OSIsoft.
PI World
SAN FRANCISCO
MARCH 23-26, 2020

PI World 2020 Lab

AF SDK Project Management Best Practices

Published: April 30, 2020

Contents

1. Introduction

1.1 Background

The sheer size of a large global company poses many challenges when it comes to maintaining the company's enterprise software including the PI system. Typically, millions of physical assets, thousands of PI users/servers, and billions of PI data streams are involved with the company's daily operation. Understandably, a comprehensive strategy that can handle large-scale problems is required for managing such a complex PI landscape.

To execute this type of enterprise-wide strategy, our large enterprise customers architect company's data infrastructure using the PI AF and AF SDK. This allows data-driven solutions such as calculations, displays and reports to be templated and replicated quickly across the enterprise.

One typical example of a data infrastructure that can ensure enterprise-wide data consistency is shown schematically below in Figure 1. (This lab will use a simplified version of this example for the exercises). First, the assets are modeled into a central AF hierarchy at the corporate level (e.g. the headquarter); next, AF objects (This lab will focus on the AF template) in the central AF hierarchy are mapped/synced to the fit-for-purpose AF hierarchy at the local sites (e.g. the satellite sites) with a custom AF SDK application. While the central AF hierarchy provides the single point of truth, facilitates standardization and enterprise-wide collaboration; the fit-for-purpose local AF server provides flexibility and helps user to perform data navigation more easily and locally.

This lab aims to provide you with a general project management tool to help your team to make plans for and execute your AF SDK projects. So that your project is in line with your company's enterprise infrastructure management strategy.

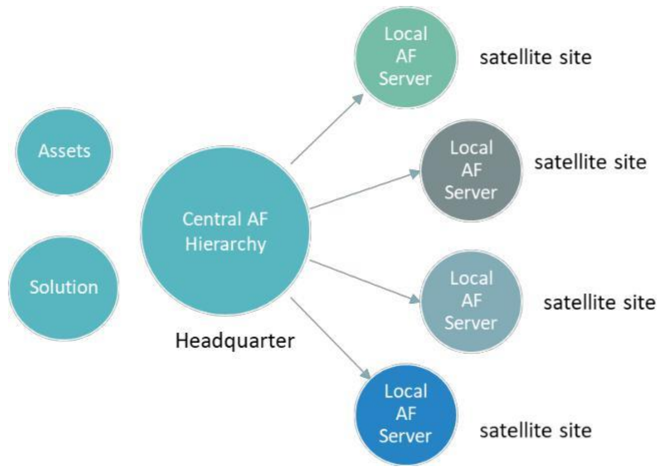


Figure 1: an example of a data infrastructure that can be used to ensure enterprise-wide data consistency.

1.2 Tasks

First, we will help you to get familiar with some of the general project management concepts and present you with our 3-Step Model Project Management Tool. Next, we will guide you to use these concepts and tools to work on a real-world project i.e. a PI AF SDK project:

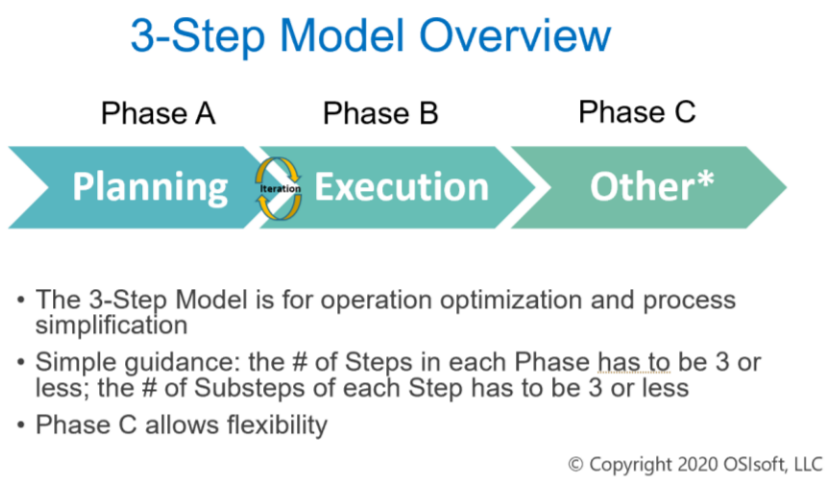
- Project Planning Tasks
- Project Execution Tasks

You will be divided into teams and will take turns to serve as the Project Manager. If time permits, we will show you how you may integrate your 3-Step Model with our world-class OSIsoft Technical Support to optimize your code.

1.3 Breakdown of Lab Content

Your objectives in this section are the following.

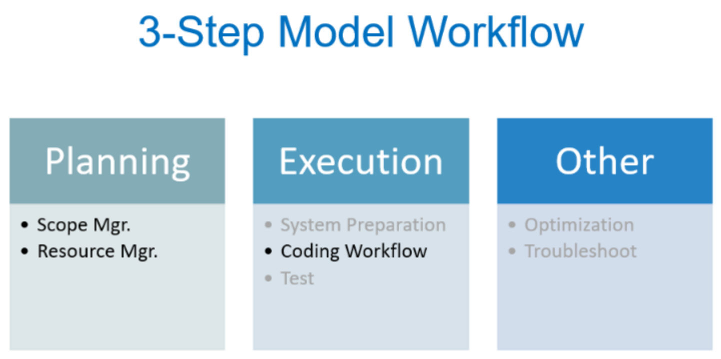
Get familiar with the 3-Step Project Management Model



3-Step Model Overview

- The 3-Step Model is for operation optimization and process simplification
- Simple guidance: the # of Steps in each Phase has to be 3 or less; the # of Substeps of each Step has to be 3 or less
- Phase C allows flexibility

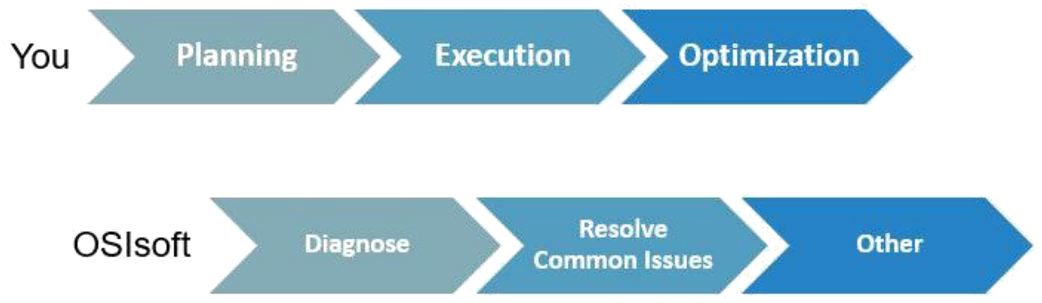
© Copyright 2020 OSIsoft, LLC



3-Step Model Workflow

Planning	Execution	Other
• Scope Mgr. • Resource Mgr.	• System Preparation • Coding Workflow • Test	• Optimization • Troubleshoot

Plan/Troubleshoot/Optimize with OSIsoft



Phase A Project Planning Tasks. These tasks aim to give you a taste of what AF SDK project planning is and help you to address the following issues:

- What are the high-level requirements?
- What is the specific problem you try to solve?
- What resource do you have for the “must haves” and “nice to haves”?

Phase B Project Execution Tasks. These tasks aim to help you to design and utilize a 3-Step workflow to construct deliverables i.e. implement the project scope to your AF SDK code:

- General Information about AF Templates
- Design & Use the Coding Workflow

2. Organizing Work Group

2.1 Objective

- Create groups that will work on the activities together
- Develop an environment of collaboration to make sure everyone in the group can participate and promote agile decision making

2.2 Instructions

All attendees will be divided in groups of 4 (or less). Each participant will be given a number: 1, 2, 3, and 4.

Follow the directions of the facilitator to find your seating arrangement. Everyone may be asked to change groups at any time based on the type of activities being performed, the goals of the participants and the length of the workshop.

2.3 Task - Review the Guidelines

In this section you will take a quick look at the simple guidelines:

- **Guideline 1**

  In each Directed Activity, one of the participants on each team may serve as the project manager. The other members may serve as PI Developers. The facilitator will help you to identify your role.

- **Guideline 2**

  To make the most out of your time, please follow the agile principles:

  Simplicity is essential;

  Deliver frequently and continuously;

  Adjust behaviors/communication to become more effective at regular intervals.

- **Guideline 3**

  Adhere to Code of Professional conduct and foster mutual respect

3. Directed Activity – Project Team Assignment

3.1 Objective of this Activity

- Get ready for the subsequent directed activities **as soon as possible**
- Get familiar with the pre-planning phase of the project.
- Get familiar with the resource management concepts.

3.2 Team Assignment

In this activity, the participant who is holding **number 1** will serve as the project manager and the lead developer. The other participants will serve as the PI Developers.

3.3 Agile Decision Making Group Exercises

In this section you will complete the following tasks to fill Table 1 on the next page.

- **Task 1 – Name of Team (duration 1.5 minutes)**

  Each participant will first spend 1 minute to come up with one word (can be any word). And then take turns to speak out that word. The project manager will then combine use these words to establish the team name. Example: "Happy Watermelon Devil House".

- **Task 2 – Company Resource Management (duration 6 minutes)**

  Each participant will first spend 5 minutes on thinking of one key skill and one experience that is relevant to custom PI application development. And then take turns to tell the team about the skills/experiences in a few sentences with in 1 minute. The project manager will summarize each participant's skills & experiences using 5 or less words. Each participant will use these words to fill Table 1.

- **Task 3 – Team Name Announcement (duration less than 5 minutes)**

  Each project manager will tell the entire class about their team name.

Name of Team	Team Member Name	Relevant Skills & Experiences

Table 1: team structure table.

4. Directed Activity – Phase A Project Planning

4.1 Objective of Activity

- Get familiar with some of the project planning concepts and the 3-step model
  - Scope management
  - Resource management
- Use the above concepts to tackle a real-world problem – Let's get it started!

4.2 Team Assignment

In this activity, the participant who is holding **number 2** will serve as the project manager to control the time.

4.3 Understand High Level &Specific Requirements

Task 1 – When should I use AF SDK ? (5 min+5 min)

-
-
-

Task 2 – What does Asset Framework (AF) do for me? (5 min+ 5 min)

The Asset Framework (AF) supplements the architecture by providing a Meta-data structure for all data in the organization ("Data Directory"). Asset Framework (AF) has a rich set of features and functions to organize and enhance the data in the PI Data Archive. Because it offers user-friendly access to the data it is the preferred way for users to interact with their PI system data.

-
-
-

Task 3 – What does (AF) Template do for me? (5 min reading)

At many sites there are no rigorous naming standards for the points. There may be missing descriptions and/or engineering units. The PI System is often used to integrate information from different sources and these sources may not have been configured consistently.

Element templates in AF provide the basis for standardization. When applied for elements that represent the same type equipment, all elements have the same set of attributes with a consistent, user friendly naming. The attributes have same unit of measurement, same data type, the same description, etc. This allows a harmonized, consistent representation of your system.

(Group or Individual Tasks):

Task 1 – From High-Level Requirements to Simple Project Scope

Please read the following user's request:

User's Request:

Today users can selectively and manually reconcile AF template changes between the central corporate office's AF server and remote satellite site's AF servers using PI Builder in Excel (in order to keep track of object changes and/or object deletions). However, this manual method can be very **time-consuming** and prone to **human errors**.

