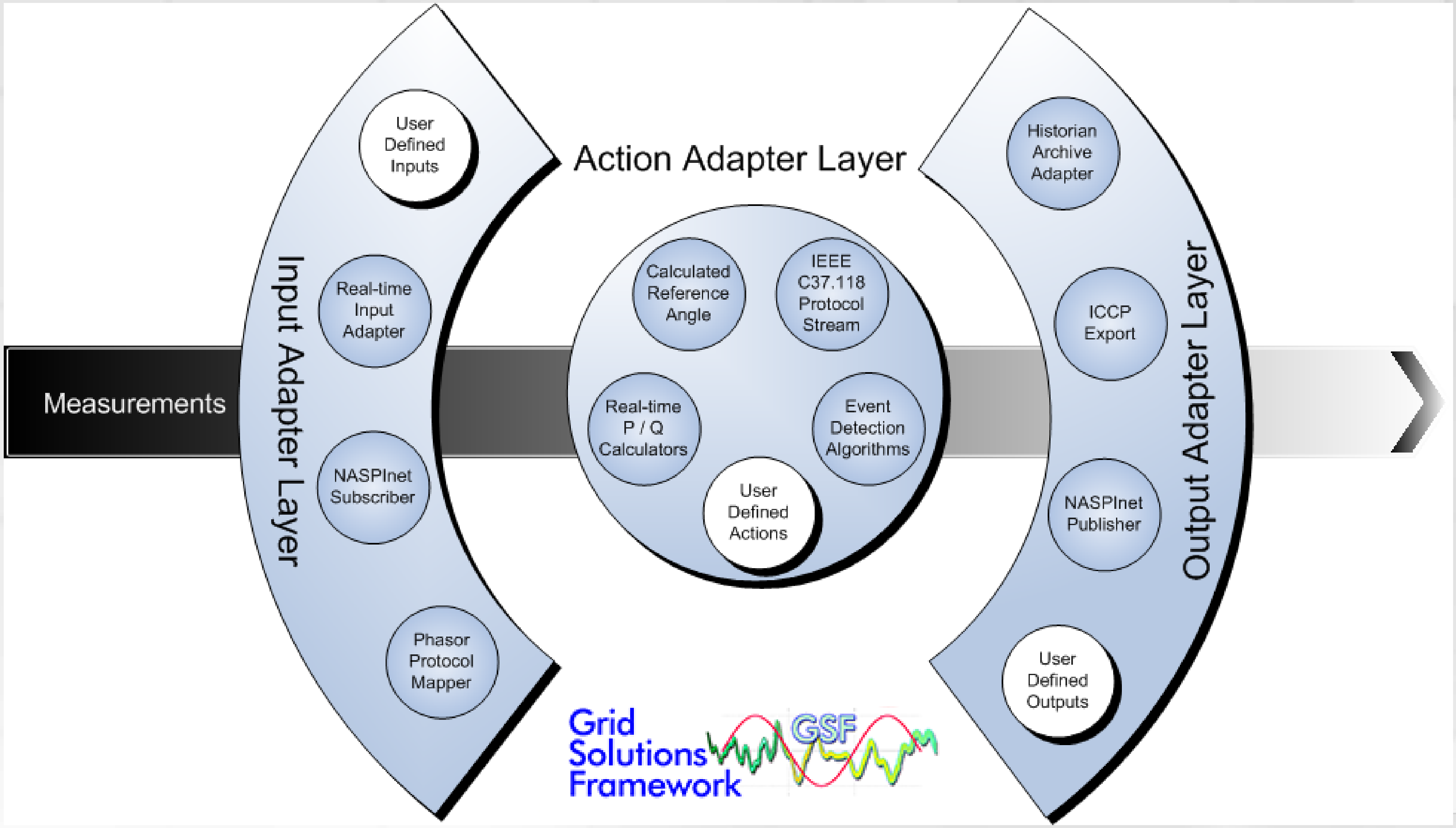
Custom Adapter Development

Tutorial Session 3

J. Ritchie Carroll

August 12, 2014

Overview of the Adapter Architecture Layer



GRID PROTECTION ALLIANCE

Input Adapters

- InputAdapterBase
  - Initialize
  - AttemptConnection
  - AttemptDisconnection
  - GetShortStatus
  - UseAsyncConnect

GRID PROTECTION ALLIANCE

Action Adapters

- ActionAdapterBase
  - Initialize
  - PublishFrame
  - GetShortStatus
- FacileActionAdapterBase
  - Initialize
  - QueueMeasurementsForProcessing
  - GetShortStatus

