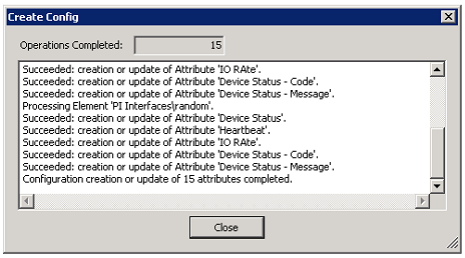
✓ Best Practice	Always be mindful of scheduling so that you select the most appropriate frequency – do not to schedule expensive high frequencies when not needed

With the Element Template updated creating the PI Points for the outputs is a simple task.

- Step 5, once the analysis is created, verified and checked in return to the **Elements** panel and **right-click** on the **PI Interfaces container element** and select **Create or Update Data Reference**.



This will create the three PI Points defined in the PI Interface template for each of the child PI Interface elements

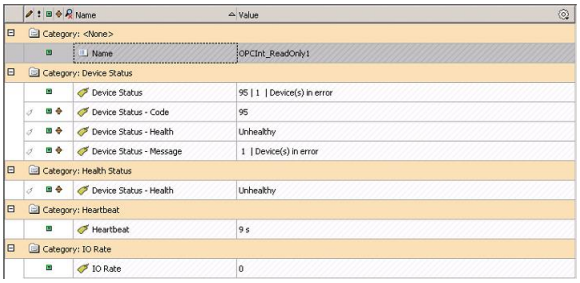


The Device Status will now be evaluated and its Code, Message and Health will be stored in PI Points. This can be used for immediate information about the status of the data source for an interface, and also used for historical analysis for processes such as root cause analysis.

By interrupting the data source a simulated outage can be performed to test the result.

- Step 6, select the **OPCInterface_Readonly1** element and observe the Device Status attributes. They are likely to be Pt Created or – hopefully – indicating healthy.

- Step 7, on the **Desktop** of the virtual machine is a batch file to stop the PI OPC Server service. This is the data source for this interface. Execute the **Stop OPC Server** batch file and observe the Device Status attributes. You may need to use the refresh button to update the values.



Name	Value
Category: <None>	
Name	OPCInt_ReadOnly1
Category: Device Status	
Device Status	95 | 1 | Device(s) in error
Device Status - Code	95
Device Status - Health	Unhealthy
Device Status - Message	1 | Device(s) in error
Category: Health Status	
Device Status - Health	Unhealthy
Category: Heartbeat	
Heartbeat	9 s
Category: IO Rate	
IO Rate	0

- Step 8, execute the **Start OPC Server** batch file to restore the data source and observer that the status will return to healthy.

5. Directed Activity – Interface Status (Heartbeat)

5.1 Capturing the Status of an Interface

The Heartbeat Health Point is an indication if the interface is running and should update regularly. As there is no embedded information to decode from the Heartbeat deriving health is a simple check if the heartbeat is updating.

5.2 Analyzing Heartbeat Status

- Create attribute to show heartbeat health
- Create expression to derive health
- Test heartbeat health

5.3 Analyzing Heartbeat Status – Step by Step

- Step 1, return to the PI Interface Element Template and create a new Attribute Template:

Name	Description	Categories	Default UOM	Value Type	Data Reference	Display Digits	Settings
Heartbeat – Health	**Interface Running**	**Heartbeat, Health Status**	**<None>**	**Health**	**PI Point**	**0**	**\\%Server%\%Element%.%Attribute%;pointtype=Int32**

- Step 2, in the PI Interface Template create a new Analysis Template

  The following are suggested expressions to identify if the heartbeat is good and updating

Name	Expression	Value at Evaluati	Value at Last Tri	Output Attribute	
HB	`'Heartbeat'`			Map	⊗
StaleLimit	`// check if the heartbeat has updated in the last 30 seconds` `'*-30s'`			Map	⊗
LastHB	`PrevEvent(HB, '*')`			Map	⊗

HBgood	`// is the heartbeat good and is the latest timestamp more rececent than the defined limit`
	`If BadVal(HB) Then "Unhealthy" Else`
	`  If LastHB < StaleLimit Then "Unhealthy" Else "Healthy"`
				Heartbeat - Health	⊗

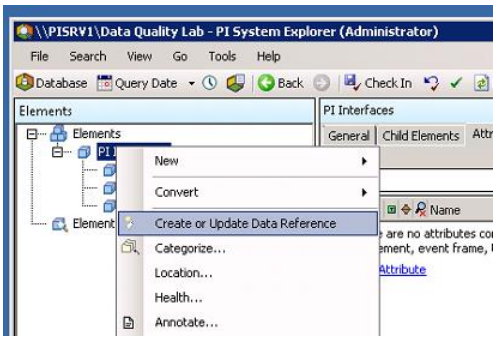
  Remember to map the Output Attribute

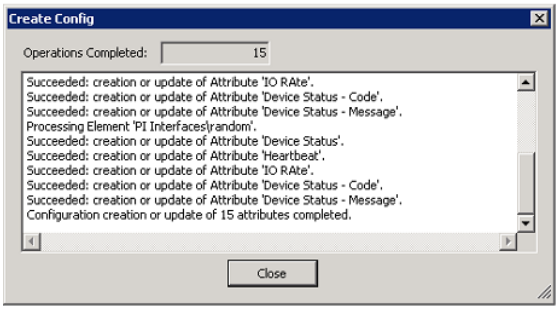
- Step 3, **evaluate** the expression to ensure it works as expected. To do this you will need to click the **Select an example** hyperlink and select an element to perform the evaluation against.

- Step 4, to schedule this analysis **Periodic** is appropriate as the result needs to be calculated independently of incoming data. How frequently the PI Analysis should be scheduled will be determined by how critical the interface is. In this

example case the schedule will be to execute **every 15 seconds**.

