OSIsoft. WIPRO Applying Thought

vCampus Live! 2012

Custom PI AF Data Reference Master Class

Inheriting from a PI Point DR

A Case Study of a Real-time Calculation Engine

Presented by **Zev Arnold**
Peter Jackson

Presenters

- Zev Arnold
  - PI project and support work for several large international O&G companies including Shell, BP, and Chevron.
  - Consultant with Wipro ENU Consulting Practice.
  - Product Manager for RtKPI application.

- Peter Jackson
  - *PI System Specialist – 14 yrs.*
  - *Developer/Architect – 30 yrs.*
  - Consultant with Wipro ENU Consulting Practice.
  - Lead developer for RtKPI application.

Overview

RtKPI – Real-time Key Performance Indicator

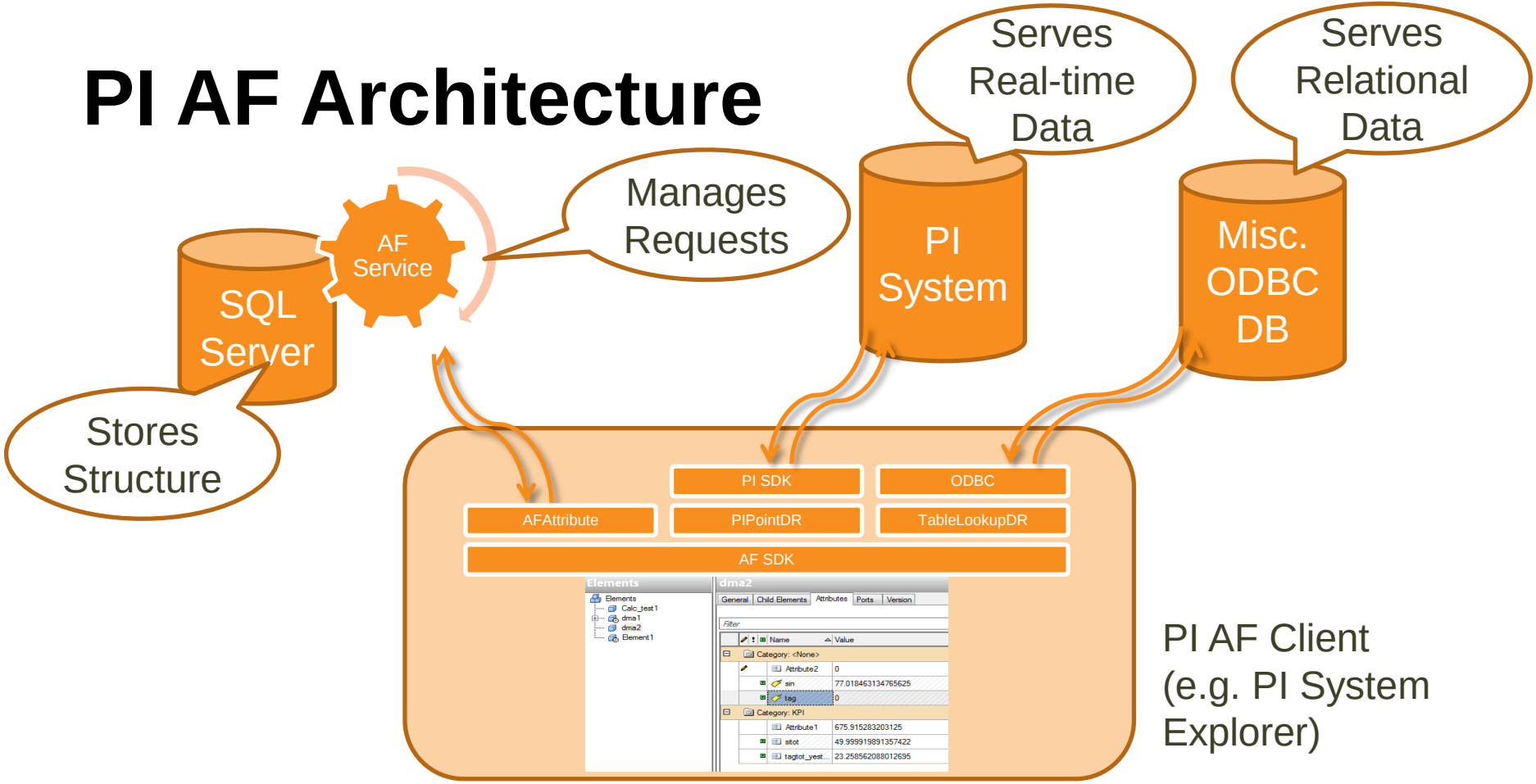
- User-configurable Calculation Engine
- Built on top of the PI System
- Developed for an international O&G company
- Dev team: 3 developers | Dev time: 4 months
- Accelerated development by inheriting from the PI Point Data Reference

