Best Practices for Building AF SDK Applications

Presented by **David Moler, Development Lead**

AF SDK: What is it and when should I use it?

- AF SDK is
  - A .NET library for Windows
  - Fastest way to get data from AF Server, PI Data Archive

- AF SDK is for
  - Reporting
  - Long-running, stateful services
  - Interactive, user driven applications/services

What can you tell me in 45 minutes?

- Optimization:
  - How to determine what is slow
  - Techniques for making things faster
- Design:
  - What to think about before writing code
  - What problems need to be addressed in architecture

Optimization

- Don’t waste effort
  - Don’t do work that isn’t necessary
  - Avoid per-call overhead with bulk calls
- Minimize time spent waiting
  - Concurrency of requests
  - Continue processing as data becomes available

Tools for data driven optimization

- RPC Metrics (RPC = Remote Procedure Call)
  - How many calls are being made?
  - Is most time spent processing on server or client?
  - Can the calls be bulked up?
  - Can the results be cached?
- Client profiling
  - Where is time spent?
  - Can work be avoided or done in parallel?

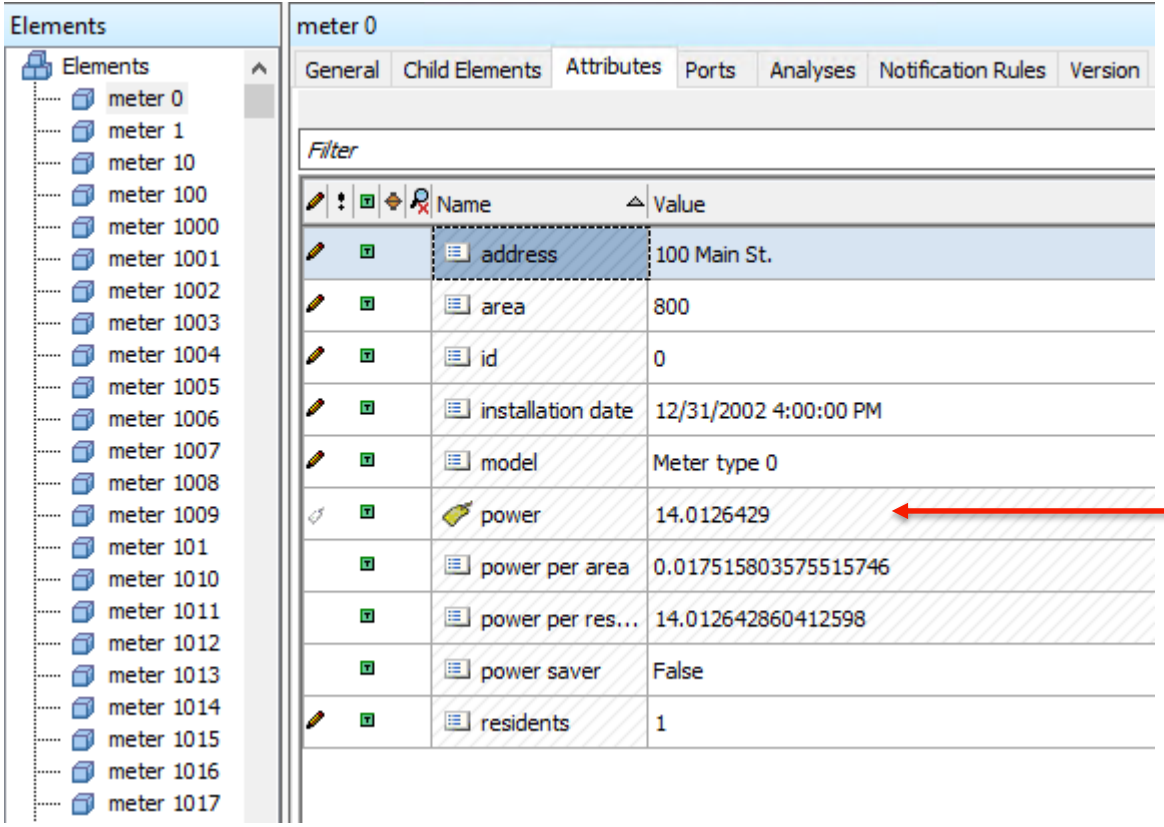
Measuring RPC Metrics

```
AFRpcMetric[] serverRpcs = piSystem.GetRpcMetrics();
AFRpcMetric[] clientRpcs = piSystem.GetClientRpcMetrics();
AFRpcMetric[] piRpcs = piServer.GetClientRpcMetrics();

var sw = Stopwatch.StartNew();
action();
sw.Stop();

var piDiff = AFRpcMetric.SubtractList(piServer.GetClientRpcMetrics(), piRpcs);
var clientDiff = AFRpcMetric.SubtractList(piSystem.GetClientRpcMetrics(), clientRpcs);
var serverDiff = AFRpcMetric.SubtractList(piSystem.GetRpcMetrics(), serverRpcs);
```