- Step 5, once the analysis is created, verified and checked in return to the **Element** panel and **right-click** on the **PI Interfaces container element** and select **Create or Update Data Reference**

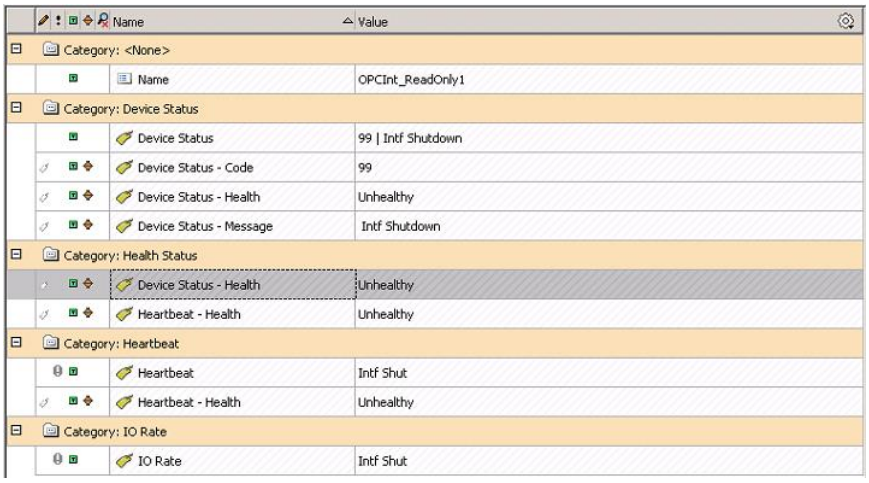




Testing the heartbeat is simply a case of stopping the PI Interface and observing the result.

- Step 6, select the **OPCInterface_ReadOnly1** element and observe the **Heartbeat** attributes – they are likely to be Pt Created or – hopefully – healthy.

- Step 7, on the **Desktop** of the virtual machine is a batch file to stop the PI OPC Interface service. Execute the **Stop Interface** batch file and observe the **Heartbeat** attributes. You may need to use the Refresh button to update the values.



Name	Value
Category: <None>	
Name	OPCInt_ReadOnly1
Category: Device Status	
Device Status	99 | Intf Shutdown
Device Status - Code	99
Device Status - Health	Unhealthy
Device Status - Message	Intf Shutdown
Category: Health Status	
Device Status - Health	Unhealthy
Heartbeat - Health	Unhealthy
Category: Heartbeat	
Heartbeat	Intf Shut
Heartbeat - Health	Unhealthy
Category: IO Rate	
IO Rate	Intf Shut

- Step 8, execute the **Start Interface** batch file to restore the interface.

6. Directed Activity – Interface Status (IO Rate)

6.1 Capturing Data Transfer

IO Rate counts the number of all values sent to PI Server before exception processing occurs. IO Rate can be used to great effect to identify more subtle issues with data transfer. This requires an understanding of what the IO Rate should be during stable/normal operation. It is recommended to baseline a stable interface IO Rate and set limits based on that information.

6.2 Analyzing IO Rate

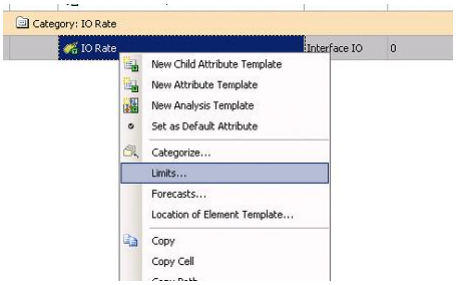
- Create attribute to show IO Rate Health
- Create configurable minimum rate Limit
- Create expression to derive IO Rate Health

6.3 Analyzing IO Rate – Step by Step

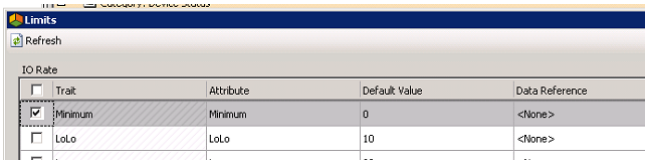
- Step 1, return to the PI Interface Element Template and create a new Attribute Template:

Name	Description	Categories	Default UOM	Value Type	Data Reference	Display Digits	Settings
IO Rate – Health	**Interface IO Health**	**IO Rate, Health Status**	**<None>**	**Health**	**PI Point**	**0**	**\\%Server%\%Element%.%Attribute%;pointtype=Int32**

- Step 2, create a Minimum limit for the IO Rate attribute by **right-clicking** on the **IO Rate attribute** in the **PI Interface Element Template** and selecting **Limits …**



- Step3, click the checkbox for Minimum, click OK and check in



- Step 4, in the PI Interface template create a new Analysis Template

  The following are suggested expressions to confirm the IO Rate above the Minimum limit.

Name	Expression	Value at Evaluati	Value at Last Tri	Output Attribute	
IO	`'IO Rate'`			Map	⊗
IOmin	`'IO Rate|Minimum'`			Map	⊗
IOmax	`// Highest IO in the last 10 mins` `TagMax(IO, '*', '*-10m')`			Map	⊗

IOgood	`// IO rate is good and is greater than the specifeid minimum limit`
	`If BadVal(IO) Then "Unhealthy" Else`
	`  If IOmax > IOmin Then "Healthy" Else "Unhealthy"`
				IO Rate - Health	⊗

  Remember to map the Output Attribute

 Tip	Use the Attributes browser on the right panel to insert child attribute references

- Step 5, **evaluate** the expression to ensure it works as expected.
- Step 6, to schedule this analysis **Event-Triggered** is appropriate as the result can be calculated for every IO value.
- Step 7, once your Analysis is created, tested and checked in return to the **Element** panel and **right-click** on the **PI Interfaces** container Element and select **Create or Update Data Reference**
- This will create the PI Points referenced by the IO Rate- Health attributes in the child elements.

7. Directed Activity – Interface Status Summary

7.1 Summarizing the Health of an Interface

In the previous activities we created three attributes indicating aspects of health for a PI Interface. These were:

- Device Status; reporting the communication between the interface and the data source.
- Heartbeat; reporting if the interface is running.
- IO Rate; reporting the quantity of data flowing through the interface.

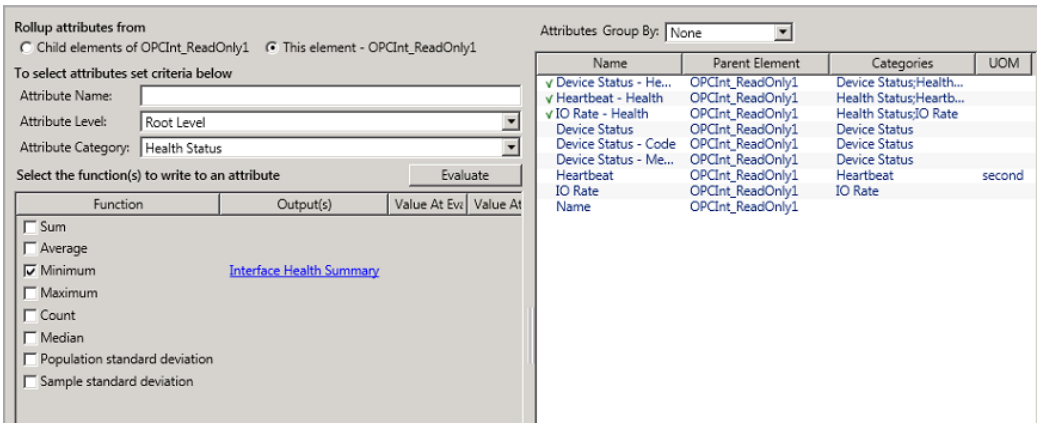
Rather than tracking these health status individually we can reduce the effort needed to monitor data flows by summarizing the health into an overall health status for the interface. If any of the status show bad health the summary will show the PI Interface as unhealthy.