Output Adapters

- OutputAdapterBase
  - Initialize
  - AttemptConnection
  - AttemptDisconnection
  - ProcessMeasurements
  - GetShortStatus
  - UseAsyncConnect
  - OutputIsForArchive

Lifecycle

- Initialize
- Start (AttemptConnection)
- Stop (AttemptDisconnection)
- Dispose

Connection Strings

- Initialize
  - (Example connection string and Initialize code sample)
- Property
  - ConnectionStringParameterAttribute
  - DefaultValue and Description
  - CustomConfigurationEditor

To be facile or not to be facile

- Action Adapter
  - Concentrated
- Facile Action Adapter
  - Not concentrated, but each individual signal stays in order

Input Measurement Keys and Output Measurements

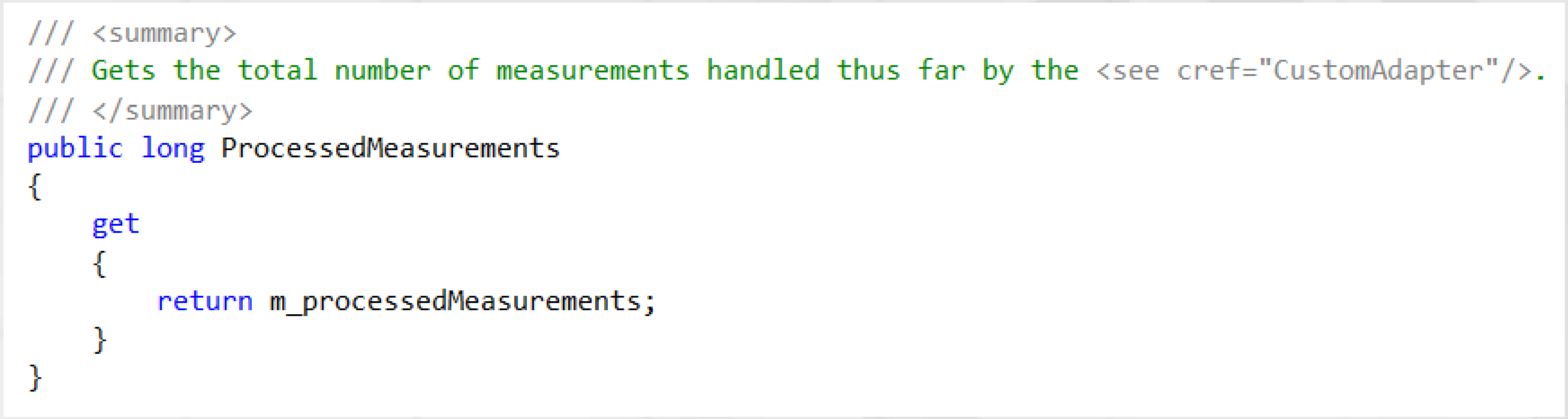
- **Measurement Keys:**
  - PPA:12; PPA:15; STAT:21
- **Guids:**
  - E5E4EE01-B3D2-4FC3-B39C-03478F0BA2B9; 5BE1FB5A-412F-4338-9BC8-08BB701221BB
- **Point Tags:**
  - TVA_SHELBY:ABBF; TVA_SHELBY-CORD:ABBI
- **Filter Expression:**
  - FILTER *ActiveMeasurements* WHERE *SignalType* = ‘FREQ'

Custom Statistics

1. Expose your statistics
2. Define a category
3. Add database records
4. Register with the statistics engine
5. Data operations