Example: Reporting on Elements

Reporting on Elements

For all meters…



Compute the total power

Straightforward Implementation (slow)

```
// define a search for elements from a template
AFElementSearch search = new AFElementSearch(db, "",
    "Template:'meter template'");

// enumerate the matches
foreach(var meter in search.FindElements())
{
    // query for the end-of-stream value
    var powerValue = meter.Attributes["power"].Data.EndOfStream(null);

    // if the value is good, add it to the total
    if (powerValue.IsGood)
        total += powerValue.ValueAsSingle();
}
```

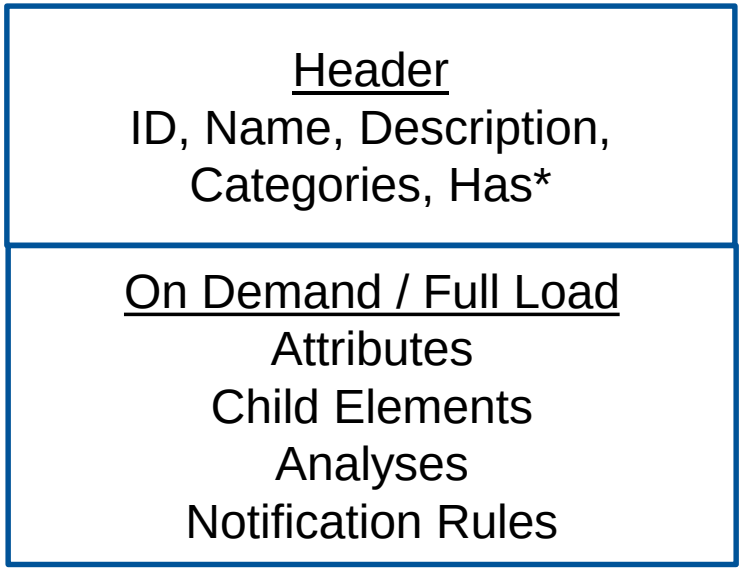
Straightforward Implementation (slow)

```
 Type      Elapsed    Count       Name
PI Client:  16,436 (10,000) getendofstreams|pisnapss|1

AF Client:      55      (1) GetElementTemplateList
AF Client:   5,155     (11) SearchElements
AF Client:  54,270 (10,000) GetElement
AF Client:      39      (1) GetElementTemplate
AF Client:      53      (1) GetCategory

AF Server:      12      (1) GetElementTemplateList
AF Server:       2      (1) GetCategory
AF Server:       4      (1) GetElementTemplate
AF Server:  37,606 (10,000) GetElement
AF Server:   4,365     (11) SearchElements

Elapsed time: 01:24.601
```

One-at-a-time loading of elements is biggest cost

Loading Objects in AF

- Header holds limited info
- Fast to retrieve, load
- Full information available on-demand or via “full load”




```
foreach(var meter in search.FindElements())
{
    // query for the end-of-stream value
    var powerValue = meter.Attributes["power"].Data.EndOfStream(null);

    // …
}
```

Full Load (better, but still slow)

```
// define a search for elements from a template
AFElementSearch search = new AFElementSearch(db, "",
    "Template:'meter template'");

// enumerate the matches
foreach(var meter in search.FindElements(fullLoad: true))
{
    // query for the end-of-stream value
    var powerValue = meter.Attributes["power"].Data.EndOfStream(null);

    // if the value is good, add it to the total
    if (powerValue.IsGood)
        total += powerValue.ValueAsSingle();
}
```