"Inheritance enables you to create a new class that reuses, extends and modifies the behavior that is defined in another class." – MSDN

Agenda

- PI AF Architecture – A Review for the Rest of Us
- Custom Data References: The Basics
- RtKPI – An AF-based Calculation Engine
- “Inheriting” from PIPointDR: A Case Study
  - Reuse
  - Modify
  - Extend

PI AF Architecture



PI AF Client (e.g. PI System Explorer)

So Many Classes?

But polymorphism makes it easy…

Data Reference implementations inherit from AFDataReference.

```
public class PIPointDR : AFDataReference
{…}
public class TableLookupDR : AFDataReference
{…}
```

Each implementation overrides methods as needed, like GetValue for instance…

```
public override AFValue GetValue(object context, object timeContext, …){…}
```

Clients don't need to know what kind of Data Reference, they just call GetValue and… magic!

```
AFAttribute myAttribute = new AFAttribute(afDatabase, "\\AttributePath")
myAttribute.DataReference.GetValue(context, timeContext, inputAttributes,
inputValues);
```

But really you should use the AFAttribute GetValue method which simplifies the call…

```
myAttribute.GetValue();
```

Make Your Own

```
public class PinkElephantDR : AFDataReference
```

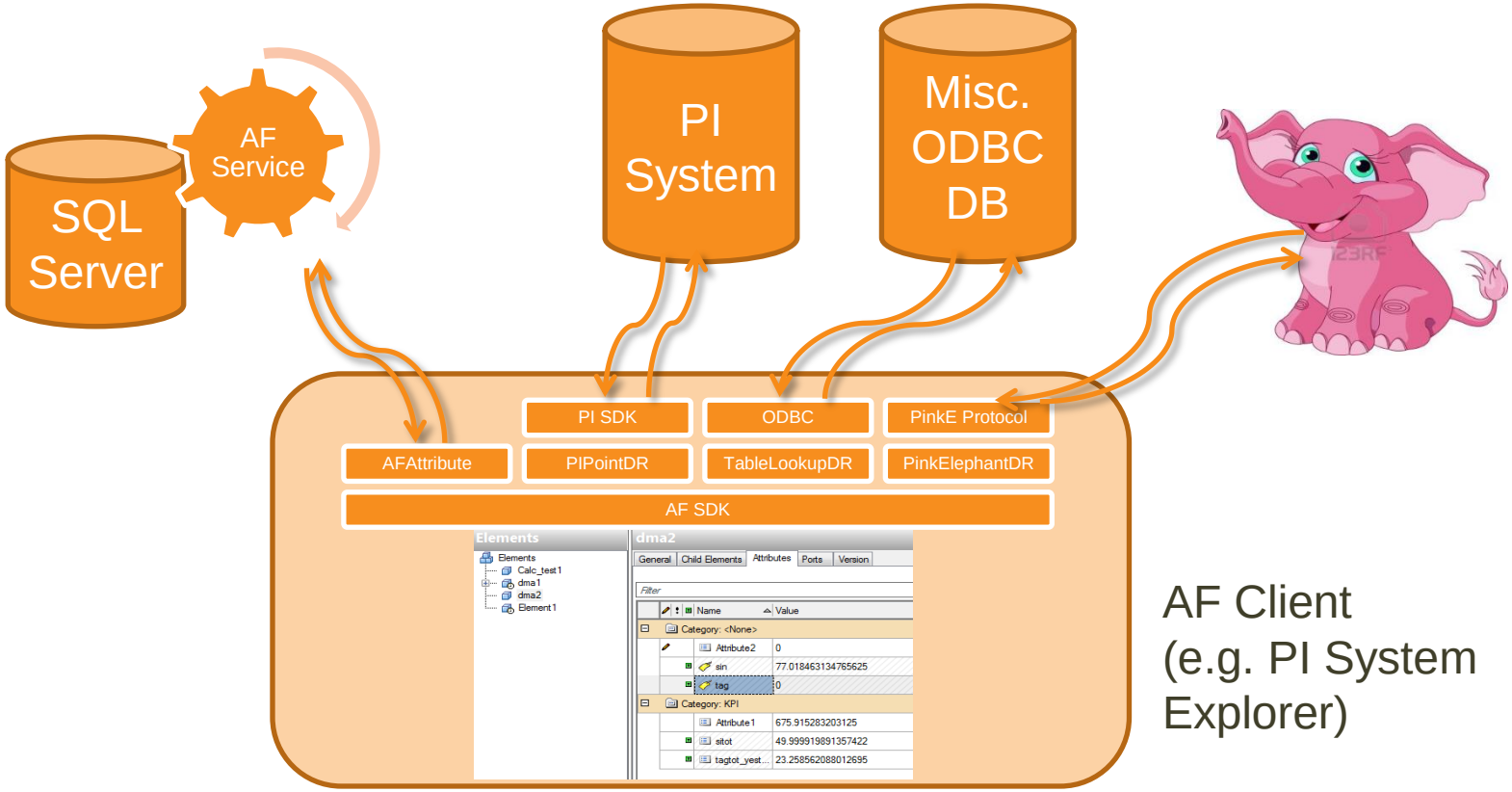
Override likely functions…

- GetValue – get a single value from the data source
- GetValues – get a range of values from the data source
- SetValue – write a value to the data source

And properties…

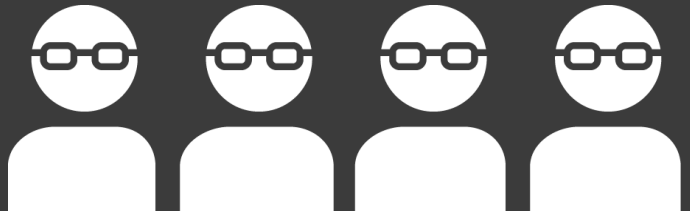
- ConfigString – store and load the configuration for the data reference

Custom PI AF Architecture



AF Client
(e.g. PI System Explorer)

RtKPI – Design and Requirements



RtKPI – Design Requirements

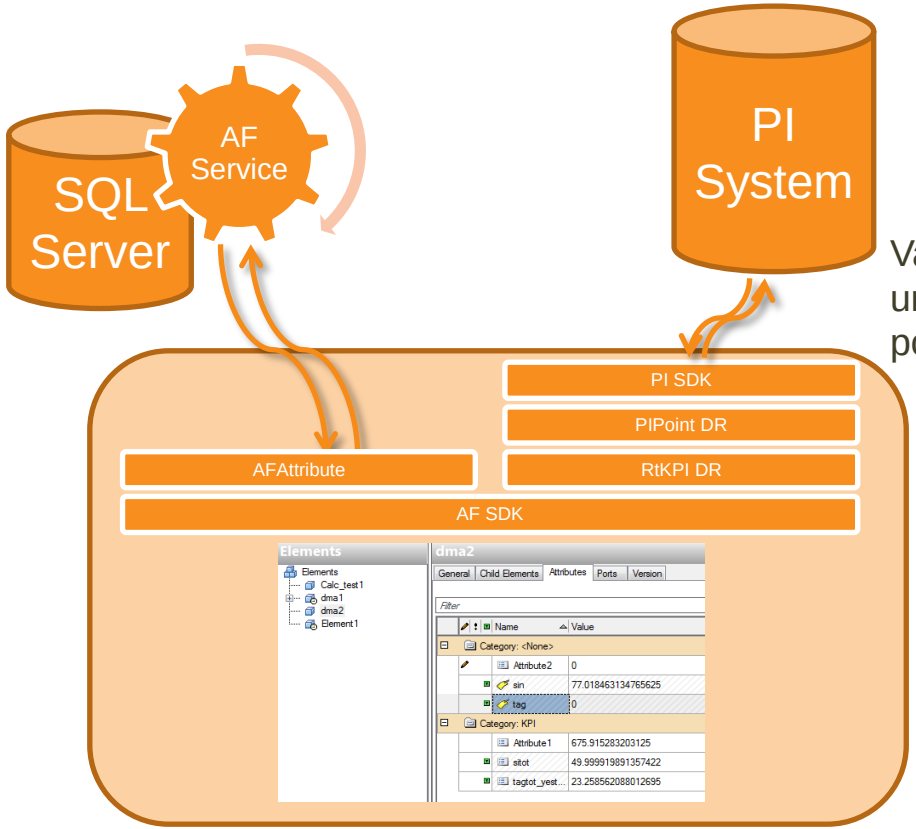
- Historize calculated values in the PI System
- Be able to configure calculations through PI System Explorer
- Minimal administration required
  - RtKPI should automatically pick up new calculations and begin historizing them
  - RtKPI should monitor the PI AF database for changes/deletions to calculations
- Allow for easy migration to Abacus