7.2 Summarizing Health

- Create Attribute for PI Interface Health Summary
- Create Rollup to Summarize the PI Interface Health
- Create Event Frames to capture Unhealthy periods
- Backfill results for visualization

7.3 Summarizing Health – Step by Step

- Step 1, create a new **Analysis Template**, select the **Rollup** analysis type
- Step 2, configure the Rollup attributes from This Element
- Step 3, leave the **Attribute Name** as **blank** to catch all matching attributes
- Step 4, set the Attribute Category to Health Status
- Step 5, select the summary function **Minimum** and configure the output to the **Interface Health Summary** attribute
- Step 6, select **Periodic** scheduling and schedule for **1 minute frequency**



- Step 7, once your Analysis is created and tested check it in

  This rollup will now find the lowest state from Device Status – Health, Heartbeat – Health and IO – Health. If any of the health attributes are Unhealthy the overall status will be Unhealthy.

  Finally, we will capture with Event Frames when we identify Unhealthy status.

- Step 8, create a new Analysis Template, select the Event-Frame Generation analysis type and name it PI Interface – Device Status Unhealthy
- Step 9, select the Event Frame Template PI Interface Unhealthy
- Step 10, in the **StartTrigger1** Expression enter:

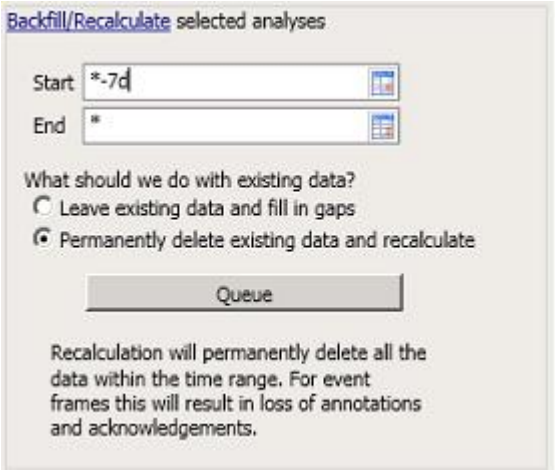
  'Device Status - Health' = "Unhealthy"

- Step 11, **Event-Triggered** scheduling is appropriate as we need to catch all Unhealthy events
- Step 12, once your Analysis is created and tested and check it in
- Step 13, repeat Steps 8 – 12 substituting Device Status for Heartbeat

Tip	You can copy and paste PI Analyses and edit them to save time

- Step 14, repeat Steps 8 – 12 substituting Device Status for IO Rate

- Step 15, in the **Management** panel you can check if the analyses created are **Enabled**. **Select all analyses** and use the **backfill/recalculate** operation to create some history. For example, backfill for 7 days.



8. Directed Activity - Visualization

8.1 Visualizing Health Detail and Overview

With the simple PI AF Hierarchy in place with the AF Element Template created the framework is there to expand and build out a complete hierarchy to ascertain the health of the interfaces in our PI environment.

PI Vision gives the opportunity to create dashboards that will give us an overview of the overall health of the PI Interfaces.

The instructor will give some examples of using PI Vision to create useful dashboards with relevant information driven by the AF Element Template and PI AF Hierarchy.

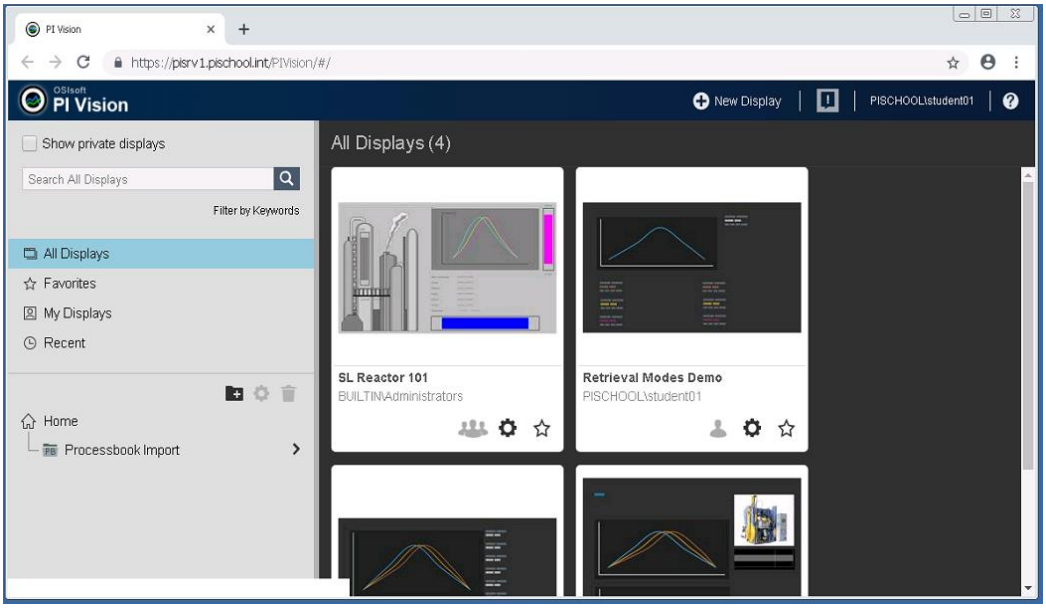
Use this time to create a dashboard.