Full load avoids RPC when accessing attribute

Already loaded

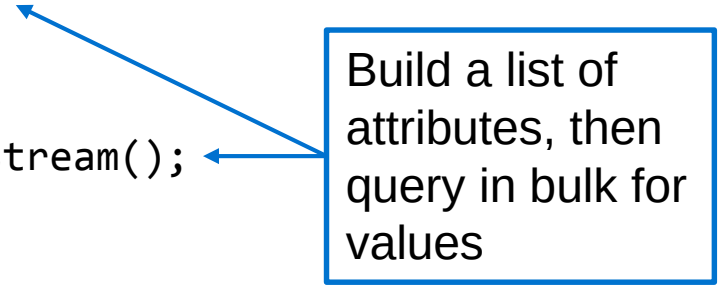
Full Load (better, but still slow)

```
 Type      Elapsed    Count       Name
PI Client:   14,192 (10,000) getendofstreams|pisnapss|1

AF Client:       89      (1) GetElementTemplateList
AF Client:    4,250     (11) SearchObjectIds
AF Client:    8,840     (10) GetModels
AF Client:      114      (2) GetCategoryList
AF Client:       67      (1) GetElementTemplates

AF Server:       11      (1) GetElementTemplateList
AF Server:       29      (2) GetCategoryList
AF Server:    5,281     (10) GetModels
AF Server:       12      (1) GetElementTemplates
AF Server:    3,989     (11) SearchObjectIds

Elapsed time: 00:33.998
```

One-at-a-time query of PIPoint is the largest cost

Bulk Load & Bulk Query (faster)

```
AFElementSearch search = new AFElementSearch(db, "",
    "Template:'meter template'");

AFAttributeList attributes = new AFAttributeList();
foreach (var meter in search.FindElements(fullLoad: true))
{
    attributes.Add(meter.Attributes["power"]);
}

IList<AFValue> values = attributes.Data.EndOfStream();
foreach(AFValue powerValue in values)
{
    if (powerValue.IsGood)
        total += powerValue.ValueAsSingle();
}
```



Bulk Load & Bulk Query (faster)

```
 Type      Elapsed    Count       Name
PI Client:      112       (1) getendofstreams|pisnapss|1

AF Client:       57       (1) GetElementTemplateList
AF Client:    3,489      (11) SearchObjectIds
AF Client:    9,322      (10) GetModels
AF Client:      130       (2) GetCategoryList
AF Client:       93       (1) GetElementTemplates

AF Server:       15       (1) GetElementTemplateList
AF Server:       14       (2) GetCategoryList
AF Server:    5,610      (10) GetModels
AF Server:       14       (1) GetElementTemplates
AF Server:    3,354      (11) SearchObjectIds
```

Loading and initializing elements is consuming most of the time

```
Elapsed time: 00:17.214
```

Bulk Load & Bulk Query (faster)

```
 Type      Elapsed   Count      Name
PI Client:     112       (1) getendofstreams|pisnapss|1

AF Client:      57       (1) GetElementTemplateList
AF Client:   3,489      (11) SearchObjectIds
AF Client:   9,322      (10) GetModels
AF Client:     130       (2) GetCategoryList
AF Client:      93       (1) GetElementTemplates

AF Server:      15       (1) GetElementTemplateList
AF Server:      14       (2) GetCategoryList
AF Server:   5,610      (10) GetModels
AF Server:      14       (1) GetElementTemplates
AF Server:   3,354      (11) SearchObjectIds

Elapsed time: 00:17.214
```

Search time is all on server. Caching can speed this.

Cache search result across pages

```
AFElementSearch search = new AFElementSearch(db, "",
    "Template:'meter template'");
search.CacheTimeout = TimeSpan.FromMinutes(1);
```

- Server caches result until timeout elapses since accessed
- Snapshot in time of search result
- Better performance when entire result set will be accessed
- Can Refresh()/Dispose() to clear cache

Cache search result across pages (faster, efficient)

```
 Type      Elapsed    Count       Name
PI Client:       85       (1) getendofstreams|pisnapss|1

AF Client:       83       (1) GetElementTemplateList
AF Client:      911      (10) SearchObjectIds
AF Client:    8,616      (10) GetModels
AF Client:       62       (2) GetCategoryList
AF Client:       62       (1) GetElementTemplates

AF Server:       19       (1) GetElementTemplateList
AF Server:       24       (2) GetCategoryList
AF Server:    5,169      (10) GetModels
AF Server:        8       (1) GetElementTemplates
AF Server:      796      (10) SearchObjectIds
```

Elapsed time: 00:13.907

Caching reduced time by 3 seconds

GetModels (full load) requires lots of client processing.