RtKPI – Functionality: RtKPI Usage

- Functionality available through PI System Explorer
  - Create a new AF Attribute in the hierarchy and make it an RtKPI Calculation Attribute
  - Configure the calculation, using references to other AF Attributes
  - See the calculated value for the RtKPI Calculation Attribute just as you would any other AF Attribute

RtKPI – Architecture: RtKPI Usage

Create/Change RtKPI Attribute Configuration

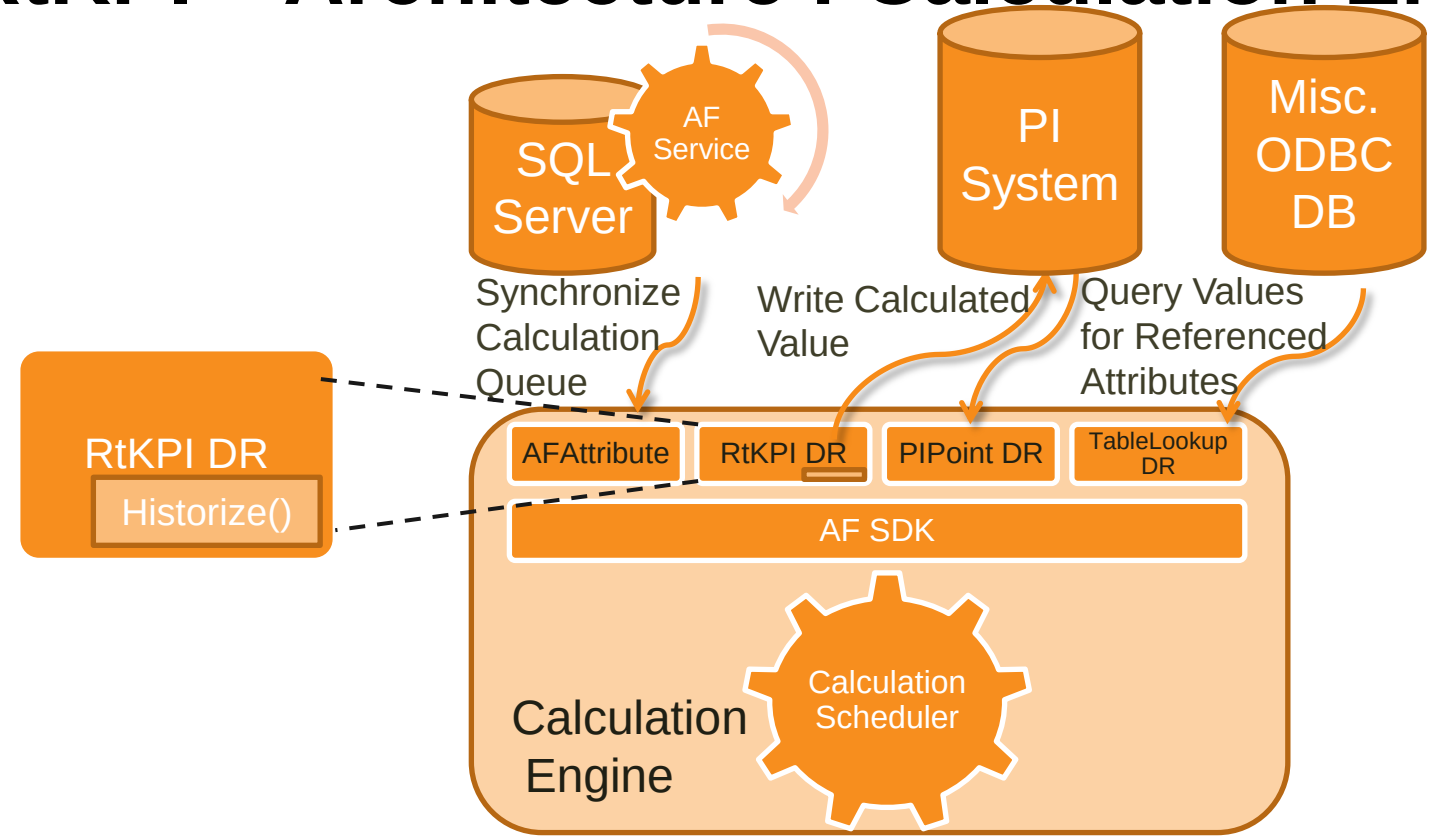


Values for the underlying PI point.

RtKPI – Functionality: Calculation Engine

- Functionality implemented in the Calculation Engine
  - Update calculation queue from the PI AF hierarchy
  - Execute periodic calculations based on RtKPI DR configuration
  - Write calculated values to the PI System

RtKPI – Architecture : Calculation Engine



But you can't just inherit from PIPointDR!

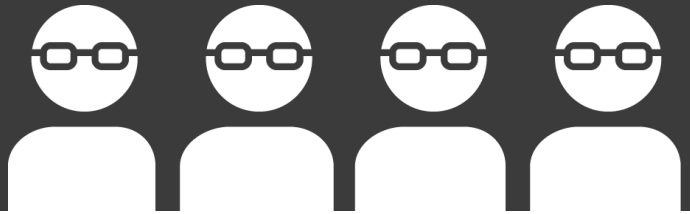
- PI DR is in a separate assembly
  - The custom DR would be dependant
  - It is not possible to register a DR as a support assembly

- DR's have a Unique GUID: a custom DR can not "be" PIPointDR.

- Links the Custom code very tightly with the OSIsoft code: will an AF SDK update "Break" the custom DR?

- "...neither Encapsulation or Inheritance of other Data References are tested scenarios."

Hacking Inheritance



Hacking Inheritance

"Inheritance enables you to create a new class that reuses, extends and modifies the behavior that is defined in another class." – MSDN