8.2 Create a PI Vision Display

- Open PI Vision
- Create a New Display

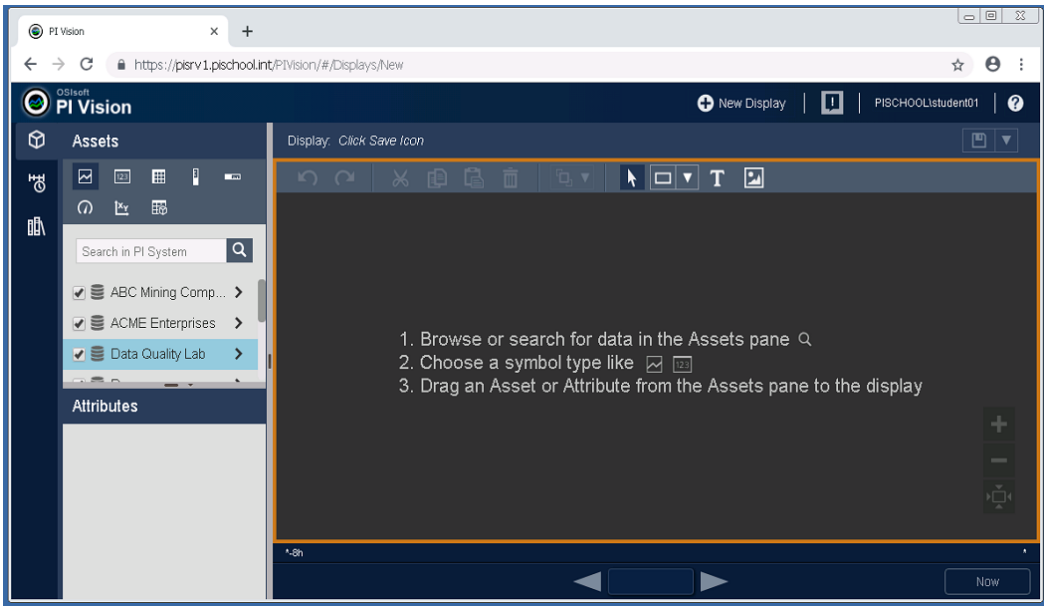
8.3 Create a PI Vision Display – Step by Step

- Step 1, on the **Desktop** of your virtual machine **double-click** on the **PI Vision icon**.



- Step 2, click the New Display

- Step 3, browse to the Data Quality Lab PI AF database



- Step 4, be creative and create your own dashboard

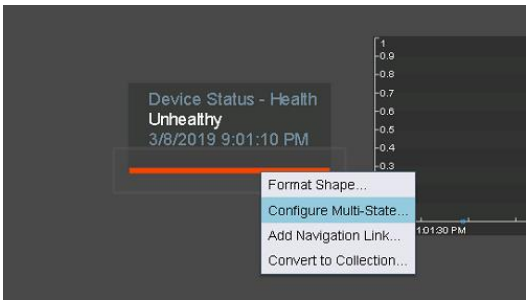
8.4 Concepts for Visualization

Basics

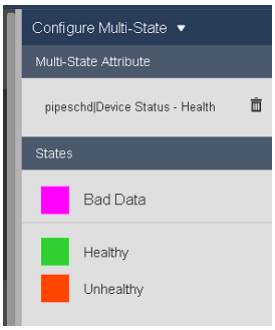
For the purpose of this lab, it is assumed that participants are familiar with the drag-and-drop functionality of PI Vision. Select an attribute, corresponding widget, and drag to the display pane. Each display element may have different configuration possibilities per its design.

Multistate

Create multistate visual indicators for data quality states. First, right-click on the display item and click “Configure Multi-State”.

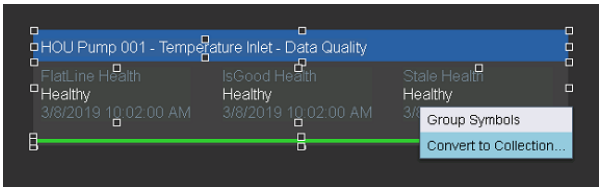


Drag-and-drop a key variable into the configuration window and identify the color scheme for each limit.

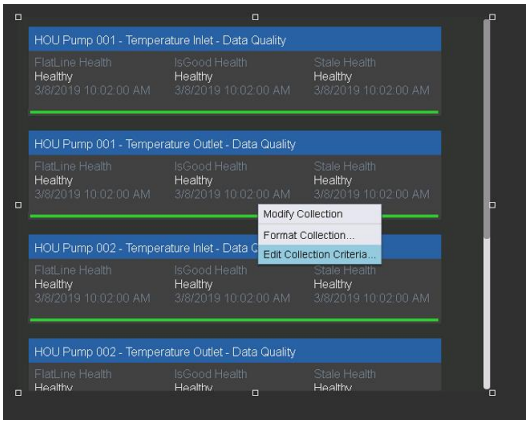


Collections

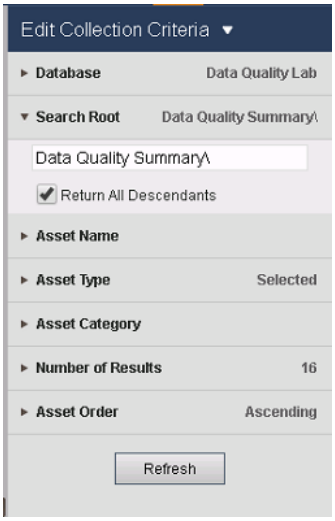
Collections are useful when developing summary pages to view a “collection” of data from elements using the same template. Build an element-specific group of data objects, select, right-click, and select “Convert to Collection”.



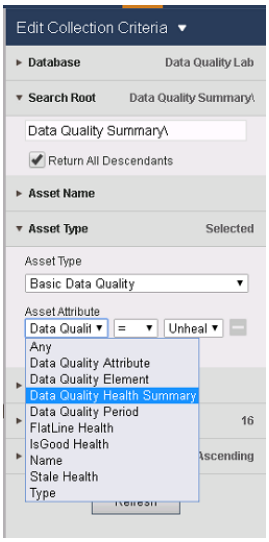
The collection will build around the element path and template used. Once the collection is designed, it is now possible to edit the search criteria used to collect the data. For example, instead of displaying only Houston-area pumps, display all pumps. This can be done by right-clicking on the collection and selecting “Edit Collection Criteria”.



Once in the configuration window, edit the Search Root for the desired element tree. To expand to children elements, select “Return All Descendants” and click “Refresh”.



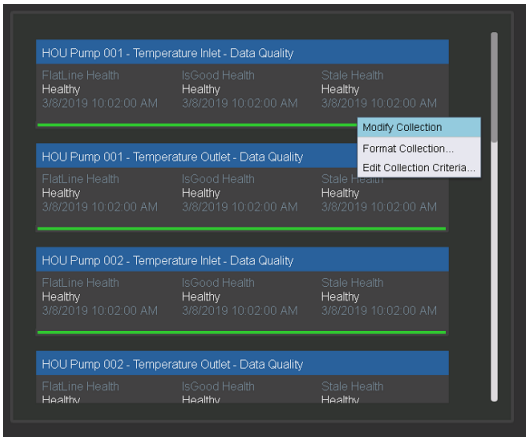
In the case of data quality, it might be useful to filter on “Unhealthy” data. In the configuration pane, edit the Asset Type for the template used select the attribute, “Data Health Quality Summary” and set equal to “Unhealthy”. Click “Refresh”. The collection will now show only members where this is true.



