GRID PROTECTION ALLIANCE

Working with openXDA

openXDA
EXTENSIBLE DISTURBANCE ANALYTICS

GPA User's Forum 2015

Atlanta, Georgia

Objective

- To provide a deep dive into openXDA’s components and constructs to demonstrate openXDA’s adaptability and ease of addition of new processes.

GRID PROTECTION ALLIANCE

Overview

What is openXDA?

- A back-office automated service to automatically process and analyze event and trending data from transmission and distribution metering – DFRs and PQ metering

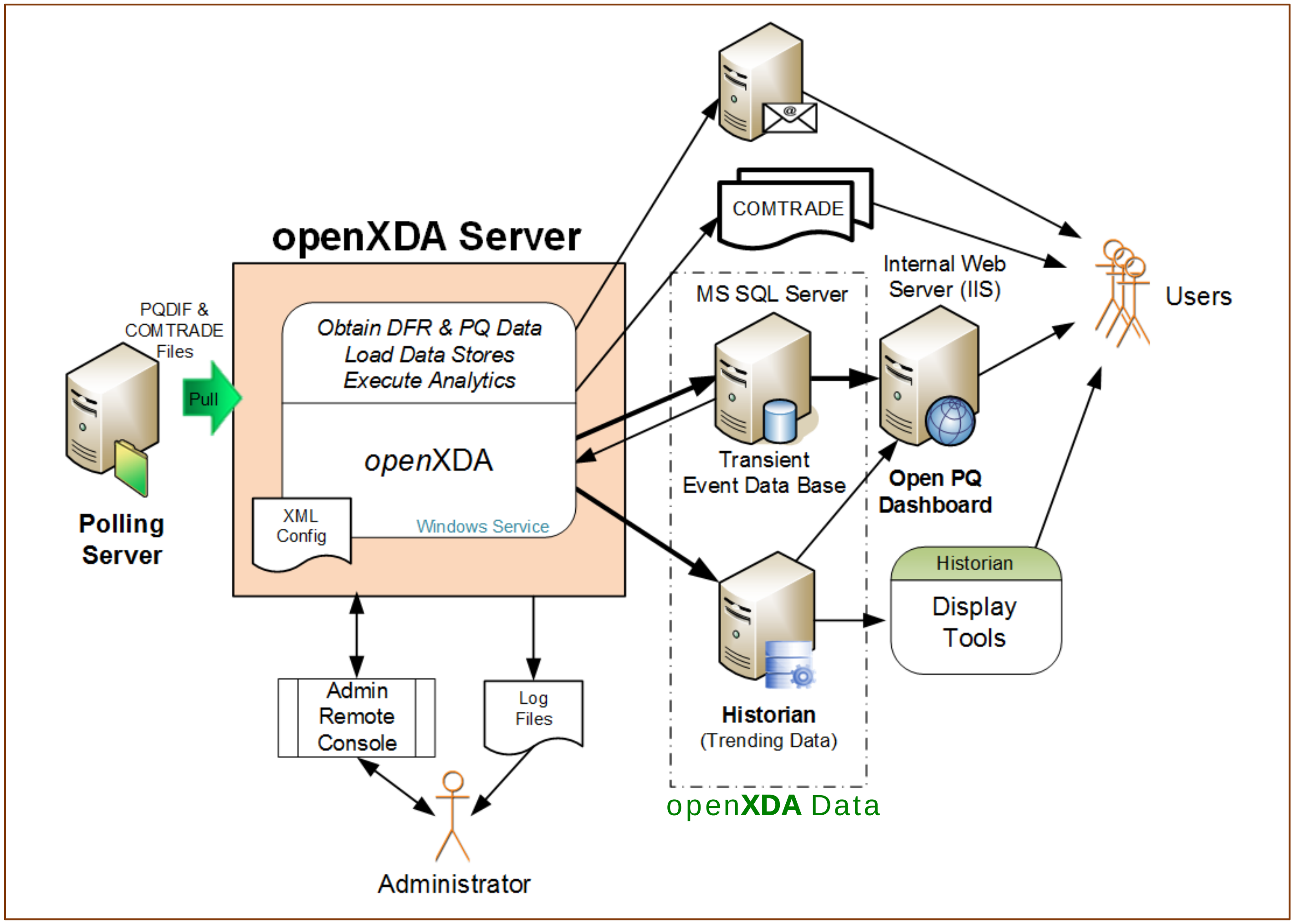
openXDA – PQ / DFR Dualism

- **PQ** – Largely distribution focused
  - Single line monitoring
  - Self-describing configuration in PQDIF
  - Data may be collected by MDM systems
  - Event and trending data
- **DFR** – Largely transmission focused
  - Multiple lines monitored
  - Meter configuration information (channel mapping) required
  - Only event data, typically as COMTRADE files

openXDA Components

- Installer for the openXDA service
- Core service (includes File Watcher, logging and notification components)
- Administrator’s remote console
- Database (MS SQL Server)
- Configuration management / loader tools

openXDA Overview



openXDA Inputs

- Configuration Data
  - Meter name and location
  - Meter channel definitions
  - Line parameters
- Waveform Data
  - COMTRADE
  - PQDIF
  - EMAX (native format)
  - SEL .eve (SEL-251, SEL-351, Sel-421 relays)

Event Data

- **Time Domain**
  - Event attributes
  - Event segments attributes
  - Waveform
- **Frequency Domain**
  - Cycle data – Full-set of RMS and synchronous component values for each full cycle of data on the waveform

Event Analysis Data

- **Sags/Swells**
  - Duration
  - Magnitude
- **Faults**
  - Type
  - Inception time
  - Duration
  - Distance
  - Prefault current
  - Fault current
  - Postfault current

Trending Data

- **Daily Values**
  - Min, Max, Average
- **Hourly Values**
  - Min, Max, Average
- **Full Resolution Values**
  - Min, Max, Average