It would be great to have an automatic tool that allow users to maintain AF template changes in multiple remote satellite AF servers.

Start small and don't try to boil the ocean

The first step is to build a prototype that covers the following 3 cases:

- The to-be imported Element Template (ELT) does not exists in the destination => It will be imported as new
- The to-be imported ELT exists with less or differently configured Attribute Templates => The standard Export / Import allows to update the destination
- The ELT exists with additional Attribute Templates (the challenging part!). This "additional" attribute template will be deleted.

Please note analyses and notifications etc are not taken into consideration for the prototype.

Project Scope Management	**Who?**	**What?**
Corporate Strategy	Enterprise	• Operation optimization • Enterprise-wide performance and standardization
Business Need	Management	• Management of change • Replicate solutions across enterprise • Rapidly solve high value problems
Deliverable	Project Manager	• Refer to 4.3 Task 1
Functionalities	PI Developer	• Must-haves? (1) (2) (3) • Nice-to-haves (optional)?

Table 2: scope management.

Task 2 – Review the available resources

Resource Management	**Who?**	**What?**
Company Resource	PI Administrator	Administrators (Admins) – are responsible for maintaining the solution. Depending on the organization, these individuals may configure PI Points, PI Interfaces, install new software, and be the first line of support for users of the solution.
	PI Power users	Use their domain expertise to build the solution. This group builds the Asset Framework layer of the PI System, as well as some high-level visualizations. They may be a team in the corporate headquarters or distributed around several sites
	PI Users	Anyone that needs to access any solution based on the PI System is in this group.
OSIsoft Resource	Techsupport	24×7 Support
	Enterprise Advisors and Experts	Provide strategic advisory services for Enterprise Agreement customers; work with your organization on the rollout of your PI System Data Infrastructure, with a focus on implementation of your most critical initiatives.
Coding Resource	N/A	Live Library

Table 3: resource management.

Coding Resource

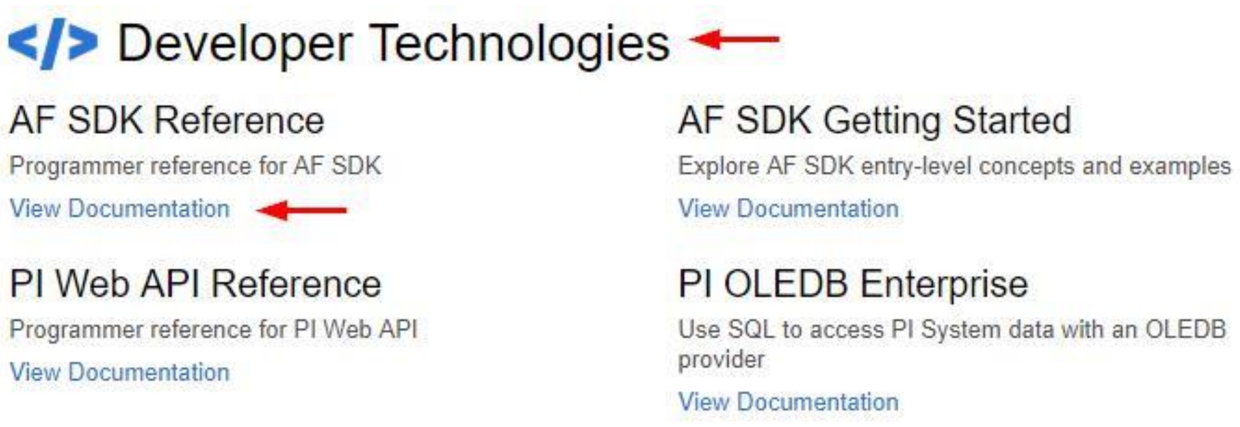
- **Live Library**

Familiarize yourself with the AF SDK Help and Live Library.

For Live Library, open a web browser and enter the URL:

https://livelibrary.osisoft.com/LiveLibrary/

Scroll to the section listed as Developer Technologies. Click on AF SDK Reference.



Developer Technologies

AF SDK Reference
Programmer reference for AF SDK
View Documentation

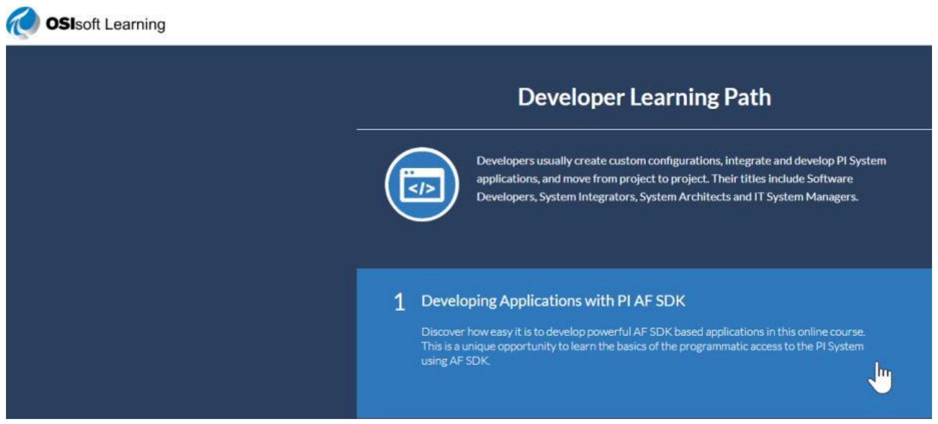
AF SDK Getting Started
Explore AF SDK entry-level concepts and examples
View Documentation

PI Web API Reference
Programmer reference for PI Web API
View Documentation

PI OLEDB Enterprise
Use SQL to access PI System data with an OLEDB provider
View Documentation

- **OSIsoft Learning Resources**

  https://learning.osisoft.com/series/developer#en-language



OSIsoft Learning

Developer Learning Path

Developers usually create custom configurations, integrate and develop PI System applications, and move from project to project. Their titles include Software Developers, System Integrators, System Architects and IT System Managers.

1 Developing Applications with PI AF SDK

Discover how easy it is to develop powerful AF SDK based applications in this online course. This is a unique opportunity to learn the basics of the programmatic access to the PI System using AF SDK.

- **AF SDK Getting Started**

  https://livelibrary.osisoft.com/LiveLibrary/web/ui.xql?action=html&resource=publist_home.html&pub_category=AF-SDK-Getting-Started

5. Directed Activity – Phase B Project Execution

5.1 Objective of Activity

- Design the coding workflow
- Use the coding workflow to implement the project scope

5.2 Get familiar with the Lego pieces (Prework 1 and 2)

The purpose of this exercise is to help you to get familiar with the “Lego” pieces you may use to build the project including: basic AF functionalities and code snippets

5.2.1. Get familiar with general information about the Template:

Before we go the Exercise 1, please feel free to pause here and go through the optional Prework 1 section in your workbook to get familiar with some of the AF SDK Lego pieces such as code syntax you may use to build your solution. If you are an experienced AF SDK programmer, you can skip Prework 1 and jump to Prework 2.

The PISystem represents a single logical data store for the AF SDK. PISystems represents the global collection of PISystem objects. Syntax:

```
public sealed class PISystems : IList<PISystem>,
      ICollection<PISystem>, IEnumerable<PISystem>,
      IEnumerable, IAFList, IList, ICollection, IAFChangedEvent
```

```
public sealed class PISystem : AFObject, IAFSecurable,
IAFTimeSource, IAFChangedEvent, IComparable<PISystem>,
IEquatable<PISystem>, IDisposabl
```

Reference:
https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/T_OSIsoft_AF_PISystems.htm

The AFDatabase represents a single database (data archive) in a PI AF Server. Syntax:

```
public sealed class AFDatabase : AFObject, IAFSecurable,
IAFChangedEvent, IComparable<AFDatabase>
```

Reference:
https://techsupport.osisoft.com/Documentation/PI-AF-SDK/Html/T_OSIsoft_AF_AFDatabase.htm

The AFElementTemplate represents a template of information used to create an AFBaseElement. Several types of objects are elements, including an AFElement, AFEventFrame, or AFModel.

```
public class AFElementTemplate : AFObject, IAFTransactable,
IAFChangedEvent, IAFSecurable, IComparable<AFElementTemplate>
```

Reference:
https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/T_OSIsoft_AF_Asset_AFElementTemplate.htm

AFElementTemplate::Name Property:

```
public string Name { get; set; }
```

Reference:
https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/P_OSIsoft_AF_Asset_AFElementTemplate_Name.htm

AFElementTemplate.CheckIn Method

```
public void CheckIn()
```

Reference:

https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/M_OSIsoft_AF_Asset_AFElementTemplate_CheckIn.htm

Code Snippet:

```
// Create an Element Template AFElementTemplate
myElemTemplate =
myDB.ElementTemplates.Add("MyTemp1");
myElemTemplate.Description = "MyTemp1 Template";
myDB.CheckIn();
```

The AFElementTemplate.FindInstantiatedElements Method method returns a non-paged collection of AFBaseElement objects that were created with this template.

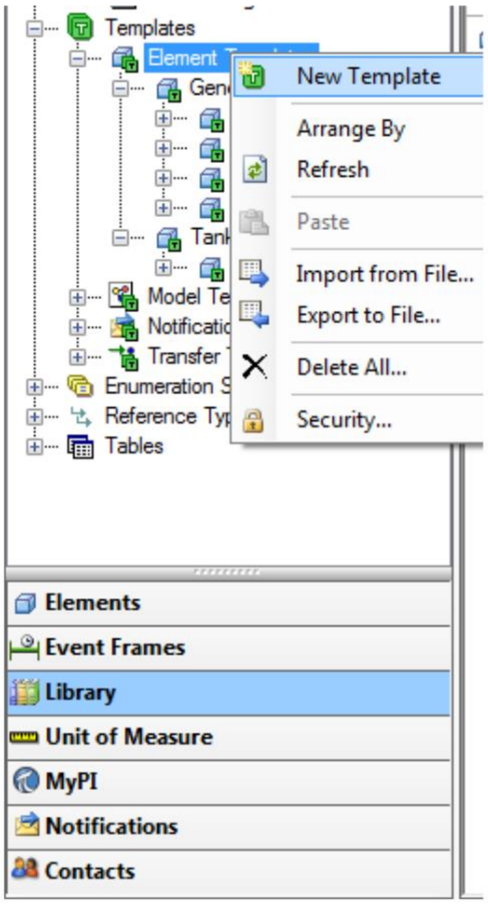
```
public AFNamedCollectionList<AFBaseElement>
FindInstantiatedElements(
       bool includeDerived,
       AFSortField sortField,
       AFSortOrder sortOrder,
       int maxCount
)
```

Reference:
https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/Overload_OSIsoft_AF_Asset_AFElementTemplate_FindInstantiatedElements.htm

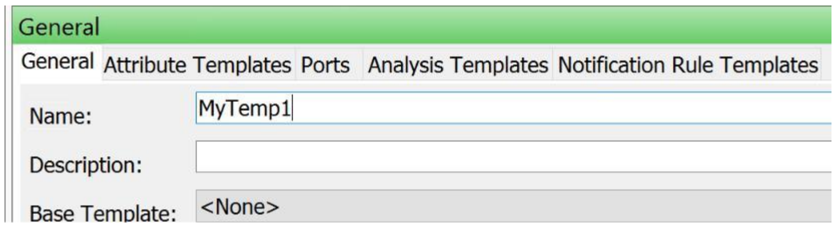
Basic Structure of the AF Element Template:

Element Template Name Combustion Turbine	**Attribute Template Name**	**Naming Pattern:** OOC %Element% %STARTTIME:yyyy-MM-dd HH:mm:ss% (%@Reference Type%-Tire Production)
	CT Gas Flow	Name: **CT Gas Flow** UOM: **pound per second** Data Reference Configuration: **\\%@\PI Data Archive|Name%\%@.|Tagname%;pointtype=Float32; pointsource=OSIDemo-AFAnalysis;span=40;typicalvalue=22.1**
	CT Speed	Name: **CT Speed** UOM: **revolution per minute** Data Reference Configuration: **\\%@\PI Data Archive|Name%\%@.|Tagname%;pointtype=Float32;pointsource=OSIDemo-AFAnalysis**

To create Element Template from within PI System Explorer (PSE), select Library > Templates > Element Templates > New Template.