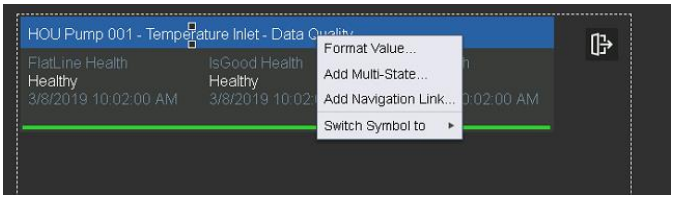
Explore other capabilities within collection configuration.

Navigation

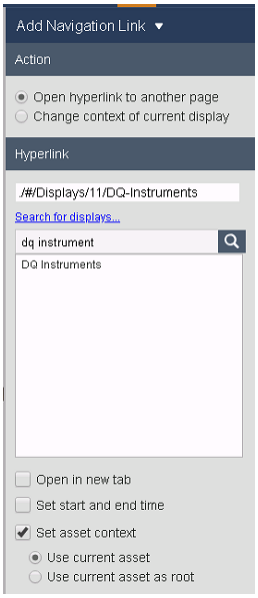
Once a summary collection is built, it would be useful to navigate a more detailed display based on the items displayed. Right-click on the collection, and select “Modify Collection”.



Select a data element to add navigation and right-click, selecting “Add Navigation Link”.



The navigation configuration can be used to open a separate display or to change the context of the existing one.



9. Directed Activity – Basic Data Quality Checks

9.1 Identifying Common Data Quality Issues

With Interface Health Points we were able to identify incidents with:

- The interface service by monitoring the interface Heartbeat.
- The data source by monitoring the Device Status.
- The data flow by monitoring the IO Rate.

These are all good indicators of data collection and data flow, but do not make any analysis of the data itself being collected and stored.

In this section we will apply PI Analyses to perform basic data quality checks to monitor for when critical data is:

- Stale – the data has stopped updating, there are no new values for a defined period.
- Flat-line – the data is updating but the same value is repeated for a defined period.
- Bad – the data is bad as indicated by a System State value.

For further information about System States, see:

https://livelibrary.osisoft.com/LiveLibrary/content/en/server-v11/GUID-7F11EB89-A874-4628-A772-0CA90B7EF716

In this example, we have a company that has many chemical pumps where it is critical to measure the temperature difference between the inlet and outlet. With these attributes defined as our critical measurements we will build analyses that will allow us to ensure that the quality of the data is good.

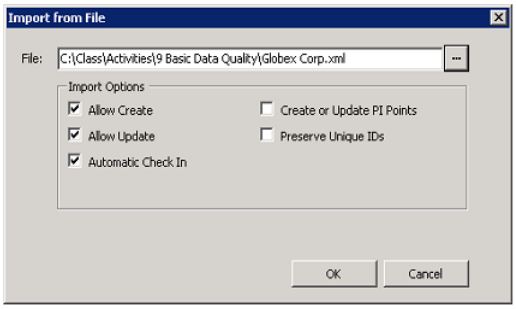
9.2 Catching Stale, Flat-Line and Bad Data

- Import the example PI AF hierarchy and templates
- Create PI Analysis to identify stale, flat-line and bad data
- Deploy elements to check data quality for the pumps

9.3 Catching Stale, Flat-Line and Bad Data – Step by Step

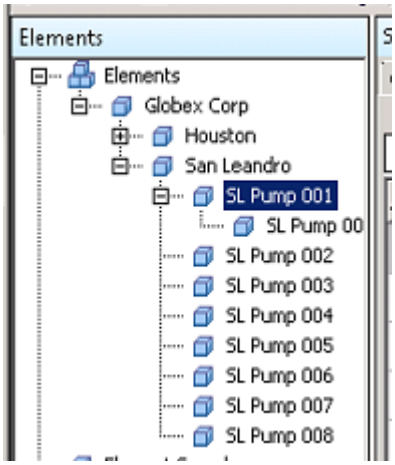
- Step 1, in **PI System Explorer** select **File** from the menu and select the option **Import from File …**

- Step 2, select the xml file **Globex Corp.xml**
- Step 3, retain the default checkbox selection (Allow Create, Allow Update, Automatic Check In) and **click OK**



- Step 4, once the process is complete **click Close**

In the PI AF Hierarchy you should now have a structure with pumps for San Leandro and for Houston



- Step 5, in the **Library** select the newly imported Element Template **Basic Data Quality**

This template includes the following attributes, which we will use:

Attribute Name	Purpose
Data Quality Period	**Both stale and flat-line analyses have a time dimension, e.g. has the data been stale for 1 hour? This attribute defines this period.**

Data Quality Attribute	**To make the Elements dynamic this attribute is a relative attribute path identifying the data to be analyzed.**
Value	**Using the Data Quality Attribute the Value attribute references the data for qualification.**

- Step 6, create a new analysis named Basic Data Quality

  The following are suggested expressions to identify if the Value is Stale, Flat-Lined or Bad for the period specified.

Name	Expression	Output Attribute	
V	`// Data Quality Target` `'Value'`	Map	⊗
P	`'Data Quality Period'`	Map	⊗
ET	`// End Time for Analysis` `'*'`	Map	⊗
ST	`// Start Time for Analysis` `ET - Convert(P, "s")`	Map	⊗

Stale	`// If the value is updating it is healthy`
	`If PrevEvent(V, ET) < ST THen`
	`  "Unhealthy"`
	`Else`
	`  "Healthy"`
		Stale Health	⊗

FlatLine	`// If the value is changing it is healthy`
	`If Stale = "Unhealthy" Then`
	`  // if it has not updated it must be flat`
	`  "Unhealthy"`
	`Else`
	`  If StDev(V, St, ET, 0) = 0 Then "Unhealthy" Else "Healthy"`
		FlatLine Health	⊗

Bad	`// If the value has been good during the period is is healthy` `If PctGood(V, ST, ET) = 0 Then "Unhealthy" Else "Healthy"`	IsGood Health	⊗

  Remember to map the Output Attributes

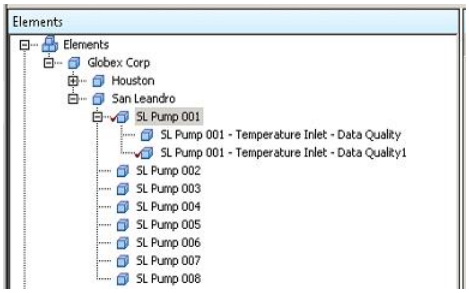
- Step 7, for this PI Analysis we schedule to evaluate **every minute**
- Step 8, once your Analysis is created and tested check it in

  Now we have an Element Template that can be used to create Elements to check data quality of our critical data.