Multithreaded Load and Query (fastest)

```
private static Task<IList<AFValue>> GetValuesAsync(
    PISystem piSystem, Guid[] ids, string attributeName)
{
    // start a background task to load elements and query attributes
    return Task.Run(() =>
    {
        // load the elements with the specified IDs
        var elements = AFElement.LoadElements(piSystem, ids, queryDate: null);

        // create a list of attributes to query
        var attributeList = new AFAttributeList(
                                    elements.Select(e => e.Attributes["power"]));

        // return the end of stream values for each attribute
        return attributeList.Data.EndOfStream();
    });
}
```

Multithreaded Load and Query (fastest)

```
List<Guid> ids = new List<Guid>();
List<Task<IList<AFValue>>> tasks = new List<Task<IList<AFValue>>>();
foreach (var meterId in search.FindObjectIds(pageSize: PageSize))
{
    ids.Add(meterId);
    if (ids.Count == PageSize)
    {
        tasks.Add(GetValuesAsync(piSystem, ids.ToArray(), "power"));
        ids.Clear();
    }
}

if (ids.Count != 0)
    tasks.Add(GetValuesAsync(piSystem, ids.ToArray(), "power"));
```

Load chunks of elements on another thread

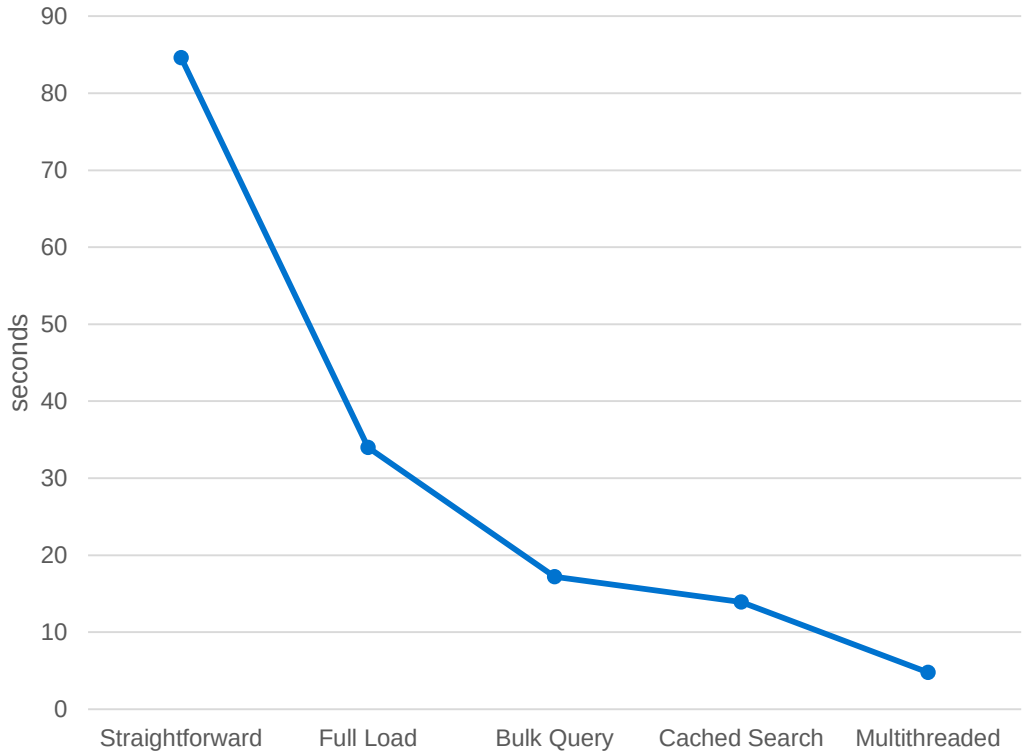
Multithreaded Load and Query (fastest)

```
 Type      Elapsed   Count       Name
PI Client:      179       (5) getendofstreams|pisnapss|1

…
AF Client:      894       (5) SearchObjectIds
AF Client:   12,549      (10) GetModels
AF Client:       37       (2) GetCategoryList
AF Client:       98       (2) GetElementTemplates

…
AF Server:    7,730      (10) GetModels
AF Server:       15       (2) GetElementTemplates
AF Server:      795       (5) SearchObjectIds
```