Then name the template, e.g. MyTemp1.


General

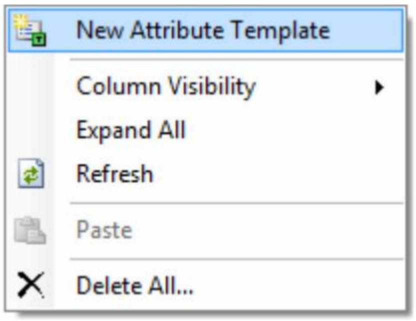
General | Attribute Templates | Ports | Analysis Templates | Notification Rule Templates

Name: MyTemp1

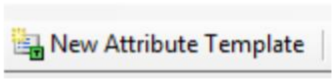
Description:

Base Template: <None>

To create Attribute Template from within the AF Template, select the Attribute Template tab, In the white space, right-click, then select New Attribute Template or select the New Attribute Template button in the menu.



Or



Then name the Attribute, e.g. MyAttr1.

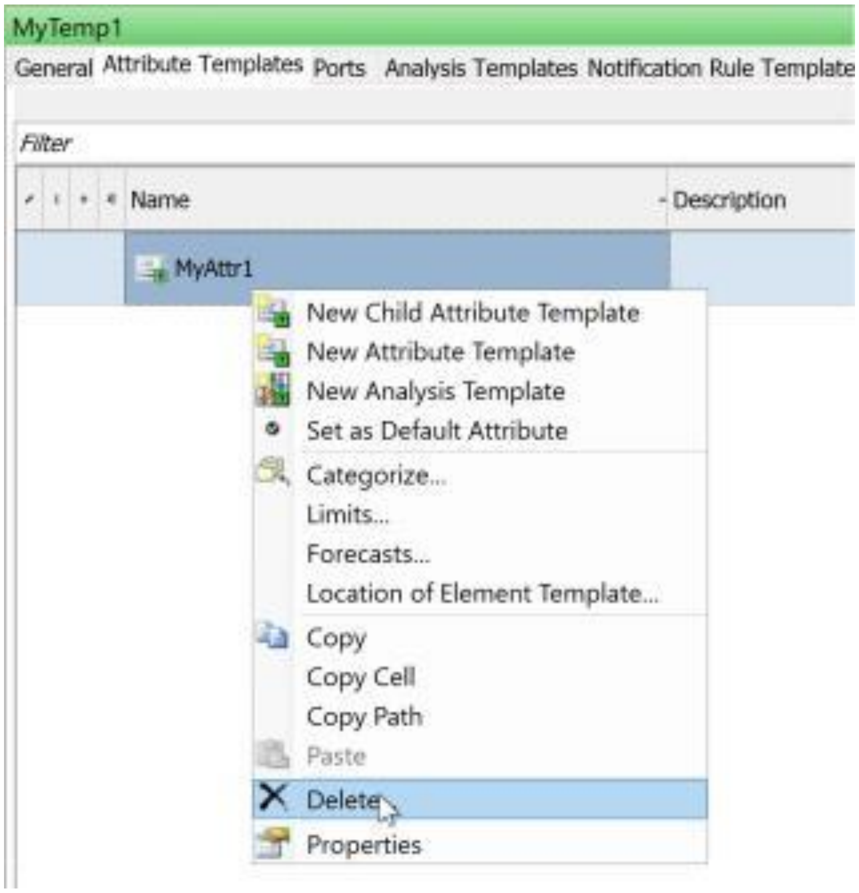
AFElementTemplate.AttributeTemplates code syntax:

```
public sealed class AFAttributeTemplates :
AFNamedCollection<AFAttributeTemplate>,
        IAFNamedCollection<IAFAttribute>, ICollection<IAFAttribute>,
IEnumerable<IAFAttribute>,
        IEnumerable, IList<IAFAttribute>
```

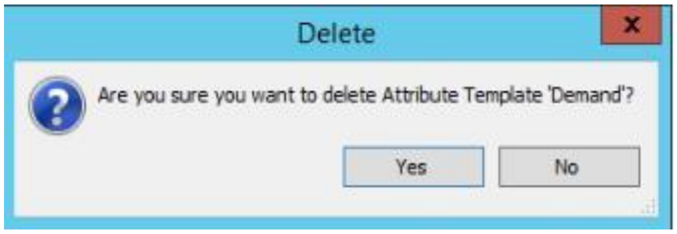
Reference:
https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/T_OSIsoft_AF_Asset_AFAttributeTemplates.htm

To delete AF attributes within the AF templates:

(Step I) From within PSE, select Library > Templates > Element Templates > Attribute Templates, right click and select Delete



(Step II) Confirm the deletion



Are you sure you want to delete Attribute Template 'Demand'?

Exports the XML representation of any object(s) to an XmlWriter.

```
public void ExportXml(
       Object exportObject,
       PIExportMode exportMode,
       XmlWriter writer,
       Object startTime,
       Object endTime,
       EventHandler<AFProgressEventArgs> eventHandler
)
```

Reference:
https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/M_OSIsoft_AF_PISystem_ExportXml_2.htm

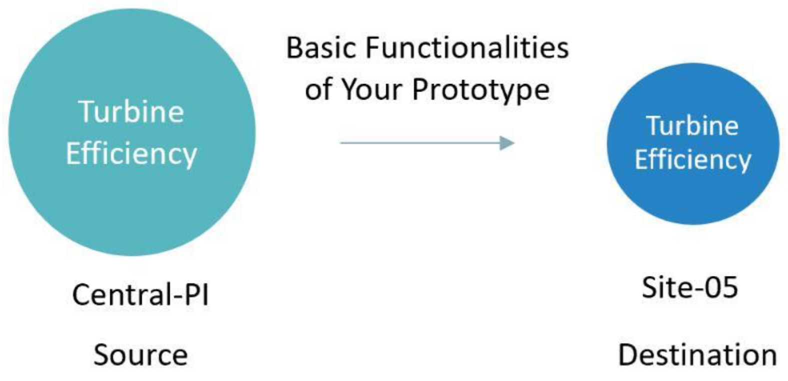
Imports the XML representation of any object from an **XmlTextReader**.

```
public int ImportXml(
       Object importObject,
       PIImportMode importMode,
       XmlReader reader,
       EventHandler<AFProgressEventArgs> eventHandler
)
```

Reference:
https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/M_OSIsoft_AF_PISystem_ImportXml_2.htm

The PI System we have built for you is a simplified version of the PI architecture we presented at the beginning of Lesson 1.

Recall from the scope management section, your prototype should basically do 3 things: browse, export from source and import to the destination. In this simplified architecture you are looking at, your prototype number 1: should allow you to browse the Turbine Efficiency hierarchy in both source Central-PI and destination Site-05; number 2 it should allow you to export AF templates from the source Central-PI AF server, and number 3 import the AF template to the destination Site-05 AF server.



5.3 Prototype Exercises

Exercise 1 - Prompt user to choose…

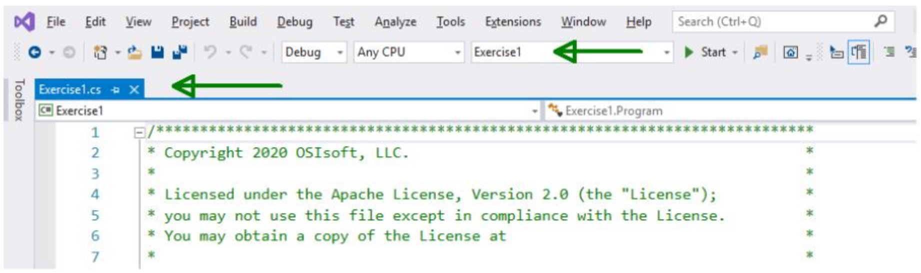
Your action items of this exercise

- Start Visual Studio and open the solution for this lab
- Execute and debug project Exercise1; Use the **AF SDK Programming** reference and get a first idea about Namespaces, Classes, Objects and their Properties and Methods; Understand the code
- Get an idea of the limitations / flaws of the code sample and think about how to make the methods introduced by this exercise more mature

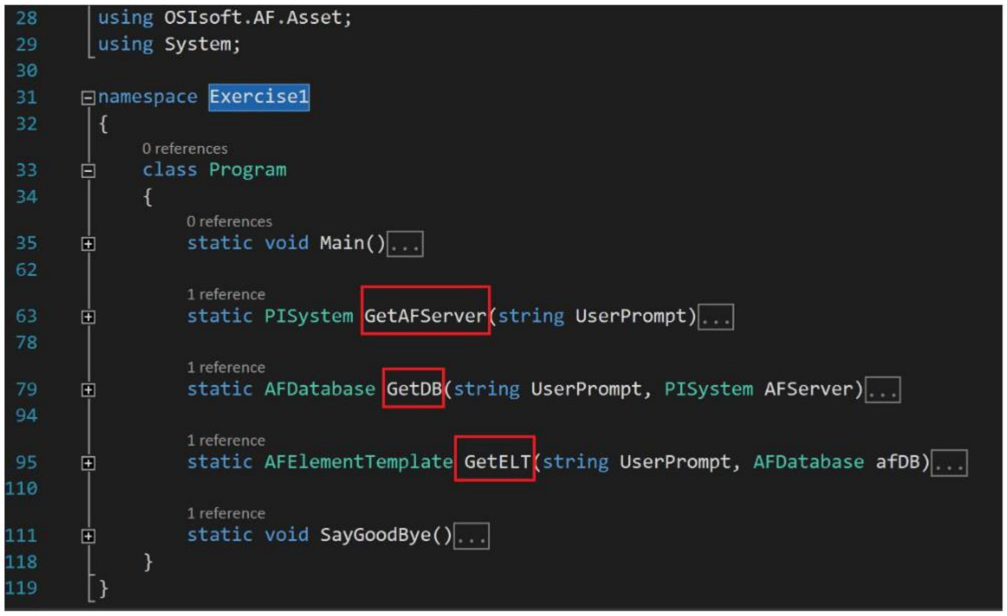
The time planned for this exercise is 30 minutes.