- How do we reuse PIPointDR functionality?
- How do we modify PIPointDR functionality?
- How do we extend PIPointDR functionality?

Reuse PIPointDR Functionality

```
public static AFAttribute FindAttribute(string path,
AFObject relativeFrom){...}
```

If “path” is the DataReference name followed by its configuration string (e.g. '*DataReferenceName.ConfigString*'), you can *dynamically* create an attribute at runtime

For PIPoint attributes just the Tag is enough

e.g. “\\piserver\sinusoid”

Code

```
public override string ConfigString
{ …
    internalPiAttr =
AFAttribute.FindAttribute(TagName+";ReadOnly=False", this.Database);
 }
```

Once you have a reference to a PIPoint attribute, you can reuse its functionality in your overridden function.

```
public override AFValues GetValues(Object context….)
{
        return internalPiAttr.DataReference.GetValues(context, …);
}
```

Modify PIPointDR Functionality

- Modify PIPointDR functionality in overriden functions

```
public override AFValues GetValues(Object context….)
{
    AFValues afvals= internalPiAttr.DataReference.GetValues(
                                context,… );
            for (int i=0; i< afvals.Count;i++)
            {
                afvals[i] = checkValue(afvals[i], null);
            }
}
```

Modify PIPointDR Functionality

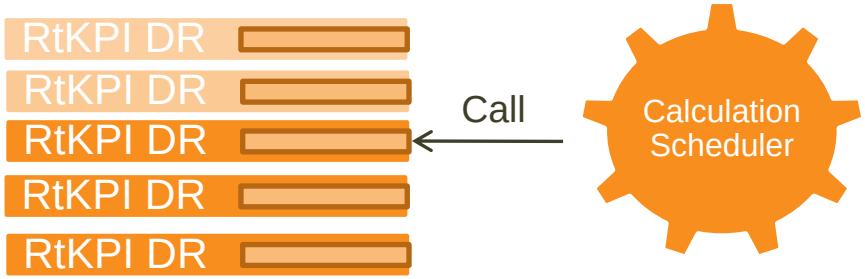
Other likely candidates:

- ConfigString
  - Required for any CustomDR that needs configuration
- RawPIPoint
  - allows client access to underlying PI-tag (if any)
  - Enables “use PI Point directly” in Processbook
- Zero, Span, Step
  - Same function as similarly name PI tag attributes
  - Zero, Span can be used to set trend scales in Processbook

Extend PIPointDR Functionality

We extend the PIPointDR by adding the Historize function to the interface.

This allows us to decouple the calculation scheduler from the implementation details of calculating a value.

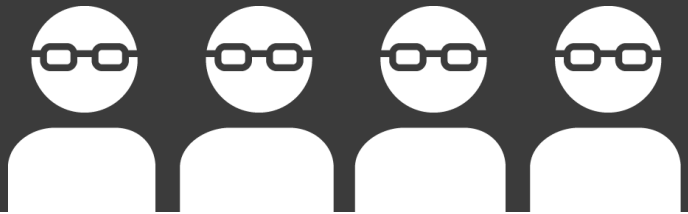


Extend PIPointDR Functionality

- AF SDK client applications cannot know about non-standard methods. They just get an AFDataReference object.
- We use Reflection to get a handle to the function that we're interested in. Reflection lets us expand the interface.

```
AFDataReference RtKPIDR = RtKPIAttribute.DataReference;
Type dt_MyDataReference = RtKPIDR.GetType();
//get the "historize" function
MethodInfo dt_MyExtraFunction = dt_MyDataReference.GetMethod("Historize",new Type[0]);
//invoke it
dt_MyExtraFunction.Invoke(RtKPIDR, null);
```

Future – PI AF 2012 and RDA



Using RDA in PI AF 2012

- With an “Embedded” PIPoint Attribute, Rich Data Access can be used
  - Direct access to PI data:
  - Eliminate use of PI-SDK (and COM)

AFAttribute.Data Property

Provides methods to Access:

- Recorded Values
- Interpolated Values
- Summary Data

AFAttribute.Data Property

- Summary(timeRange, summaryType, ...)
  - Total
  - Average
  - Minimum
  - Maximum
  - StdDev

AfCalculation

- **AFCalculation** class provides mechanisms to evaluate expressions.
- Follows PI Performance Equation syntax.
- Expression variables are references to Attributes or PI Points