RPC time is greater than elapsed time because RPCs were made concurrently and this measures the aggregate time.

```
Elapsed time: 00:04.777
```

Reporting on Elements



- 18 times faster
- Further optimizations in specific cases

Reporting on Elements

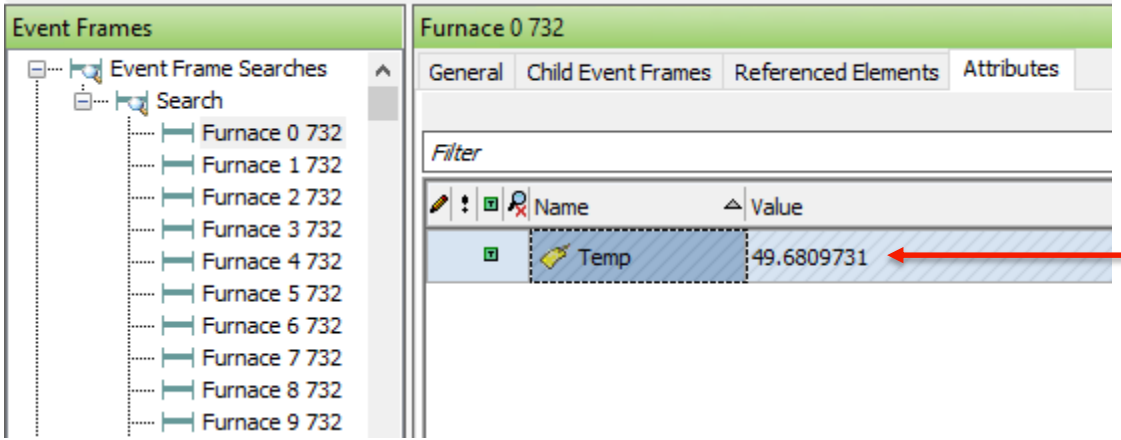
- Optimizations
  - Many RPCs? Make bulk calls
  - Long search time? Cache search result
  - Long client processing time? Chunk & parallelize

- Let data drive optimizations!

Example: Aggregating Event Frames

Aggregating Event Frames

For each furnace event…



Compute the average temperature

Event Frame Report

```
AFEventFrameSearch search = new AFEventFrameSearch(db, "",
    "Template:batch Start:> 21-feb-2016");
search.CacheTimeout = TimeSpan.FromSeconds(10);

foreach (var batch in search.FindEventFrames(fullLoad: true))
{
    // event frame values are captured so can query directly
    var tempValue = batch.Attributes["temp"].GetValue();
    if (tempValue.IsGood)
    {
        ++count;
        average += (tempValue.ValueAsSingle() - average) / count;
    }
}
```

Event Frame Report

```
 Type      Elapsed   Count       Name
AF Client:       57       (1) GetElementTemplateList
AF Client:      548      (10) SearchObjectIds
AF Client:    3,037      (10) GetEventFrames
AF Client:       29       (2) GetCategoryList
AF Client:       50       (2) GetElementTemplates
AF Client:    2,957      (10) GetModels

AF Server:       16       (1) GetElementTemplateList
AF Server:        5       (2) GetCategoryList
AF Server:    1,313      (10) GetEventFrames
AF Server:    1,259      (10) GetModels
AF Server:       16       (2) GetElementTemplates
AF Server:      446      (10) SearchObjectIds
```

```
Elapsed time: 00:10.558
```

Light-Weight Search

- Request specific fields for search matches
- Bring back only required data from SQL
- Send only requested data to client
- Does not load SDK objects

Light-Weight Search

```
foreach (var fieldList in search.FindObjectFields("|temp"))
{
    var tempValue = (AFValue)fieldList[0];

    if (tempValue.IsGood)
    {
        ++count;
        average += (tempValue.ValueAsSingle() - average) / count;
    }
}
```

Choose which fields to bring back.
Can also get DTO.