Expose Your Statistics

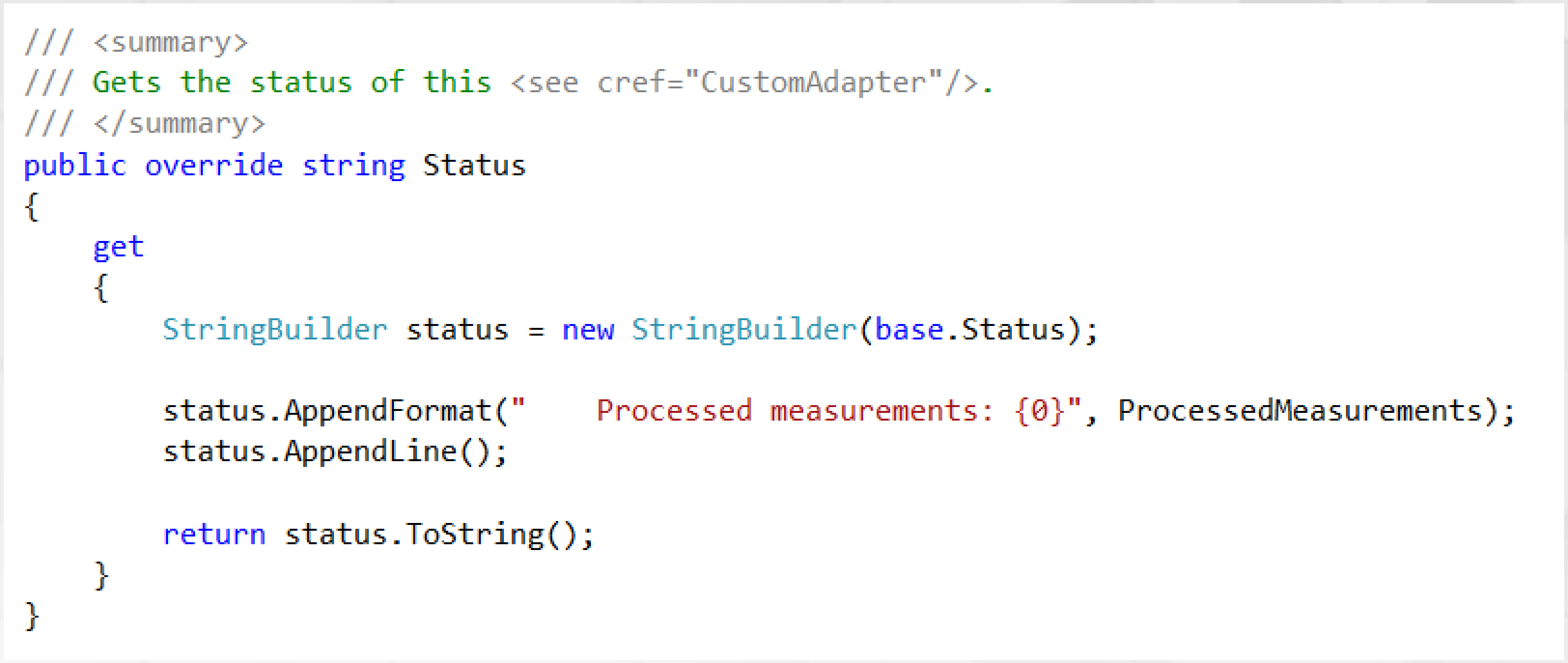
- Create a public property



```
/// <summary>
/// Gets the total number of measurements handled thus far by the <see cref="CustomAdapter"/>.
/// </summary>
public long ProcessedMeasurements
{
    get
    {
        return m_processedMeasurements;
    }
}
```

Expose Your Statistics (cont.)

- The Status property



```csharp
/// <summary>
/// Gets the status of this <see cref="CustomAdapter"/>.
/// </summary>
public override string Status
{
    get
    {
        StringBuilder status = new StringBuilder(base.Status);

        status.AppendFormat("    Processed measurements: {0}", ProcessedMeasurements);
        status.AppendLine();

        return status.ToString();
    }
}
```

Exposing Your Statistics (cont.)

- Statistic Calculation Function

```
private static double GetCustomStatistic_ProcessedMeasurements(object source, string arguments)
{
    double statistic = 0.0D;
    CustomAdapter adapter = source as CustomAdapter;

    if ((object)adapter != null)
        statistic = adapter.ProcessedMeasurements;

    return statistic;
}
```

Define a Category

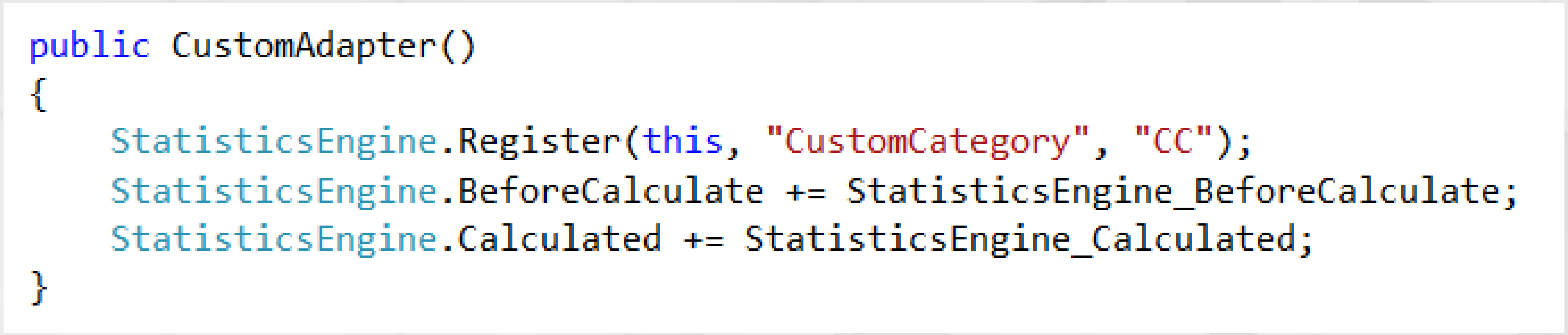
- Source type
  - Your custom adapter
- Category name
  - InputStream, OutputStream, System, etc.
- Category acronym
  - IS, OS, SYSTEM, etc.