(I) <u>Run the code:</u>

- Start Visual Studio 2015 and open Solution *PI-World-2020-AF-SDK-Lab01*
- Set the Startup Project property of the solution to Exercise1 and open Exercise1.cs



```
/*****************************************************************************
* Copyright 2020 OSIsoft, LLC.                                              *
*                                                                           *
* Licensed under the Apache License, Version 2.0 (the "License");          *
* you may not use this file except in compliance with the License.         *
* You may obtain a copy of the License at                                  *
*                                                                           *
```

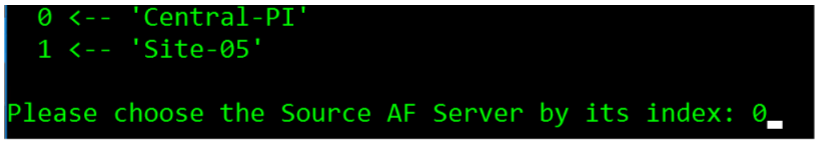


```
using OSIsoft.AF.Asset;
using System;

namespace Exercise1
{
    0 references
    class Program
    {
        0 references
        static void Main()[...]

        1 reference
        static PISystem GetAFServer(string UserPrompt)[...]

        1 reference
        static AFDatabase GetDB(string UserPrompt, PISystem AFServer)[...]

        1 reference
        static AFElementTemplate GetELT(string UserPrompt, AFDatabase afDB)[...]

        1 reference
        static void SayGoodBye()[...]
    }
}
```

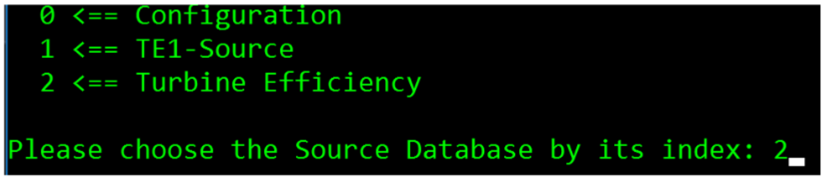
- Start debugging the application and follow the instructions of the Console Application. Make the following choices:

- AF Server:	Central PI
- AF Database:	Turbine Efficiency
- Element Template:	Combustion Turbine

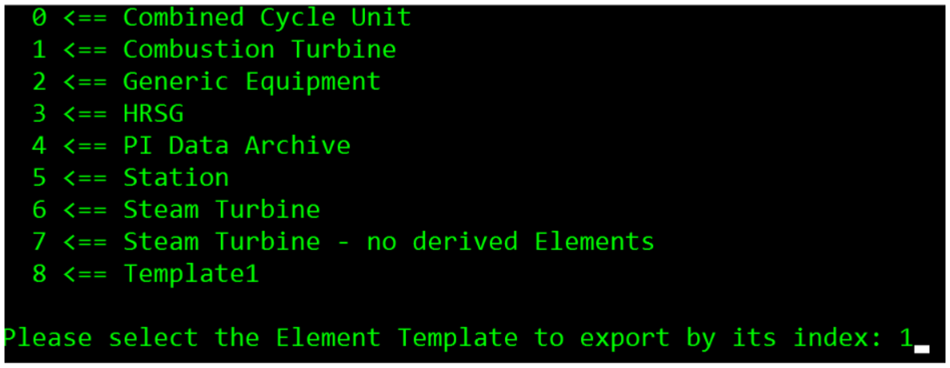
Inspect the console output which should look like the following screenshot



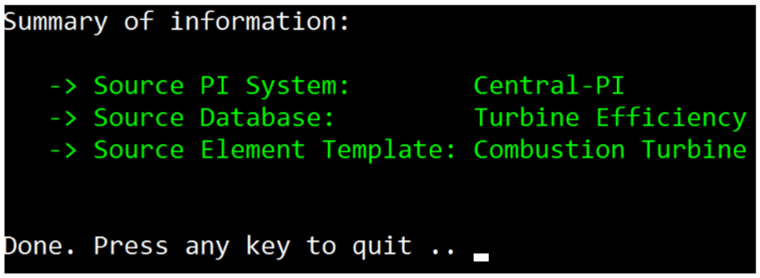
```
  0 <-- 'Central-PI'
  1 <-- 'Site-05'

Please choose the Source AF Server by its index: 0
```



```
  0 <== Configuration
  1 <== TE1-Source
  2 <== Turbine Efficiency

Please choose the Source Database by its index: 2
```



```
  0 <== Combined Cycle Unit
  1 <== Combustion Turbine
  2 <== Generic Equipment
  3 <== HRSG
  4 <== PI Data Archive
  5 <== Station
  6 <== Steam Turbine
  7 <== Steam Turbine - no derived Elements
  8 <== Template1

Please select the Element Template to export by its index: 1
```



```
Summary of information:

   -> Source PI System:        Central-PI
   -> Source Database:         Turbine Efficiency
   -> Source Element Template: Combustion Turbine


Done. Press any key to quit ..
```

(II) Review the code and test your knowledge:

Please answer the following questions which could help to foster understanding the code. We encourage you to use the **PI AF SDK Programming** reference at Live Library

Q1: What does the object **PISystem** represent?
A1: ____
____

Q2: Which are the 2 properties of the **PISystem** object used in this exercise?
A2: ____
____

Q3: Which 3 methods make use of parent -> child relationships?
A3: ____

Bonus: What other options (controls) could be used to allow users to navigate from PISystem over AFDatabase to AFElementTemplate e.g. in an application offering a GUI (Graphical User Interface)?
A: ____

Please see “cheat sheet” on next page for the answers expected. Please raise your hand if something isn’t clear.

(A.2. Answers):

Q1: What does the object **PISystem** represent?
A1: ***PISystem*** *represents a single logical data store (AF Server). Each machine which has AF SDK Client installed, also has a table of known AF Server connections, also referred to as Known Servers Table (KST).*

Q2: Which are the 2 properties of the **PISystem** object used in this exercise?
A2: *The Name property used to identify the PISystem object. The Databases property representing the collection of existing AFDatabase objects.*

Q3: Which 3 methods make use of parent -> child relationships?
A3: *GetAFServer(), GetDB() and GetELT()*

Bonus: What other options (controls) could be used to allow users to navigate from PISystem over AFDatabase to AFElementTemplate e.g. in an application offering a GUI (Graphical User Interface)?
A: *ListView, ComboBox, TreeView*

A.3. Break your code :

Run Exercise1.cs again in Debug mode. When being prompted for a selection, enter something outside the offered choice e.g. a number, higher than the offered indexes, a character or a string. Observe what happens. Why is this?

Note: All methods have less than 20 simple code lines and are relatively easy to understand but this way we won't pass any user acceptance test.

Exercise 1 Code Appendix:

Main code:

```
// Prompt for source information
Console.ForegroundColor = ConsoleColor.Green;

string userPrompt = "Please choose the Source AF Server by its index:
"; PISystem sourcePISystem = GetAFServer(userPrompt);

userPrompt = "Please choose the Source Database by its index:
"; AFDatabase sourceDB = GetDB(userPrompt, sourcePISystem);

userPrompt = "Please select the Element Template to export by its index: ";
AFElementTemplate sourceELT = GetELT(userPrompt, sourceDB);

// Inform user about the choices made
Console.Clear(); Console.ForegroundColor =
ConsoleColor.White; Console.WriteLine("Summary of
information: \r\n"); Console.ForegroundColor =
ConsoleColor.Green;
Console.WriteLine("   -> Source PI System:        {0}", sourcePISystem);
Console.WriteLine("   -> Source Database:         {0}", sourceDB);
Console.WriteLine("   -> Source Element Template: {0}\r\n", sourceELT);

// We need to prevent our app from closing before the user was able to digest
information SayGoodBye();
```

1.1. Implement GetAFServer

```
        static PISystem GetAFServer(string UserPrompt)
        {
            PISystems knownServers = new PISystems();
            int index = -1;
            Console.Clear();
            foreach (PISystem knownServer in knownServers)
            {
                index++;
                Console.WriteLine("{0,3} <-- '{1}' ", index, knownServer);
            }
            Console.Write("\r\n{0}", UserPrompt);
            string userInput = Console.ReadLine();
            int chosenIndex = int.Parse(userInput);
            return knownServers[chosenIndex];
        }
```

1.2. Implement GetDB

```
static AFDatabase GetDB(string UserPrompt, PISystem AFServer)
{
    AFDatabases afDBs = AFServer.Databases;
    int index = -1;
    Console.Clear();
    foreach (AFDatabase afDB in afDBs)
    {
        index++;
        Console.WriteLine("{0,3} <== {1}", index, afDB.Name);
    }
    Console.Write("\r\n{0}", UserPrompt);
    string userInput = Console.ReadLine();
    int chosenIndex = int.Parse(userInput);
    return afDBs[chosenIndex];
}
```

1.3. Implement GetELT

```
static AFElementTemplate GetELT(string UserPrompt, AFDatabase afDB)
{
    AFElementTemplates ELTs = afDB.ElementTemplates;
    int index = -1;
    Console.Clear();
    foreach (AFElementTemplate ELT in ELTs)
    {
        index++;
        Console.WriteLine("{0,3} <== {1}", index, ELT.Name);
    }
    Console.Write("\r\n{0}", UserPrompt);
    string userInput = Console.ReadLine();
    int chosenIndex = int.Parse(userInput);
    return ELTs[chosenIndex];
}
```

Exercise 2 – Improve the code

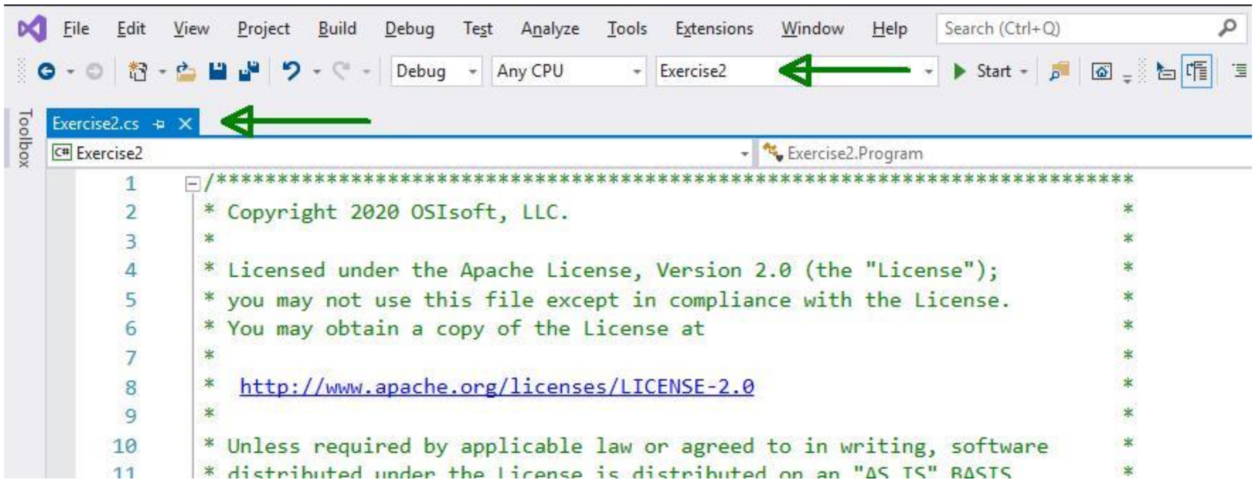
For this exercise we have enhanced the code from the previous exercise as follows:

- Add a method which Exports the chosen Element Template to an Xml file
- Make GetAFServer(), GetDB() and GetELT() more robust. Consider the following situations with potential to raise an exception:
  - User enters something that cannot be parsed as Integer
  - User enters a number outside the expected range
  - For GetAFServer make sure the chosen Server can be

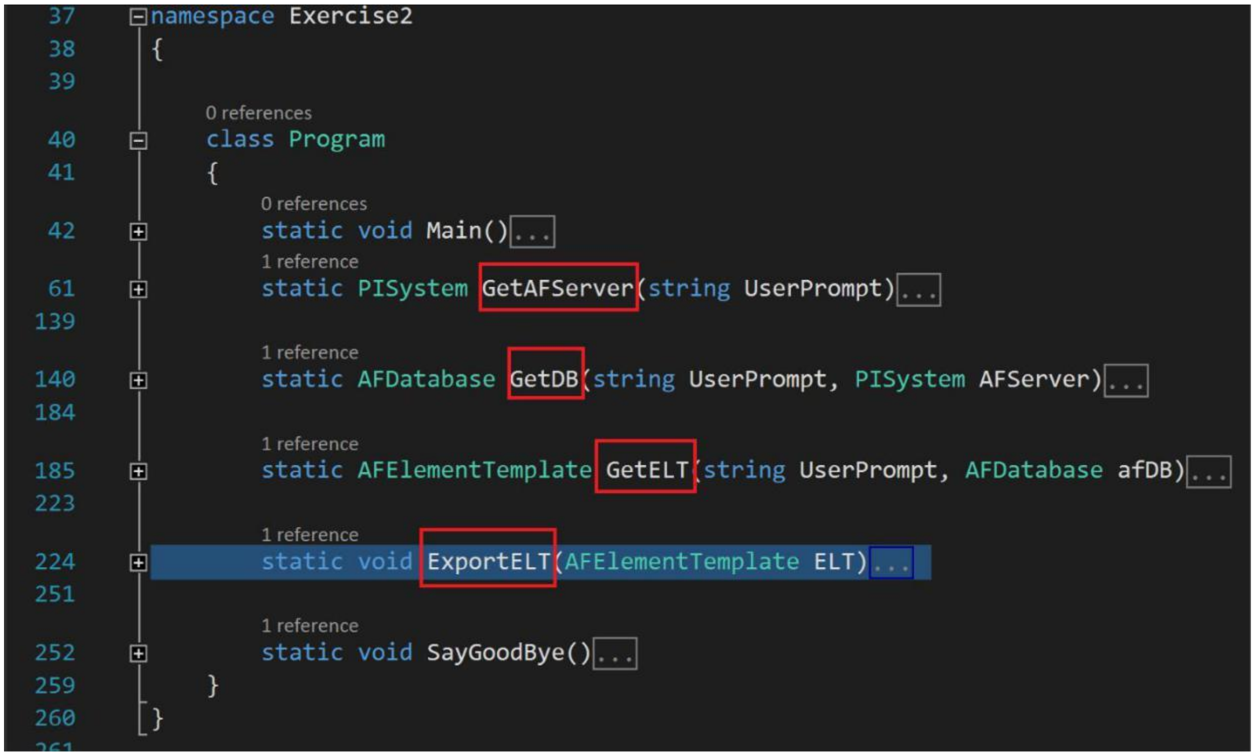
connected 20 minutes are planned for this exercise.

(I) <u>Run the code:</u>

- Change to Project Exercise2 and open Exercise2.cs



```
/*****************************************************************************
* Copyright 2020 OSIsoft, LLC.                                               *
*                                                                            *
* Licensed under the Apache License, Version 2.0 (the "License");            *
* you may not use this file except in compliance with the License.          *
* You may obtain a copy of the License at                                    *
*                                                                            *
*   http://www.apache.org/licenses/LICENSE-2.0                               *
*                                                                            *
* Unless required by applicable law or agreed to in writing, software        *
* distributed under the License is distributed on an "AS IS" BASIS           *
```

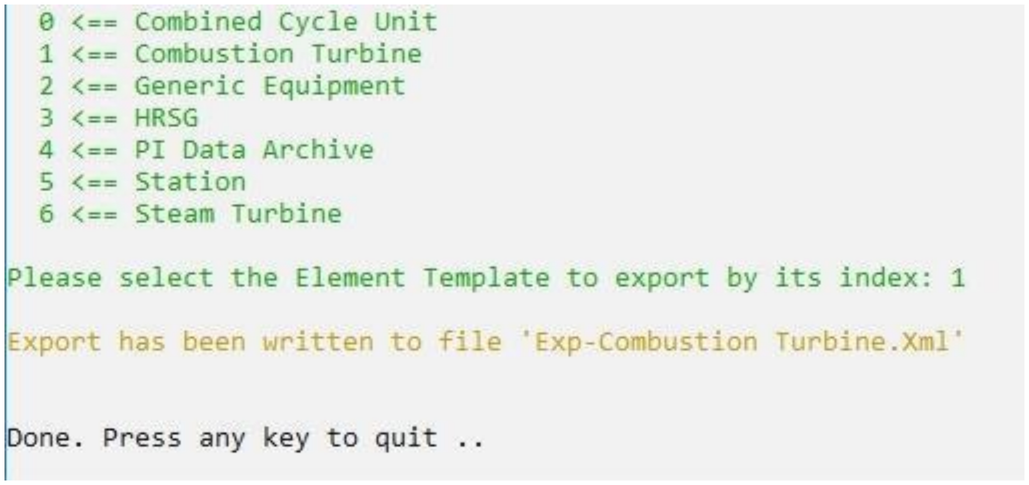


```
namespace Exercise2
{

    0 references
    class Program
    {
        0 references
        static void Main()...
        1 reference
        static PISystem GetAFServer(string UserPrompt)...

        1 reference
        static AFDatabase GetDB(string UserPrompt, PISystem AFServer)...

        1 reference
        static AFElementTemplate GetELT(string UserPrompt, AFDatabase afDB)...

        1 reference
        static void ExportELT(AFElementTemplate ELT)...

        1 reference
        static void SayGoodBye()...
    }
}
```

- Start debugging the application and Follow the instructions of the Console Application. Make the following choices:

- AF Server:	Central PI
- AF Database:	Turbine Efficiency
- Element Template:	Combustion Turbine

- Observe the console output and solve the following tasks in your team



```
  0 <== Combined Cycle Unit
  1 <== Combustion Turbine
  2 <== Generic Equipment
  3 <== HRSG
  4 <== PI Data Archive
  5 <== Station
  6 <== Steam Turbine

Please select the Element Template to export by its index: 1

Export has been written to file 'Exp-Combustion Turbine.Xml'


Done. Press any key to quit ..
```

(II) Test yourself:

- What’s the name of the export file? Where was it stored? Which Application can you use to inspect the file for its content?
- What does “Updating Config for…” mean? Which line of the code?

Note: We are not looking at the updated GetAFServer(), GetDB and GetELT methods yet but we will keep this as an optional exercise to be revisited later, if time permits.

Exercise 2 Code Appendix:

2.1. A more robust GetAFServer, GetDB and GetELT

```
        static PISystem GetAFServer(string UserPrompt)
         {
            PISystems piSystems = new PISystems();
            List<string> knownServers = new List<string>();
            PISystem piSystemChosen = null;
            bool IsValidUserChoice = false;
            if (knownServers.Count == 0)
            {
                foreach (PISystem piSystem in piSystems)
                {
                    knownServers.Add(piSystem.ConnectionInfo.Host);
                }
            }

            // Repeat this loop until the user entered a valid choice and the Server can
be connected
            while (!IsValidUserChoice)
            {
                int index = -1;
                Console.Clear();
                foreach (string knownServer in knownServers)
                {
                    index++;
                    Console.WriteLine("{0,3} <-- '{1}' ", index, knownServer);
                }
                Console.Write("\r\n{0}", UserPrompt);
                string userInput = Console.ReadLine();

                // Check if whatever was entered can be parsed into an
                integer if (int.TryParse(userInput, out int chosenIndex))
                {
                    // The parsed integer is outside the expected range
                    if ((chosenIndex < 0) || chosenIndex > index)
                    {
                        ConsoleColor colorBuffer = Console.ForegroundColor;
                        Console.ForegroundColor = ConsoleColor.Yellow;
                        Console.Write("\r\n{0} is not between 0 and {1}. Please press
[Enter] to try again.", chosenIndex, index);
                        Console.ReadLine();
                        Console.ForegroundColor = colorBuffer;
                    }
                    else
                    {
                        // Check if the PISystem can be connected
                        try
                        {
                            piSystemChosen = piSystems[chosenIndex];
                            piSystemChosen.Connect();
                            if (piSystemChosen.ConnectionInfo.IsConnected)
                            {
                                IsValidUserChoice = true;
                            }
                        }
                        catch (Exception ex)
                        {
                            ConsoleColor colorBuffer = Console.ForegroundColor;
                            Console.ForegroundColor = ConsoleColor.Yellow;
                            Console.WriteLine("\r\nUnable to connect '{0}'",
piSystemChosen.ConnectionInfo.Host);
                            Console.ForegroundColor = ConsoleColor.Red;
                            Console.WriteLine("An exception was raised: {0} - {1}",
ex.HResult, ex.Message);
                            Console.Write("Please press [Enter] to try again.");
                            Console.ReadKey();
                            Console.ForegroundColor = colorBuffer;
                        }
                    }
                }
                // The string eneterd by the user cannot be parsed to
                int else
                {
                    ConsoleColor colorBuffer = Console.ForegroundColor;
                    Console.ForegroundColor = ConsoleColor.Yellow;
                    Console.Write("\r\nFailed to convert '{0}' to a number.
Please [Enter] enter to try again.", userInput);
                    Console.ReadLine();
                    Console.ForegroundColor = colorBuffer;
                }
            }
            // If we get here, the user has made a valid choice and the server could be
connected
            return piSystemChosen;
        }
        static AFDatabase GetDB(string UserPrompt, PISystem AFServer)
        {
            AFDatabases afDBs = AFServer.Databases;
            AFDatabase afDBChosen = null;
            bool IsValidUserChoice = false;
            while (!IsValidUserChoice)
            {
                int index = -1;
                Console.Clear();
                foreach (AFDatabase afDB in afDBs)
                {
                    index++;
                    Console.WriteLine("{0,3} <== {1}", index, afDB.Name);
                }
                Console.Write("\r\n{0}", UserPrompt);
                string userInput = Console.ReadLine();
                if (int.TryParse(userInput, out int chosenIndex))
                {
                    if ((chosenIndex < 0) || chosenIndex > index)
                    {
                        ConsoleColor colorBuffer = Console.ForegroundColor;
                        Console.ForegroundColor = ConsoleColor.Yellow;
                        Console.Write("{0} is not between 0 and {1}. Please press [Enter]
to try again.", chosenIndex, index);
                        Console.ReadLine();
                        Console.ForegroundColor = colorBuffer;
                    }
                    else
                    {
                        afDBChosen = afDBs[chosenIndex];
                        IsValidUserChoice = true;
                    }
                }
                else
                {
                    ConsoleColor colorBuffer = Console.ForegroundColor;
                    Console.ForegroundColor = ConsoleColor.Yellow;
                    Console.Write("Failed to convert '{0}' to a number. Please [Enter]
enter to try again.", userInput);
                    Console.ReadLine();
                    Console.ForegroundColor = colorBuffer;
                }
            }
            return afDBChosen;
        }
        static AFElementTemplate GetELT(string UserPrompt, AFDatabase afDB)
        {
            AFElementTemplates ELTs = afDB.ElementTemplates;
            AFElementTemplate ELTChoosen = null;
            bool IsValidUserChoice = false;
            while (!IsValidUserChoice)
            {
                int index = -1;
                Console.Clear();
                foreach (AFElementTemplate ELT in ELTs)
                {
                    index++;
                    Console.WriteLine("{0,3} <== {1}", index, ELT.Name);
                }
                Console.Write("\r\n{0}", UserPrompt);
                string userInput = Console.ReadLine();
                if (int.TryParse(userInput, out int chosenIndex))
                {
                    if ((chosenIndex < 0) || chosenIndex > index)
                    {
                        Console.Write("{0} is not between 0 and {1}. Please press [Enter]
to try again.\r\n", chosenIndex, index);
                        Console.ReadLine();
                    }
                    else
                    {
                        ELTChoosen = ELTs[chosenIndex];
                        IsValidUserChoice = true;
                    }
                }
                else
                {
                    Console.Write("Failed to convert '{0}' to a number. Please
[Enter] enter to try again.\r\n", userInput);
                    Console.ReadLine();
                }
            }
            return ELTChoosen;
        }
```