Light-Weight Search

```
 Type      Elapsed   Count       Name
AF Client:       53       (1) GetElementTemplateList
AF Client:    2,187      (11) SearchObjectFields

AF Server:       12       (1) GetElementTemplateList
AF Server:    1,686      (11) SearchObjectFields
```

```
Elapsed time: 00:02.284
```

Search Aggregates

- Queries necessary fields
- Handles type conversions, UOM, calculating summaries
- Can group discrete values
- Can bin continuous values
- Can bulk up several aggregates in one request

```
AFSummaryResult summary = search.Summary("|temp", AFSummaryTypes.Average);

average = summary.SummaryResults[AFSummaryTypes.Average].ValueAsSingle();
```

Example: Long-running Service

Long-running Services

- Faster response times
- Higher throughput
- Amortized initialization cost
- AFDataCache can efficiently sync with PI Data Archive
- Query cached data via AFData object

Using the Data Cache

```
IEnumerable<AFElement> elements = new AFElementSearch(db, "",
    "Template:'meter template'").FindElements(fullLoad: true);
var attributes = elements.Select(e => e.Attributes["power"]).ToList();

using (AFDataCache cache = new AFDataCache())
{
    var signups = cache.Add(attributes);

    // periodically poll
    cache.UpdateData();
```

Using the Data Cache

```
foreach (AFData cachedData in signups)
{
    var powerValue = cachedData.EndOfStream(desiredUOM: null);
    if (powerValue.IsGood)
        total += powerValue.ValueAsSingle();
}
```

Cache-enabled AFData

```
 Type      Elapsed   Count       Name
PI Client:       3       (1) getevents|piupdmgr|1
```

```
Elapsed time: 00:00.050
```

RPC to poll data pipe

Using the Data Cache with Metadata Changes

- Caching permits vastly improved speed
- Need to stay up-to-date to be correct

```
// collect changes
var changes = db.FindChangedItems(false, MaxChanges, updateCookie, out updateCoo

// let data cache observer changes before refreshing
var updateToken = cache.ObservePendingChanges(changes);

AFChangeInfo.Refresh(piSystem, changes); // perform refresh

// update cache
var signupChanges = cache.ProcessAppliedChanges(updateToken);
```

Using the Data Cache with Metadata Changes

```
foreach (var change in changes.Where(c => c.Identity == AFIdentity.Element))
{
    if (change.Action == AFChangeInfoAction.Added
        || (change.Action == AFChangeInfoAction.Updated
                && !elementLookup.ContainsKey(change.ID)))
    { // look for new additions
        AFElement element = AFElement.FindElement(piSystem, change.ID);
        if (search.IsMatch(element))
            attributesToAdd.Add(element.Attributes["power"]);
    }
    else if (change.Action == AFChangeInfoAction.Updated
                && elementLookup.ContainsKey(change.ID))
    { // look for attributes to remove
        if (!search.IsMatch(elementLookup[change.ID]))
            attributesToRemove.Add(elementLookup[change.ID].Attributes["power"]);
    }
}
```

Using the Data Cache with Metadata Changes

```
 Type      Elapsed   Count       Name
PI Client:        2       (1) getevents|piupdmgr|1

AF Client:        3       (1) FindChangesRID

AF Server:        0       (1) FindChangesRID_NoChange

Elapsed time: 00:00.042
```

Optimization Summary

- Measure first!
  - RPC Metrics
  - Use timings or CPU profiling
- Eliminate what you can
- Be efficient with what needs to be done
- Data/Metadata reads are thread safe so work can be parallelized

Design

Reporting

- Daily report, batch import/export of data

- Runtime often dominated by startup costs
- Work backward from the queries that need to be made
  - What needs to be available to make those queries?

Interactive

- PI System Explorer, PI WebAPI

- Security model (for multi-user services)
- How aggressively should data be cached/trimmed?

Long-running, stateful services

- Asset Analytics, Notifications, PI Integrators

- Things that should be fast or responsive should be cached
- Handling metadata changes should be part of architecture

Contact Information

David Moler
dmoler@osisoft.com
Development Lead
OSIsoft, LLC

Questions

Please wait for the **microphone** before asking your questions

State your **name & company**

Please remember to…

Complete the Online Survey for this session

http://bit.ly/uc2017-app

감사합니다

谢谢

Danke

Merci

Gracias

Thank You

ありがとう

Спасибо

Obrigado