  To quickly add a check for the Outlet Temperature for the SL Pump 001:

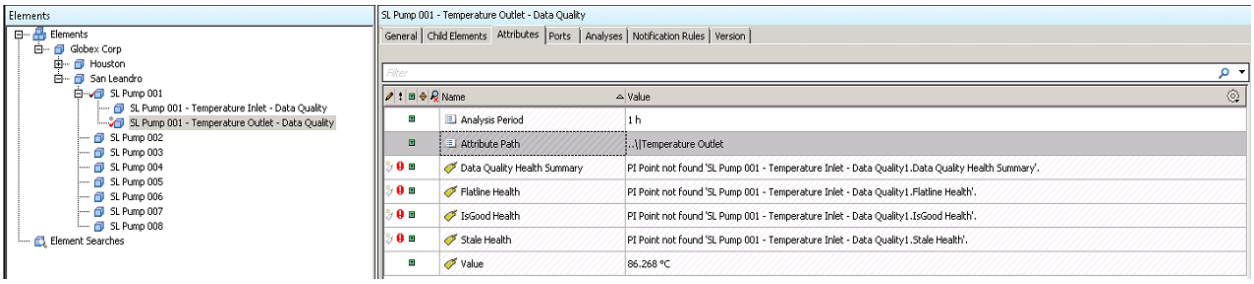
- Step 9, in the element hierarchy browse to the **SL Pump 001 – Temperature Inlet – Data Quality** element. (It is a child of Globex Corp | San Leandro | SL Pump 001)

- Step 10, Copy & Paste the SL Pump 001 – Temperature Inlet – Data Quality as a new child element under SL Pump 001

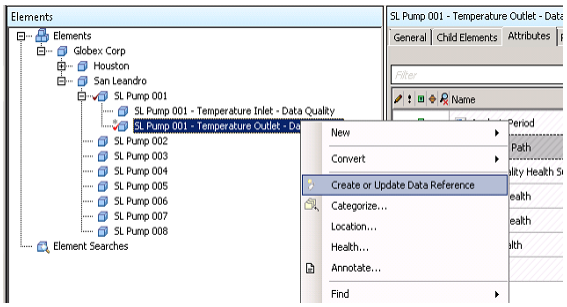


- Step 11, rename the newly created element as SL Pump 001 – Temperature Outlet – Data Quality

- Step 12, change the value of the attribute **Data Quality Attribute** to **Temperature Outlet**



- Step 13, check in then right-click on the element SL Pump 001 – Temperature Outlet – Data Quality and select Create or Update Data Reference



Steps 9-13 can be repeated to add Inlet and Outlet Temperature checks for any other critical pumps in the hierarchy. By using a child element and relative references to attributes we have the flexibility to apply the data quality checks where needed.

- Step 14, add some more child data quality checks in San Leandro and Houston.

10. Directed Activity – Data Quality Summary Hierarchy

10.1 Summarizing Data for Drill-Down Analysis

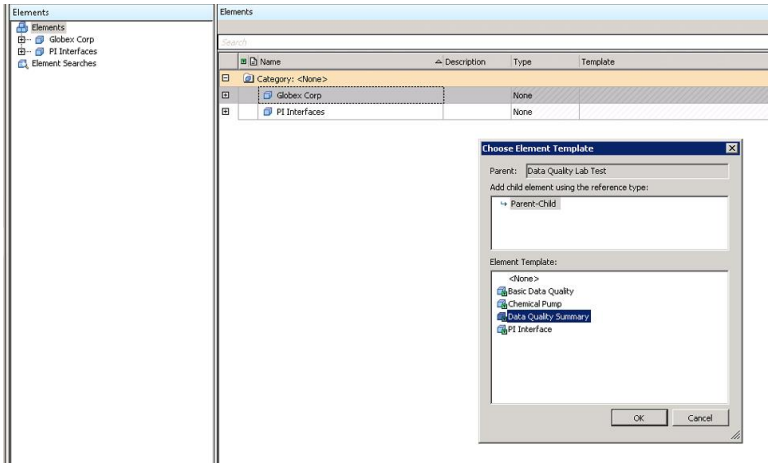
We now have built into our hierarchy elements that can check data quality. It may be useful to create a complimentary hierarchy specifically for data quality. This hierarchy will allow us to rollup data quality health so that we can have summary health, bad signal counts, etc. Creating these summary hierarchies are also useful for visualization – providing the summary values required for drill-down analysis.

10.2 Create a Roll-Up Hierarchy

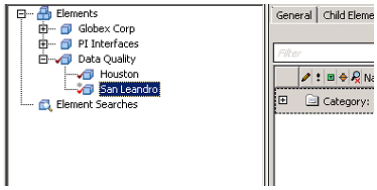
- Create reference hierarchy with rollups
- Populate with referenced elements

10.3 Create a Roll-Up Hierarchy – Step by Step

- Step 1, create a root element with the **Data Quality Health Summary** element template. Name this **Data Quality**.



- Step 2, create two child elements one named **Houston** and one named **San Leandro**, again using the **Data Quality Summary** element template.