2.2. Implement ExportELT

```
        static void ExportELT(AFElementTemplate ELT)
        {
            // Create the export file name
            string expFileName = string.Format("Exp-{0}.Xml", ELT.Name);

            // Define the settings for the XMLWriter
            XmlWriterSettings wSettings = new XmlWriterSettings
            {
                Encoding = Encoding.UTF8
            };

            // Create the XMLWriter instance using file name and settings
            XmlWriter xWriter = XmlWriter.Create(expFileName, wSettings);

            // Get the PISystem from the Element Template
            PISystem sourcePI = ELT.PISystem;

            // Write the Xml export
            sourcePI.ExportXml(ELT, PIExportMode.NoUniqueID | PIExportMode.Flat,
xWriter, null, null, null);

            // Close the XMLWriter
            object xWriter.Close();

            // Inform the user about the Export file name Console.ForegroundColor
            = ConsoleColor.Yellow; Console.WriteLine("\r\nExport has been written
            to file '{0}'\r\n",
expFileName);
        }
```

Exercise 3 – Final Project

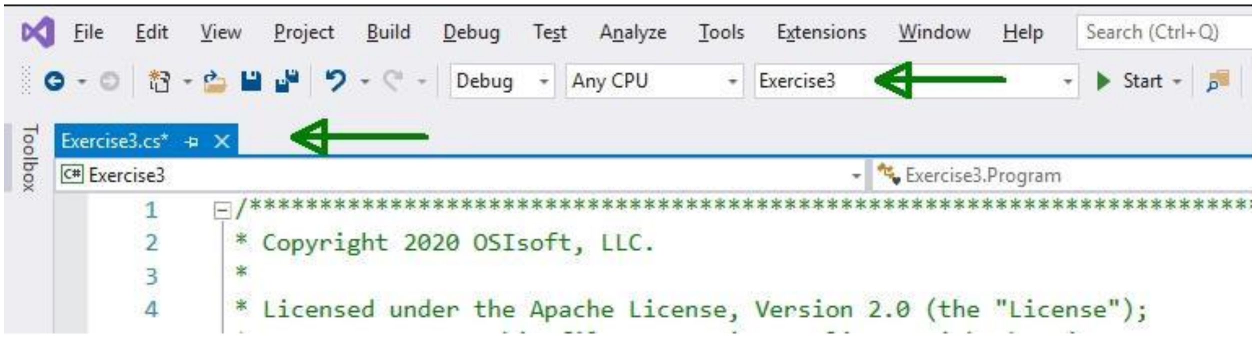
This Exercise is supposed to represent the final prototype of our feasibility test project. The complete scope is as follows:

- Prompt user to choose the source PI System
- Prompt user to choose the source Database
- Prompt user to choose the Element Template to export
- Prompt the user to choose the destination PI System
- Prompt the user to choose the destination Database
- Summarize choices at the prompt and ask confirmation
- Export from source and import to destination
    - If the Element Template does not exist, import it as new
    - If the Element Template exists, import the source as update.
    - Add Attribute Templates not existent in the destination
    - Remove Attribute Templates not existent in the source
    - For update operations, re-create the configuration of derived Elements

The time planned for this exercise is approximately 40 minutes.

(I) <u>Get ready:</u>

- Set the compiler to Exercise3 and open Exercise3.cs



```
/*****************************************************************************
* Copyright 2020 OSIsoft, LLC.
*
* Licensed under the Apache License, Version 2.0 (the "License");
```

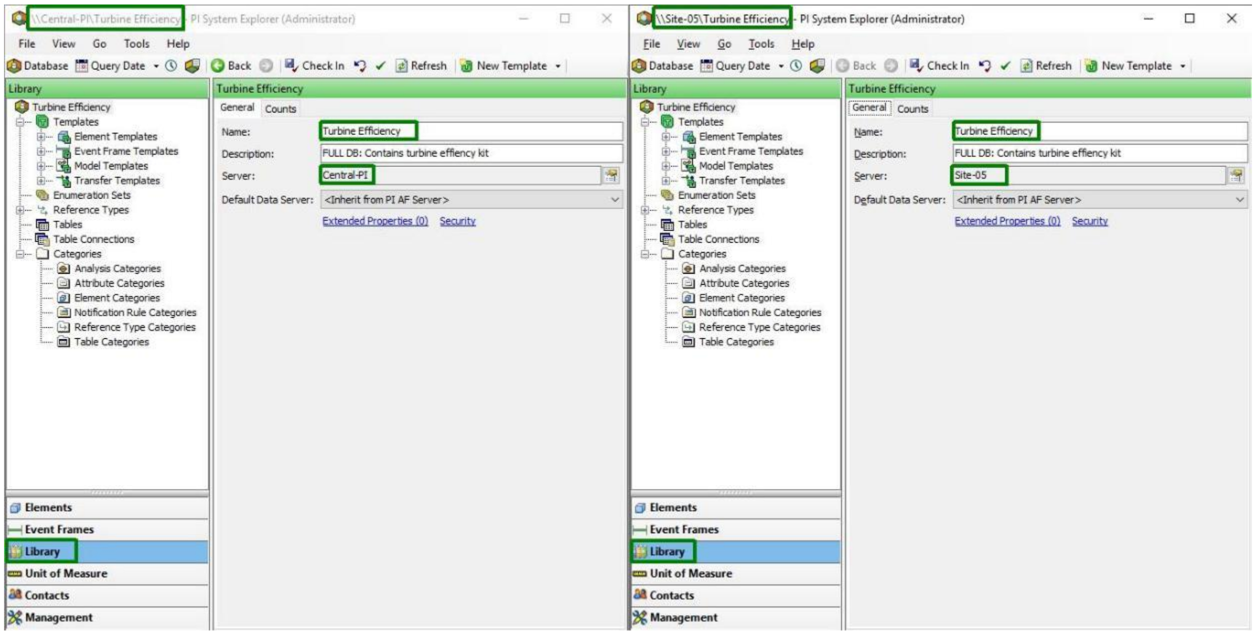
Please pay attention to your course instructor who will explain the new methods GetELTExport(), ImportELTAsNew() and ImportELTAsUpdate().

- zq

  To be able to inspect the data before and after each execution, we recommend running 2 PI System Explorer (PSE) sessions side-by-side, the one on the left-hand-side connected to the **source** and the one on the right-hand-side connected to the **destination**.

Because we are dealing with Element Templates, we further suggest focusing at the Library. When testing Element Template updates, we will also inspect Elements derived from a specific Template.

The database we've chosen for exercises is the on which is available as **Asset Based PI Example Kit Turbine Efficiency**.



\\Central-PI\Turbine Efficiency - PI System Explorer (Administrator)

Name: Turbine Efficiency
Description: FULL DB: Contains turbine effiency kit
Server: Central-PI
Default Data Server: <Inherit from PI AF Server>

\\Site-05\Turbine Efficiency - PI System Explorer (Administrator)

Name: Turbine Efficiency
Description: FULL DB: Contains turbine effiency kit
Server: Site-05
Default Data Server: <Inherit from PI AF Server>

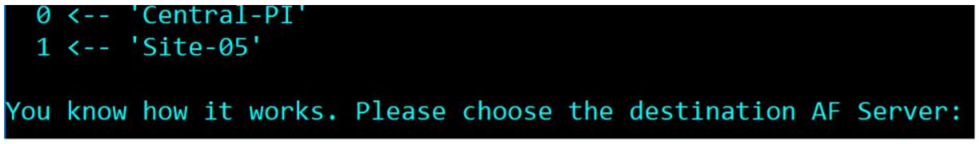
<u>(II) Run the code:</u>

Test 1 – Create new Template

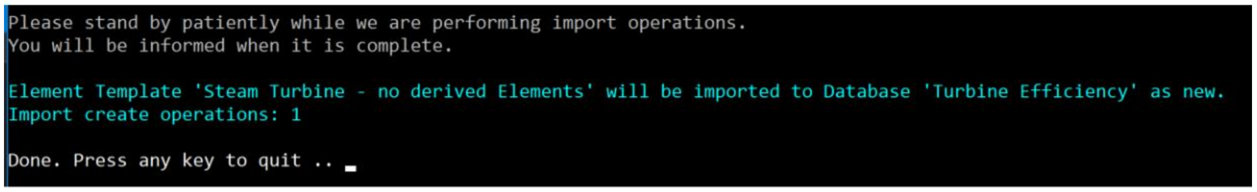
Start Exercise3 in Debug mode, follow the instructions at the prompt and make the following choices:

- Source AF Server :	Central-PI
- Source Database :	Turbine Efficiency
- Source Element Template :	**Steam Turbine - no derived Elements**
- Destination AF Server:	Site-XX (your local server)
- Destination Database:	Turbine Efficiency

Verify with the right-hand-side PSE that the Element Template has been created.