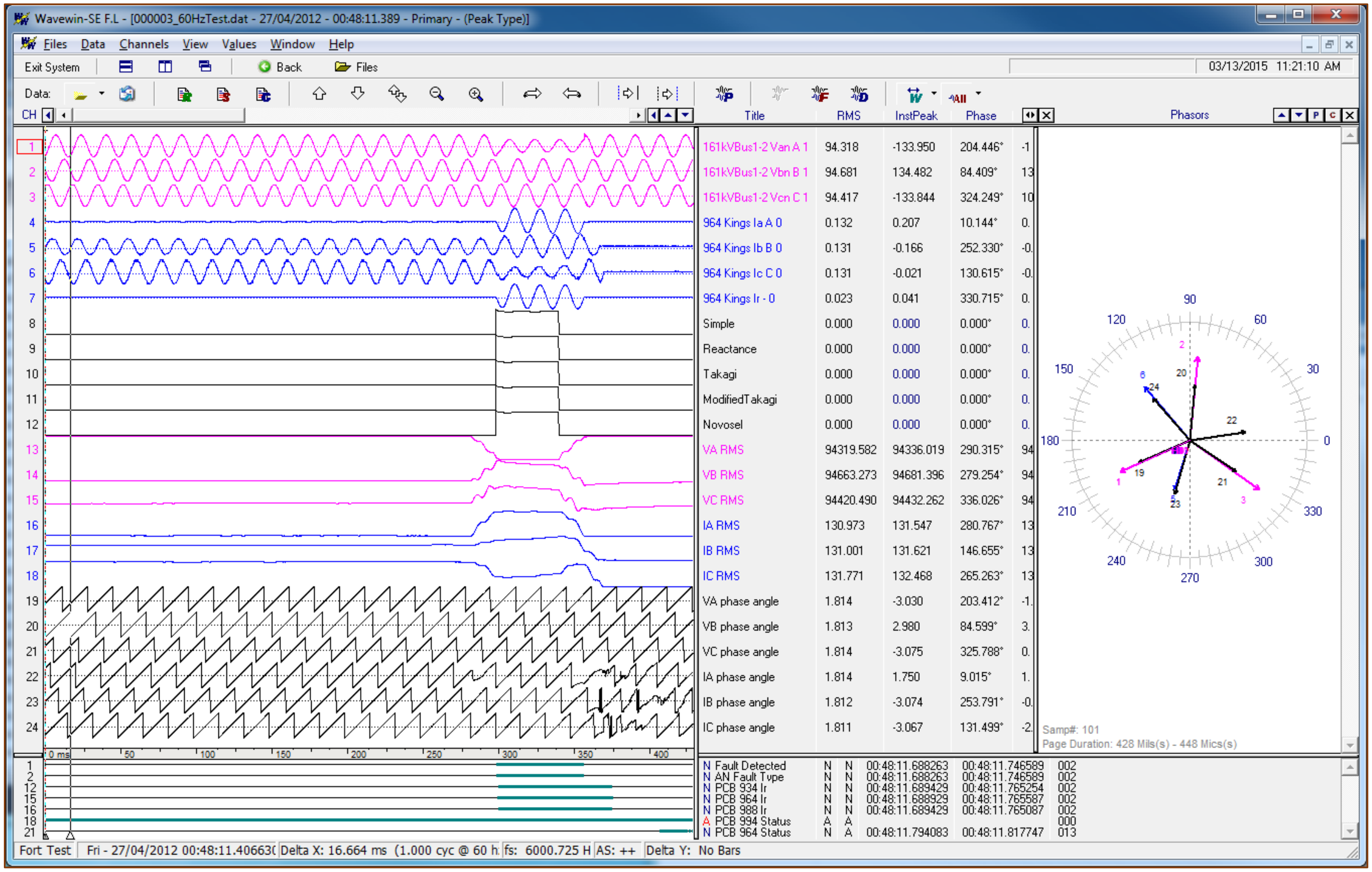
Trending Alarm Data

- **Data Quality –** Engineering reasonableness
  - Latched
  - Unreasonable (high/low limits)
  - Incongruent (max > average > min)
- **Off-Normal –** Hour-of-week 3 sigma excursions
- **Custom Alarms**

openXDA Outputs

- Analytic results saved in data base
- Automated notifications
- COMTRADE
  - Line centric
  - Includes both input and analytics cycle data