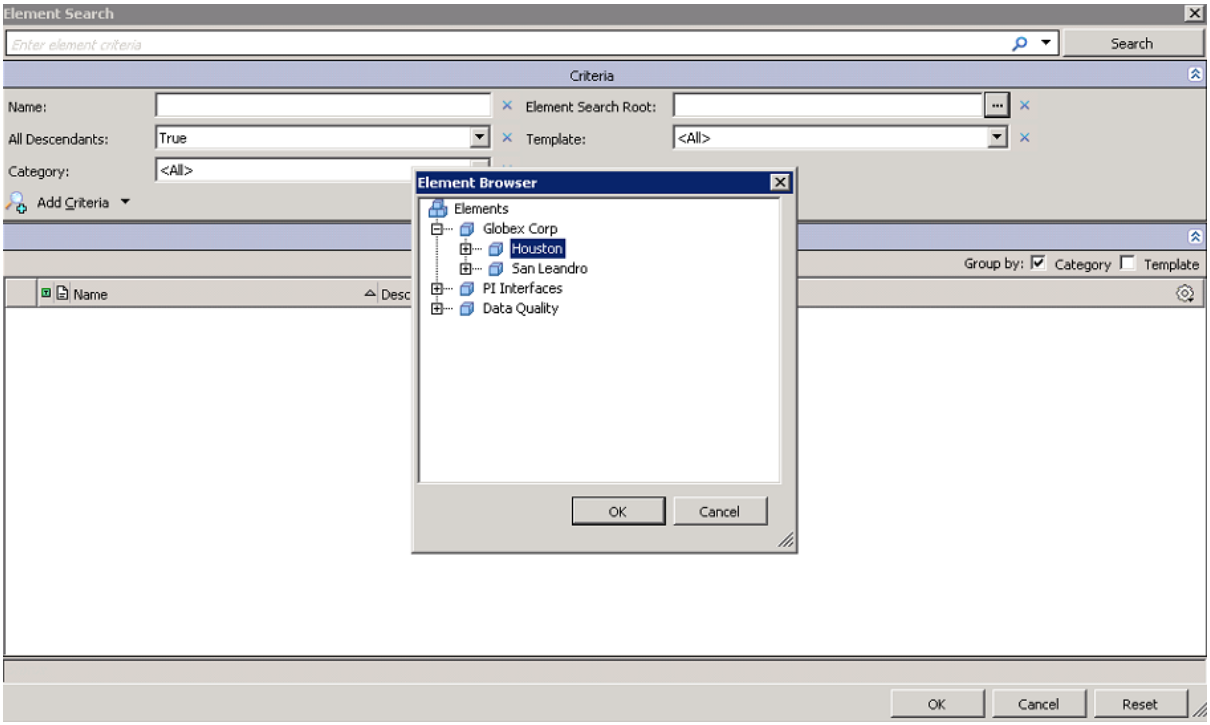


- Step 3, right-click on the Data Quality element and select Create or Update Data Reference

  With the hierarchy in place, rather than duplicating our Basic Data Quality elements we should use referenced elements to be summarized. This can be done in a number of ways – a quick way is to use Element Search.

- Step 4, click **Search** on the toolbar and select **Element Search**, or press the **F3** key

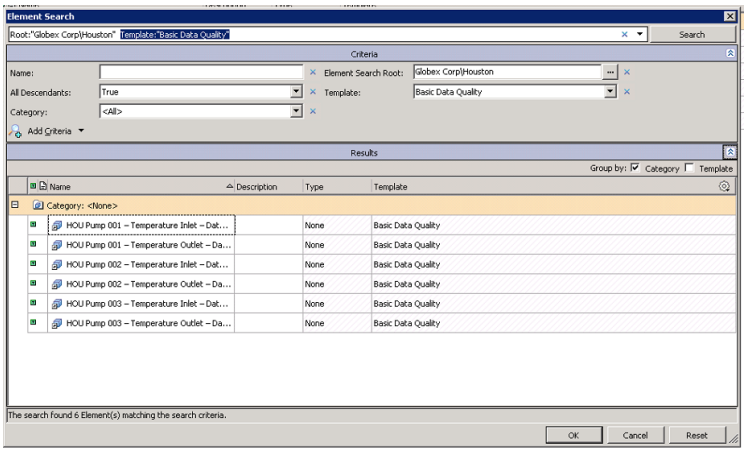
- Step 5, in the Search dialog select the Element Search Root ellipsis button. Select Houston in the Globex Corp hierarchy and click OK



- Step 6, change the **All Descendants** dropdown to **True**

- Step 7, select Basic Data Quality from the Template dropdown

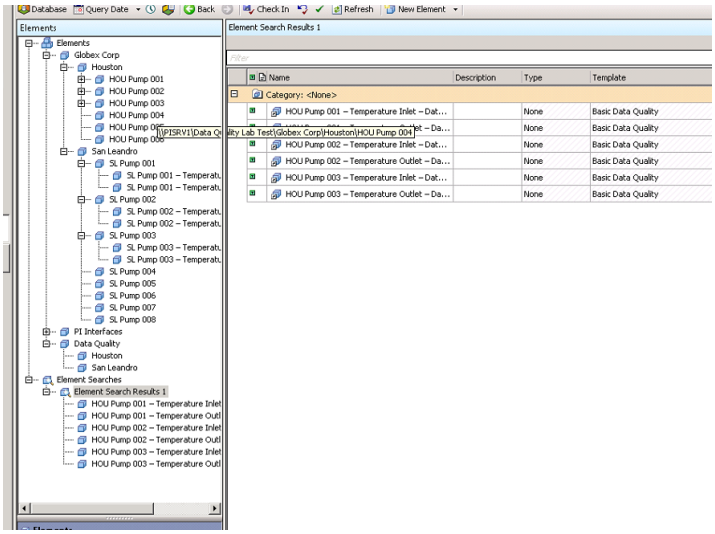
- Step 8, click the **Search** button



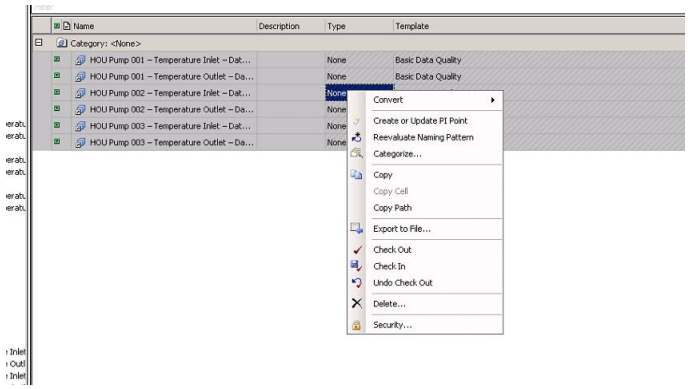
Your search results may differ from the example depending on how many Basic Data Quality elements you created.

- Step 9, Click **OK**

- Step 10, in the **Element Searches** section of the **Elements browser** you should now see the newly created search. **Select** this search.