AFCalculation

- Can use instead of `IPICalculation`

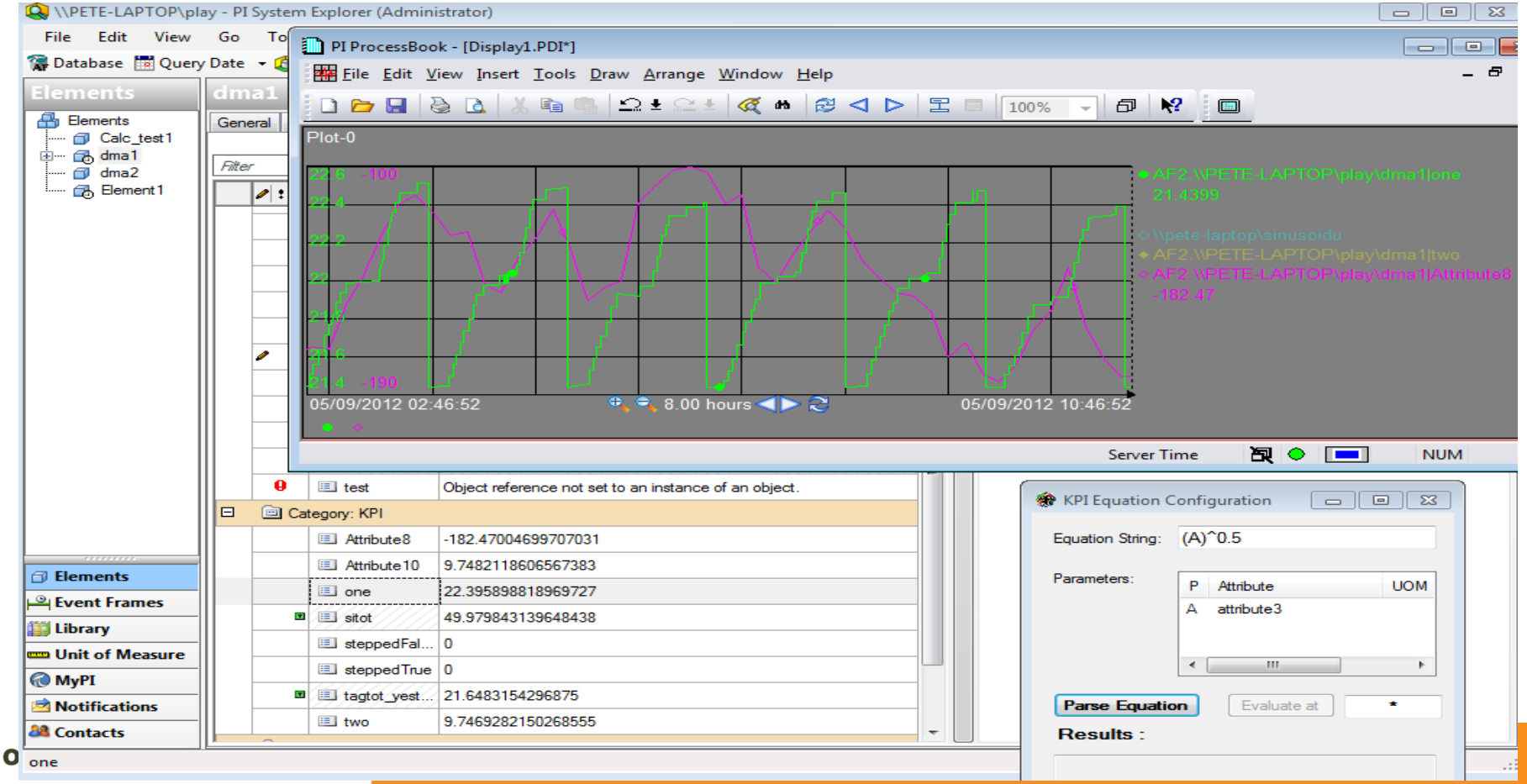
```
AFValues val =
      AFCalculation.CalculateAtTimes(_at.PIPoint.Server,
                        PEbuilder(_op, _at.PIPoint.Name,
                                    _start, _end, value.Value),
                        times);
```

PI ProcessBook and .net 4.0

- RDA *requires* .Net 4.0
- Applications linked with earlier version will not load .Net 4.0 dlls - PI ProcessBook
- Work around:

  add to Processbook.exe.config

```
<startup useLegacyV2RuntimeActivationPolicy="true">
     <supportedRuntime version="v4.0.30319"/>
     <supportedRuntime version="v2.0.50727"/>
</startup>
```


\\PETE-LAPTOP\play - PI System Explorer (Administrator)
File Edit View Go To
Database Query Date
Elements
Elements
Calc_test1
dma1
dma2
Element1
dma1
General
Filter
PI ProcessBook - [Display1.PDI*]
File Edit View Insert Tools Draw Arrange Window Help
100%
Plot-0
AF2 \\PETE-LAPTOP\play\dma1|one
21.4399
\\pete-laptop\sinusoidu
AF2 \\PETE-LAPTOP\play\dma1|two
AF2 \\PETE-LAPTOP\play\dma1|Attribute8
-182.47
05/09/2012 02:46:52
8.00 hours
05/09/2012 10:46:52
Server Time
NUM
test
Object reference not set to an instance of an object.
Category: KPI
Attribute8
-182.47004699707031
Attribute10
9.7482118606567383
one
22.395898818969727
sitot
49.979843139648438
steppedFal...
0
steppedTrue
0
tagtot_yest...
21.6483154296875
two
9.7469282150268555
KPI Equation Configuration
Equation String:
(A)^0.5
Parameters:
P
Attribute
UOM
A
attribute3
Parse Equation
Evaluate at
Results :
Elements
Event Frames
Library
Unit of Measure
MyPI
Notifications
Contacts
one

WIPRO
Applying Thought

OSIsoft®

THANK YOU