COMTRADE Output is Line Centric

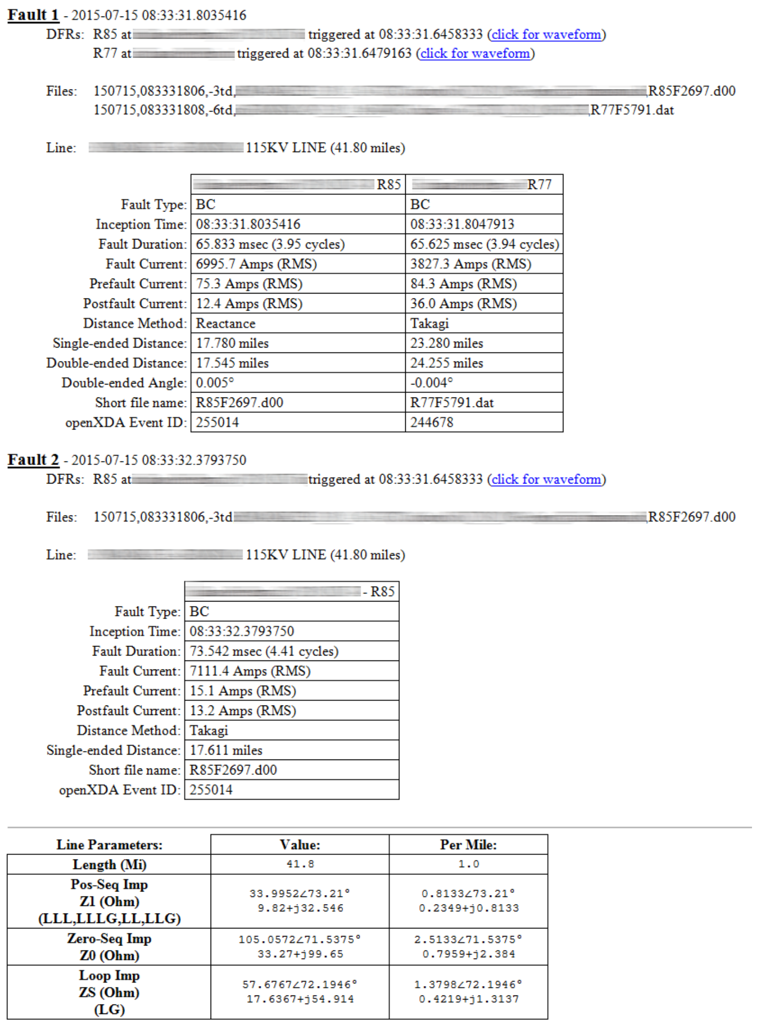

Wavewin-SE F.L - [000003_60HzTest.dat - 27/04/2012 - 00:48:11.389 - Primary - (Peak Type)]
Files Data Channels View Values Window Help
Exit System
Back
Files
03/13/2015 11:21:10 AM
Data:
CH
Phasors
0 ms 50 100 150 200 250 300 350 400
90 60 30 0 330 300 270 240 210 180 150 120
Samp#: 101
Page Duration: 428 Mils(s) - 448 Mics(s)

Title	RMS	InstPeak	Phase	
161kVBus1-2 Van A 1	94.318	-133.950	204.446°	-1
161kVBus1-2 Vbn B 1	94.681	134.482	84.409°	13
161kVBus1-2 Vcn C 1	94.417	-133.844	324.249°	10
964 Kings Ia A 0	0.132	0.207	10.144°	0.
964 Kings Ib B 0	0.131	-0.166	252.330°	-0.
964 Kings Ic C 0	0.131	-0.021	130.615°	-0.
964 Kings Ir - 0	0.023	0.041	330.715°	0.
Simple	0.000	0.000	0.000°	0.
Reactance	0.000	0.000	0.000°	0.
Takagi	0.000	0.000	0.000°	0.
ModifiedTakagi	0.000	0.000	0.000°	0.
Novosel	0.000	0.000	0.000°	0.
VA RMS	94319.582	94336.019	290.315°	94
VB RMS	94663.273	94681.396	279.254°	94
VC RMS	94420.490	94432.262	336.026°	94
IA RMS	130.973	131.547	280.767°	13
IB RMS	131.001	131.621	146.655°	13
IC RMS	131.771	132.468	265.263°	13
VA phase angle	1.814	-3.030	203.412°	-1.