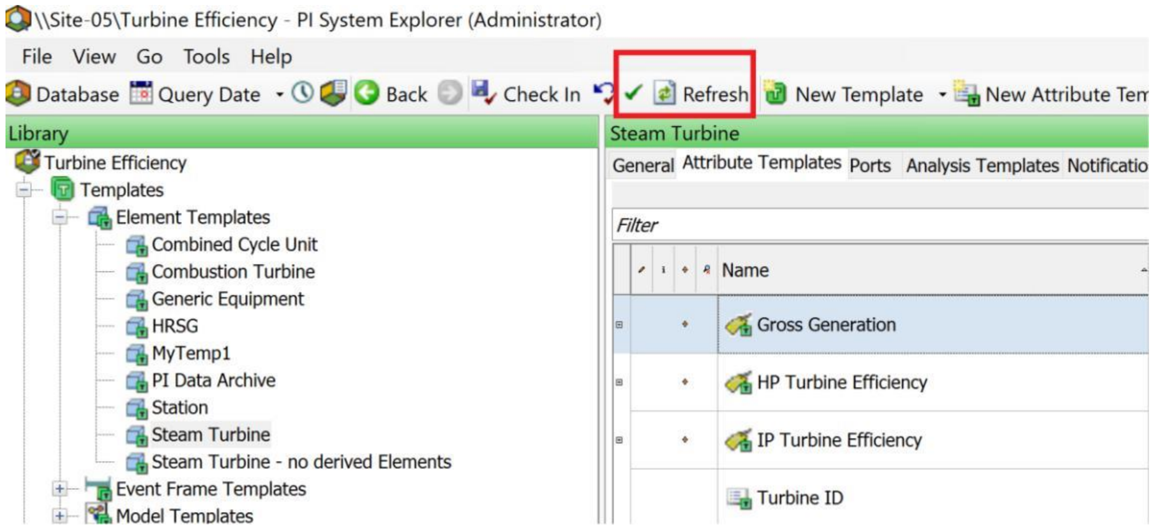


```
  0 <-- 'Central-PI'
  1 <-- 'Site-05'

You know how it works. Please choose the destination AF Server:
```



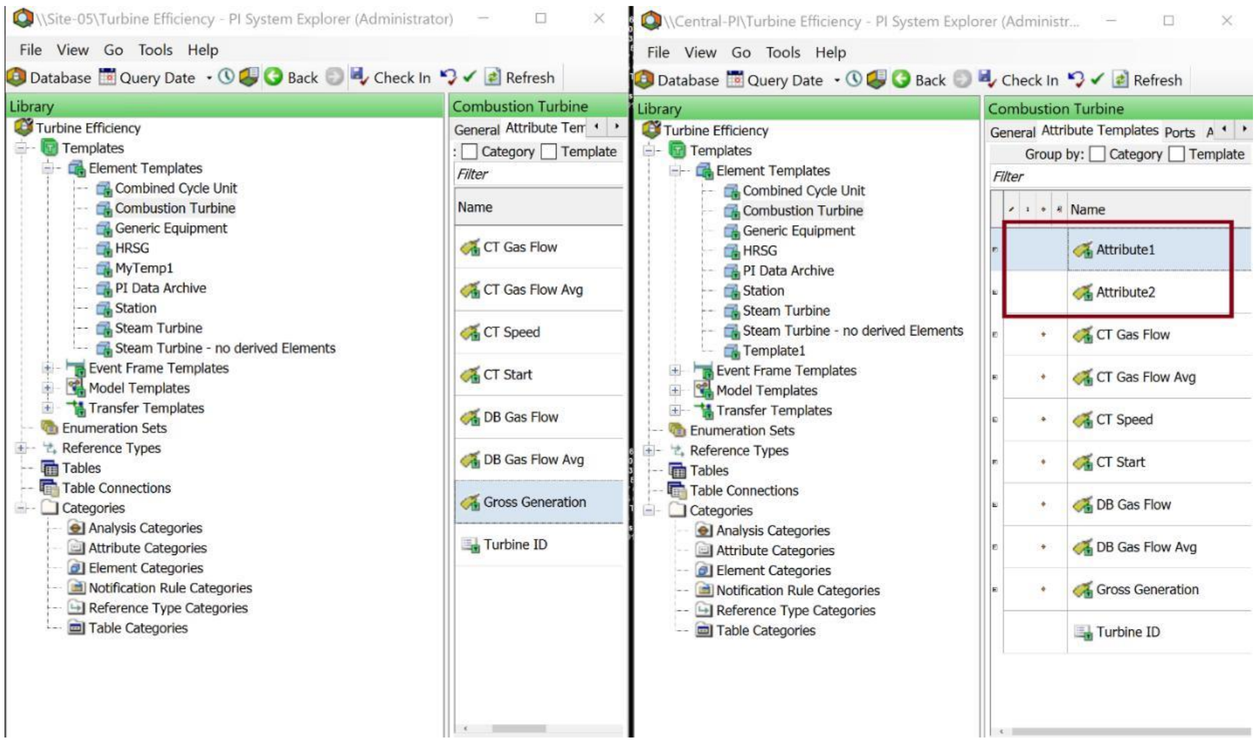
```
Please stand by patiently while we are performing import operations.
You will be informed when it is complete.

Element Template 'Steam Turbine - no derived Elements' will be imported to Database 'Turbine Efficiency' as new.
Import create operations: 1

Done. Press any key to quit ..
```

Note: You need to [Refresh] in order to update the AF SDK client cache and to make changes visible.



Test 2 – Update an existing Template with additional Attributes

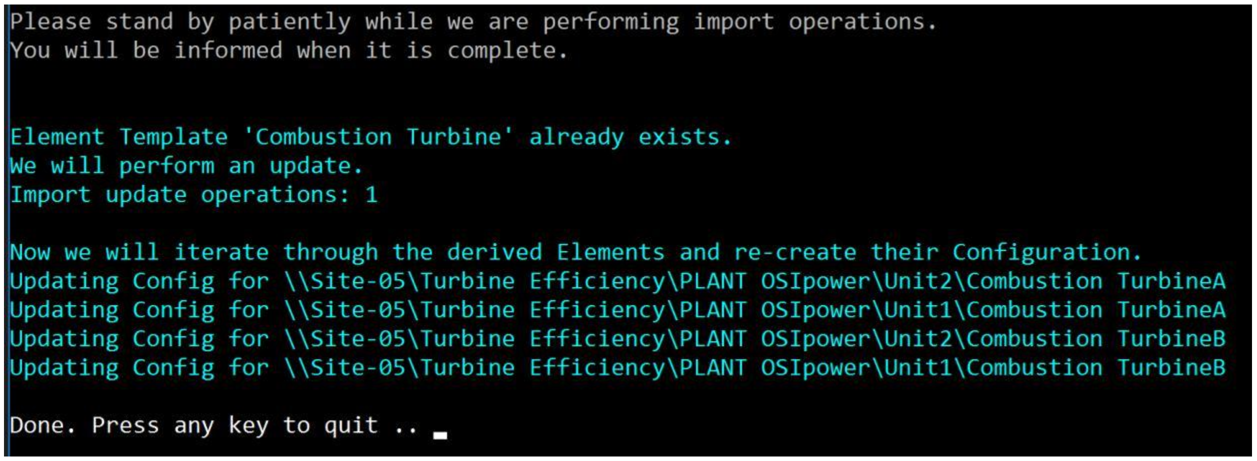
For this test, we have added 2 additional Attribute Templates (Attribute1 and Attribute2) to Element Template “Combustion Turbine” in the source (Central-PI). Before executing the test, please take some time to inspect “Combustion Turbine” template in source and destination. Please also inspect derived Elements. Witness the additional attributes:



Start Exercise3 in Debug mode and make the following choices:

- Source AF Server : Central-PI
- Source Database : Turbine Efficiency
- Source Element Template : **Combustion Turbine**
- Destination AF Server: Site-XX (your local server)
- Destination Database: Turbine Efficiency

Our application informs about the progress at the prompt. Please stand-by patiently until it reports completion. Please note the information about re-creating Elements derived from the Template.



```
Please stand by patiently while we are performing import operations.
You will be informed when it is complete.


Element Template 'Combustion Turbine' already exists.
We will perform an update.
Import update operations: 1

Now we will iterate through the derived Elements and re-create their Configuration.
Updating Config for \\Site-05\Turbine Efficiency\PLANT OSIpower\Unit2\Combustion TurbineA
Updating Config for \\Site-05\Turbine Efficiency\PLANT OSIpower\Unit1\Combustion TurbineA
Updating Config for \\Site-05\Turbine Efficiency\PLANT OSIpower\Unit2\Combustion TurbineB
Updating Config for \\Site-05\Turbine Efficiency\PLANT OSIpower\Unit1\Combustion TurbineB

Done. Press any key to quit ..
```

[Refresh] the right-hand-side PSE session and observe the changes for the Element Template “Combustion Turbine”. Please also inspect the Attributes again showing for derived Elements.

Test 3 – Update an existing Template with additional Attributes

Our final test is covering a scenario where Attribute Templates have been removed in the source and others were added.

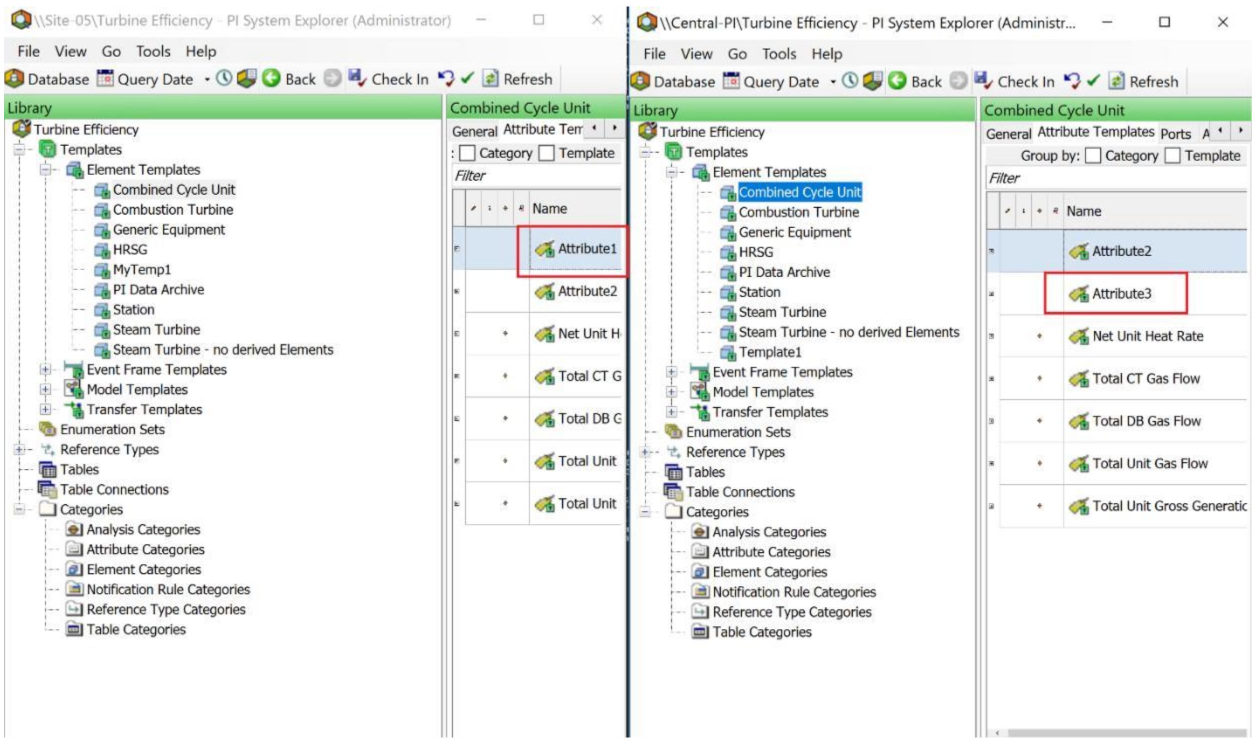
Please verify “Combined Cycle Unit” at the destination and verify the following Attribute Templates were added:

- Attribute1 Template pointing to PI Point SINUSOID
- Attribute2 Template pointing to PI Point SINUSOIDU

Before moving over to the source, check the Attribute Categories existing in the destination.