GRID PROTECTION ALLIANCE

Add Database Records

- Statistic table
  - This defines the “type” of your statistic, such as “Processed Measurements”. If your custom adapter produces n statistics, n records will be added to this table.
- Measurement table
  - Defines an instance of the types defined in the Statistic table. If you have defined m instances of your custom adapter, n*m records will be added to this table (one per statistic per adapter).

Register with the Statistics Engine



```
public CustomAdapter()
{
    StatisticsEngine.Register(this, "CustomCategory", "CC");
    StatisticsEngine.BeforeCalculate += StatisticsEngine_BeforeCalculate;
    StatisticsEngine.Calculated += StatisticsEngine_Calculated;
}
```

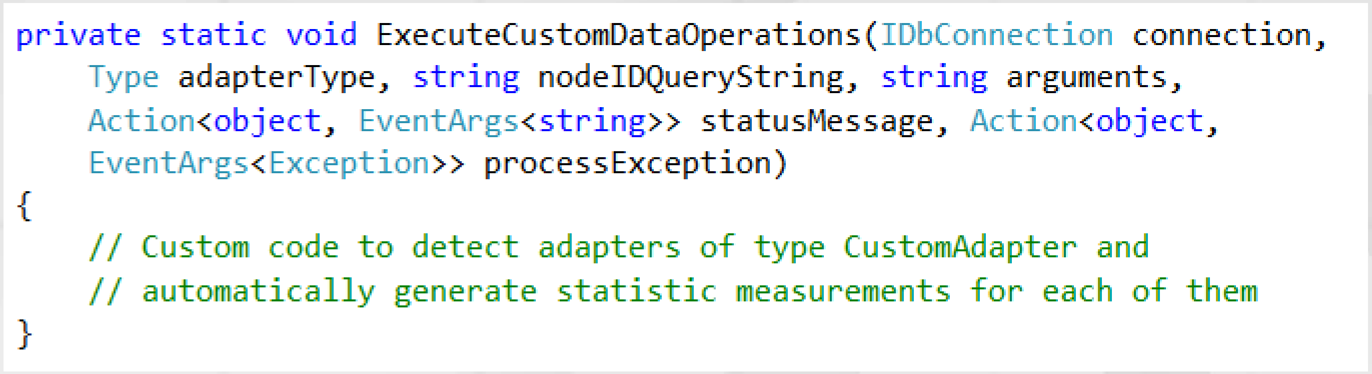
Data Operations

- Data operations are a customizable set of functions that are run before loading configuration. This makes them useful for handling automated configuration manipulation.

GRID PROTECTION ALLIANCE

Data Operations (cont.)

- Customize some automation



```
private static void ExecuteCustomDataOperations(IDbConnection connection,
    Type adapterType, string nodeIDQueryString, string arguments,
    Action<object, EventArgs<string>> statusMessage, Action<object,
    EventArgs<Exception>> processException)
{
    // Custom code to detect adapters of type CustomAdapter and
    // automatically generate statistic measurements for each of them
}
```

- Note that samples of similar code written for existing statistics can be found in:
  - **GSF.TimeSeries.TimeSeriesStartupOperations.PerformTimeSeriesStartupOperation**
  - **PhasorProtocolAdaters.CommonPhasorServices.PhasorDataSourceValidation**.

Data Operations (cont.)

- DataOperation table
  - Description
  - AssemblyName
  - TypeName
  - MethodName
  - LoadOrder
  - Enabled

Hands-on: Create and Debug an Adapter

- Download host code (e.g., openPDC)
- Setup host project configuration
- Test running in debug mode
- Add a new adapter project
- Inherit from desired Iaon base class
- Add code / run / break / debug