VB phase angle	1.813	2.980	84.599°	3.
VC phase angle	1.814	-3.075	325.788°	0.
IA phase angle	1.814	1.750	9.015°	1.
IB phase angle	1.812	-3.074	253.791°	-0.
IC phase angle	1.811	-3.067	131.499°	-2.

```
N Fault Detected    N  N  00:48:11.688263  00:48:11.746589  002
N AN Fault Type     N  N  00:48:11.688263  00:48:11.746589  002
N PCB 934 Ir        N  N  00:48:11.689429  00:48:11.765254  002
N PCB 964 Ir        N  N  00:48:11.688929  00:48:11.765587  002
N PCB 988 Ir        N  N  00:48:11.689429  00:48:11.765087  002
A PCB 994 Status    A  A                                    000
N PCB 964 Status    N  A  00:48:11.794083  00:48:11.817747  013
```

Fort Test | Fri - 27/04/2012 00:48:11.40663(| Delta X: 16.664 ms (1.000 cyc @ 60 h | fs: 6000.725 H | AS: ++ | Delta Y: No Bars

Version 1.3 Example Email



Fault 1 - 2015-07-15 08:33:31.8035416

DFRs: R85 at [redacted] triggered at 08:33:31.6458333 (click for waveform)
R77 at [redacted] triggered at 08:33:31.6479163 (click for waveform)

Files: 150715,083331806,-3td,[redacted]R85F2697.d00
150715,083331808,-6td,[redacted]R77F5791.dat

Line: [redacted] 115KV LINE (41.80 miles)