Now inspect “Combined Cycle Unit” at the source. The additions are as follows:

- Attribute2 Template pointing to PI Point CDT158
- Attribute3 Template pointing to PI Point CDM158
- Attribute3 Template showing with Category “New”

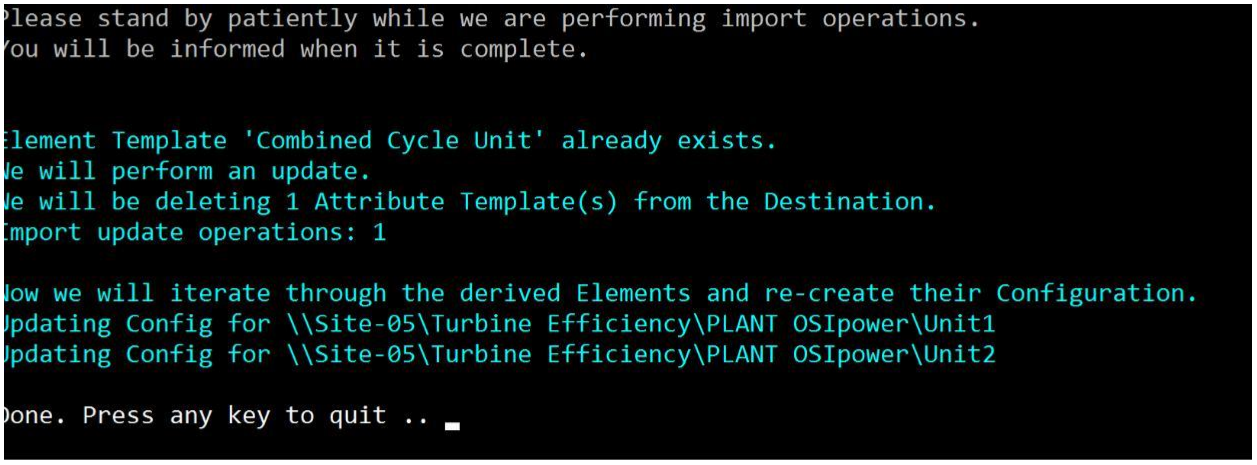


Please mark those of the 5 items above you expect to see after importing from source to destination. Please be reminded that our export / import operation is designed to

- Update existing Attribute Templates with the definitions from source
- Create Attribute Templates missing in the destination
- Delete Attribute Templates missing in the source

Start Exercise3 in Debug mode and make the following choices:

- Source AF Server :	Central-PI
- Source Database :	Turbine Efficiency
- Source Element Template :	**Combined Cycle Unit**
- Destination AF Server:	Site-XX (your local server)
- Destination Database:	Turbine Efficiency



```
lease stand by patiently while we are performing import operations.
ou will be informed when it is complete.


lement Template 'Combined Cycle Unit' already exists.
e will perform an update.
e will be deleting 1 Attribute Template(s) from the Destination.
mport update operations: 1

ow we will iterate through the derived Elements and re-create their Configuration.
pdating Config for \\Site-05\Turbine Efficiency\PLANT OSIpower\Unit1
pdating Config for \\Site-05\Turbine Efficiency\PLANT OSIpower\Unit2

one. Press any key to quit .. _
```

[Refresh] the PSE session looking at the destination (right-hand-side) and compare the results with your expectations.

Please compare if you see the following results. Please also verify those against your expectations.

Attribute1 Template pointing to PI Point SINUSOID	Deleted
Attribute2 Template pointing to PI Point SINUSOIDU	Updated
Attribute3 Template pointing to PI Point CDM158	Added
Attribute3 using Category “New”	No

Exercise 3 Code Appendix:

Implement ImportELTAsNew, ImportELTAsUpdate (and PeformImport)

```
       static void ImportELTAsNew(AFElementTemplate SourceELT, AFDatabase DestDB)
        {
            string expString = GetELTExport(SourceELT);
            PISystem destPISystem = DestDB.PISystem;
            int impCreateOps = destPISystem.ImportXml(DestDB, PIImportMode.AutoCheckIn
| PIImportMode.AllowCreate, expString);
            Console.WriteLine("Import create operations: {0}", impCreateOps);
        }


        static void ImportELTAsUpdate(AFElementTemplate SourceELT, AFElementTemplate
DestELT)
        {
            string expContent = GetELTExport(SourceELT);
            string deletesToInsert = string.Empty;
            string impContent = expContent;
            int deleteCount = 0;
            foreach (AFAttributeTemplate destAFAT in DestELT.AttributeTemplates)
            {
                bool exists = false;
                foreach (AFAttributeTemplate sourceAFAT in SourceELT.AttributeTemplates)
                {
                    if (destAFAT.Name == sourceAFAT.Name)
                    {
                        exists = true;
                    }
                }
                if (!exists)
                {
                    deleteCount++;
                    deletesToInsert += "    <AFAttributeTemplate
operation=\"delete\">\r";
                    deletesToInsert += string.Format("      <Name>{0}</Name>\r",
destAFAT.Name);
                    deletesToInsert += "    </AFAttributeTemplate>\r";
                }
            }
            if (deleteCount > 0)
            {
                int firstAFATposition = expContent.IndexOf("    <AFAttributeTemplate");
                string expContWithDeletes = expContent.Insert(firstAFATposition,
deletesToInsert);
                Console.WriteLine("We will be deleting {0} Attribute Templates from
the Destination.", deleteCount);
                impContent = expContWithDeletes;
            }
            // The next 4 Lines are for the actual import
            PISystem destPISystem = DestELT.PISystem;
            AFDatabase destDB = DestELT.Database;
            int impUpdates = destPISystem.ImportXml(destDB, PIImportMode.AutoCheckIn
| PIImportMode.AllowCreate | PIImportMode.AllowUpdate, impContent);
            Console.WriteLine("Import update operations: {0}\r\n", impUpdates);

            // The configuration of Elements derived from the updated Template need to be
re-created
            Console.WriteLine("Now we will iterate through the derived Elements and
re-create their Configuration.");
            foreach (AFBaseElement instantiatedElements in
DestELT.FindInstantiatedElements(false, AFSortField.Name, AFSortOrder.Ascending, 1000))
            {
                Console.WriteLine("Updating Config for
{0}", instantiatedElements.GetPath());
                AFDataReference.CreateConfig(instantiatedElements, false, null);
            }
            destDB.CheckIn();
        }



        static void PerformImport(AFElementTemplate Source, AFDatabase Destination)
        {
            AFElementTemplate destELT = GetDestELT(Destination, Source);
            if (destELT != null)
            {
                Console.WriteLine("\r\nElement Template '{0}' already exists.",
Source.Name);
                Console.WriteLine("We will perform an
                update."); ImportELTAsUpdate(Source, destELT);
            }
            else
            {
                Console.WriteLine("Element Template '{0}' will be imported to
Database '{1}' as new.", Source.Name, Destination.Name);
                ImportELTAsNew(Source, Destination);
            }
        }
```

6. Directed Activity – Conclusion

6.1 Objective of Activity

- What do you think you can improve this app?

6.2 Additional Thoughts

- The creation of new Attributes Templates in the destination does not require much attention. The same is true for updates of existing Attribute Templates with new configurations.
- While preparing the sample code, we found that deleting Attribute Templates in the destination requires extra care. The Element Template in the source reflects its current configuration. There is no information about Attribute Templates which potentially have been deleted. It is necessary to find those Attribute Templates in the destination and to explicitly add them to the export definition with their names and “operation = delete”.
- References to other library content e.g. Attribute Categories do not work if the content doesn’t exist in the destination. To make this possible, it is likely required to care for those additional content separately e.g. running an export-import for Attribute Categories from source to destination before executing the Element Template export-import operation. Another thing which can be helpful in that regards is the **AllReferences** option as a member of the PIExportMode Enumeration. Modifying the existing code (method GetELTExport() in Exercise3.cs) and repeating Test 3 could turn out to be a low hanging fruit to gain certainty.
- Element Templates do not just consist of Attribute Templates which was the focus with our prototype. There also exist Analysis and Notification Templates. With those we also need to consider that the experience made with Attribute Templates applies. Creates and updates appear to be perfectly covered. Deletes are potentially problematic and will likely require comparing source and destination.
- It might be easier to delete Element Templates existent in the destination before importing “as new” from source. This will likely work if there is no content depending on the Element Template in the source. Delete operations will be problematic as soon as there exists dependent content e.g. derived Templates or Elements. PI Asset Framework uses unique identifiers (GUID’s) to uniquely identify an object. When deleting and re-creating an object, the new object will have a new unique identifier which means the relationship of derived objects is broken. This could be potentially addressed by removing the NoUniqueID export option which also is a member of the PIExportMode Enumeration. Testing would be again required for different use cases to ensure proper operation.
- The testing with our prototype assumed Templates in the destination not being checked out. The import will likely fail / not be performed when objects

are checked out. The purpose of synchronizing from a central PI System to site-specific installations is to make the central PI System the master which is used to define Templates according to corporate standards. Is it reasonable to allow modifications at the sites other than through the synchronization? It may make sense to use security settings to restrict access to Library objects i.e. protect them against modifications from local users.

- Generally, the whole subject of synchronizing site-specific PI System installations with a central PI System is possible but requires good planning and testing. The use cases and expectations likely differ from customer to customer. We recommend to clearly define use cases together with expected results and invest some efforts in testing against the documented expectations. The testing could be automated and should be done before putting an automated synchronization tool into place or upgrading the existent tool with a new release. Synchronizations should be one-directional only. It should generate a log which clearly documents the situation before and after the import. In addition, the export functionality should be used to backup what's in the destination before sending updated content.
- As there is no need of sending content which has not changed it makes sense to have an automated synchronization tool check for changed items. Utilizing the AFDatabase.FindChangedItems method for this purpose is recommended.
- Another topic to think about is the frequency of pushing updates. In the early phase of defining corporate standards, frequent changes are more likely. Pushing updated content multiple times a day may be required to allow sites adopting changes fast enough i.e. deriving Elements from Element Templates. Later sending updates once a day is likely enough.

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

Save the Date!

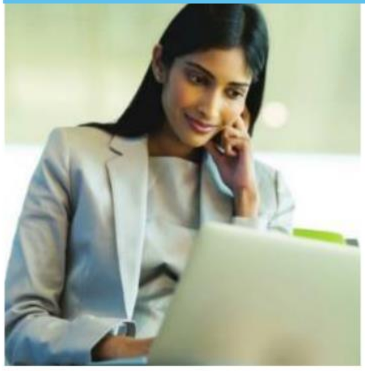
OSIsoft PI World Users Conference in Amsterdam; October 26-29, 2020.

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

OSIsoft.

PI SYSTEM LEARNING

CONTINUE YOUR PI SYSTEM LEARNING



After the conference, the PI SYSTEM LEARNING does not have to stop. All registered attendees for PI World SFO 2020 will have access to all PI World Hands-on Lab cloud environments for 21 days using the discount cod below. You will receive detailed instructions via email after the conference.

Discount Code: 2020PIWSF-LAB-100

Offer expires July 3, 2020