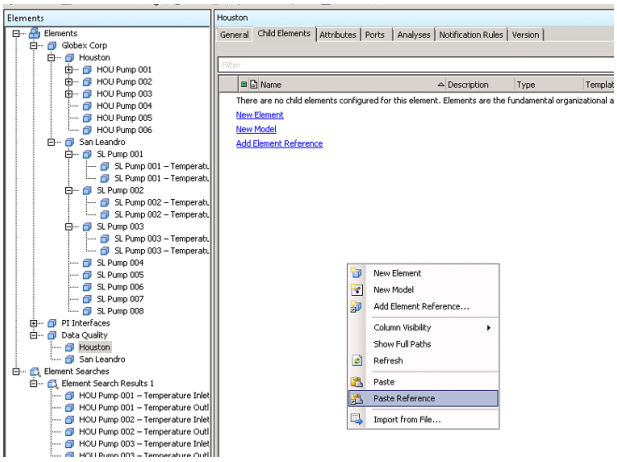


- Step 11, select all the search results and right-click then select Copy



Tip	Pressing Ctrl-A is a shortcut for Select All when selecting objects

- Step 12, select the Houston element in the Data Quality hierarchy and in the Child Elements pane right-click and select Paste Reference. (A weak reference type is appropriate for this hierarchy).



- Step 13, under the Houston hierarchy there will now be references to all the Houston Data Quality elements. **Repeat steps 4-12 for San Leandro**.

The Data Quality hierarchy now has the Data Quality checks that were created and will summarize, using Rollup, if any of the child elements contain an Unhealthy status.

11. Directed Activity – Adding Flexibility with Derived Templates

11.1 Vary the Functionality of Data Quality Templates

The Basic Data Quality template checks three of the common definitions of data quality for time-series data, but there are more definitions that can be used in addition.

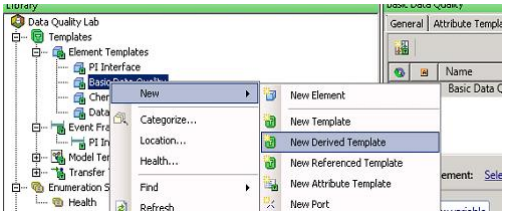
Using Derived templates we can retain the Basic Data Quality configuration but expand to include more specific data quality tests – such as data out of range.

11.2 Adding Out of Range Check

- Create Derived Template from Basic Data Quality
- Add Out of Range Health Attribute
- Add Out of Range analysis

11.3 Adding Out of Range Check – Step by Step

- Step 1, in the Library right-click on the Basic Data Quality Element Template and select New -> New Derived Template



- Step 2, rename the new Element Template **Basic Data Quality + Out of Range**
- Step 3, create an **Attribute template** named **Out of Range Health** – this is easiest achieved by copying and pasting one of the existing Health attributes such as Stale Health

Tip	If you cannot see any Attribute Templates click the **Group by: Template** checkbox

- Step 4, create a **new analysis** named **Out of Range**

  The following are suggested expressions for identifying out of range data:

Name	Expression	Value at Evaluati	Value at Last Tri	Output Attribute	
V	// Data Quality Target 'Value'			Map	⊗
P	'Data Quality Period'			Map	⊗
ET	// End Time for Analysis '*'			Map	⊗
ST	// Start Time for Analysis ET - Convert(P, "s")			Map	⊗
Zero	TagZero(V)			Map	⊗
Span	TagSpan(V)			Map	⊗

OOR	// If the value is within its range during the perid it is healthy
	If TagMin(V, ST, ET) < Zero Then
	  // value was below zero
	  "Unhealthy"
	Else
	  If TagMax(V, ST, ET) > (Zero + Span) Then
	  // value was above range
	  "Unhealthy"
	  Else
	  "Healthy"
				Out of Range Health	⊗

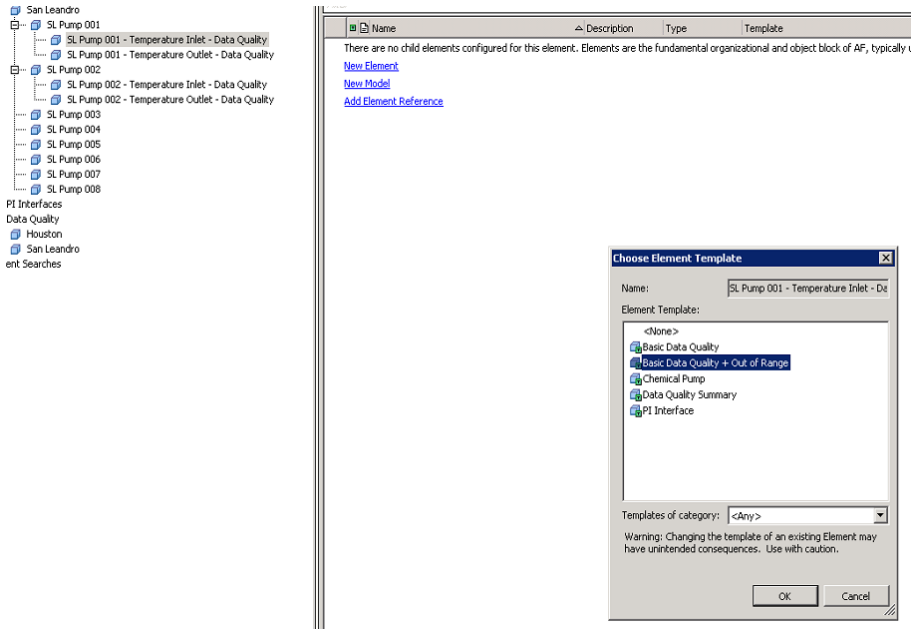
  Remember to map the Output Attribute

- Step 5, for this PI Analysis we schedule to evaluate **every minute**

- Step 6, to test the analysis you will need to change one of the existing Basic Data Quality elements to use this template.

  In the **Elements** hierarchy **right-click** on one of the Data Quality elements and select **Convert -> Change Template…**

  Select Basic Data Quality + Out of Range from the list and click OK



You can now return to the analysis in the template and use the **Select an example element** to use this element to test against.

- Step 7, once your Analysis is created and tested and **check it in**

Using this technique of expanding the capabilities of Element Templates by Derived templates allows for a variety of further checks to be made in addition to core basic quality checks. The functions in Asset Analytics support many other analyses to be made on data. Consider what checks are required for your data quality and how they can be achieved with Asset Analytics.


Have an idea how to improve our products?
OSIsoft wants to hear from you!
https://feedback.osisoft.com/

Save the Date!

OSIsoft PI World Users Conference in Gothenburg, Sweden. September 16-19, 2019.

Register your interest now to receive updates and notification early bird registration opening.

https://pages.osisoft.com/UC-EMEA-Q3-19-PIWorldGBG-RegisterYourInterest_RegisterYourInterest-LP.html?_ga=2.20661553.86037572.1539782043-591736536.1533567354