	[redacted] R85	[redacted] R77
Fault Type:	BC	BC
Inception Time:	08:33:31.8035416	08:33:31.8047913
Fault Duration:	65.833 msec (3.95 cycles)	65.625 msec (3.94 cycles)
Fault Current:	6995.7 Amps (RMS)	3827.3 Amps (RMS)
Prefault Current:	75.3 Amps (RMS)	84.3 Amps (RMS)
Postfault Current:	12.4 Amps (RMS)	36.0 Amps (RMS)
Distance Method:	Reactance	Takagi
Single-ended Distance:	17.780 miles	23.280 miles
Double-ended Distance:	17.545 miles	24.255 miles
Double-ended Angle:	0.005°	-0.004°
Short file name:	R85F2697.d00	R77F5791.dat
openXDA Event ID:	255014	244678

Fault 2 - 2015-07-15 08:33:32.3793750

DFRs: R85 at [redacted] triggered at 08:33:31.6458333 (click for waveform)

Files: 150715,083331806,-3td[redacted]R85F2697.d00

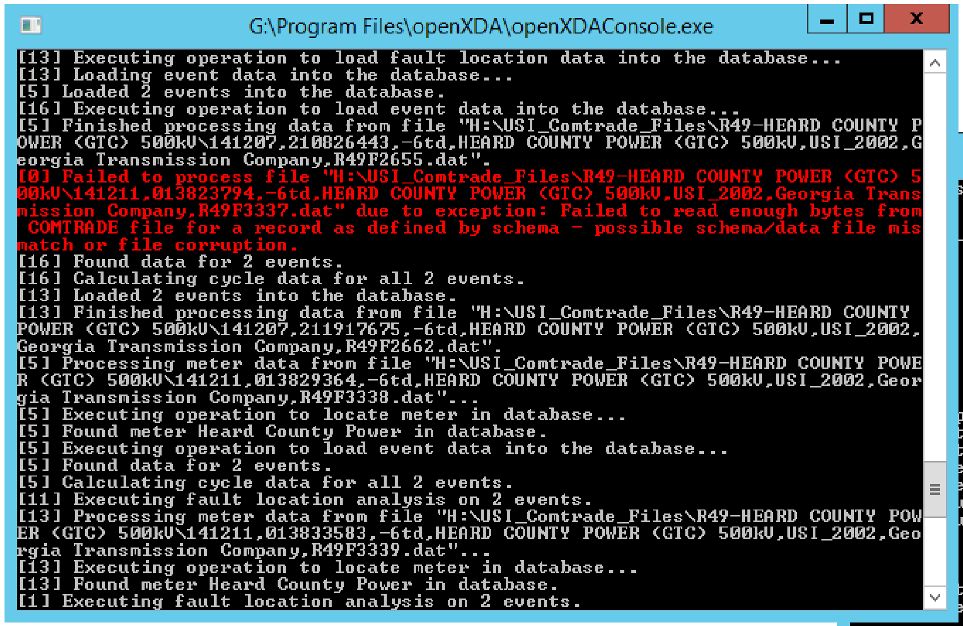
Line: [redacted] 115KV LINE (41.80 miles)

	[redacted] - R85
Fault Type:	BC
Inception Time:	08:33:32.3793750
Fault Duration:	73.542 msec (4.41 cycles)
Fault Current:	7111.4 Amps (RMS)
Prefault Current:	15.1 Amps (RMS)
Postfault Current:	13.2 Amps (RMS)
Distance Method:	Takagi
Single-ended Distance:	17.611 miles
Short file name:	R85F2697.d00
openXDA Event ID:	255014

Line Parameters:	Value:	Per Mile:
Length (Mi)	41.8	1.0
Pos-Seq Imp Z1 (Ohm) (LLL,LLLG,LL,LLG)	33.9952∠73.21° 9.82+j32.546	0.8133∠73.21° 0.2349+j0.8133
Zero-Seq Imp Z0 (Ohm)	105.0572∠71.5375° 33.27+j99.65	2.5133∠71.5375° 0.7959+j2.384
Loop Imp ZS (Ohm) (LG)	57.6767∠72.1946° 17.6367+j54.914	1.3798∠72.1946° 0.4219+j1.3137

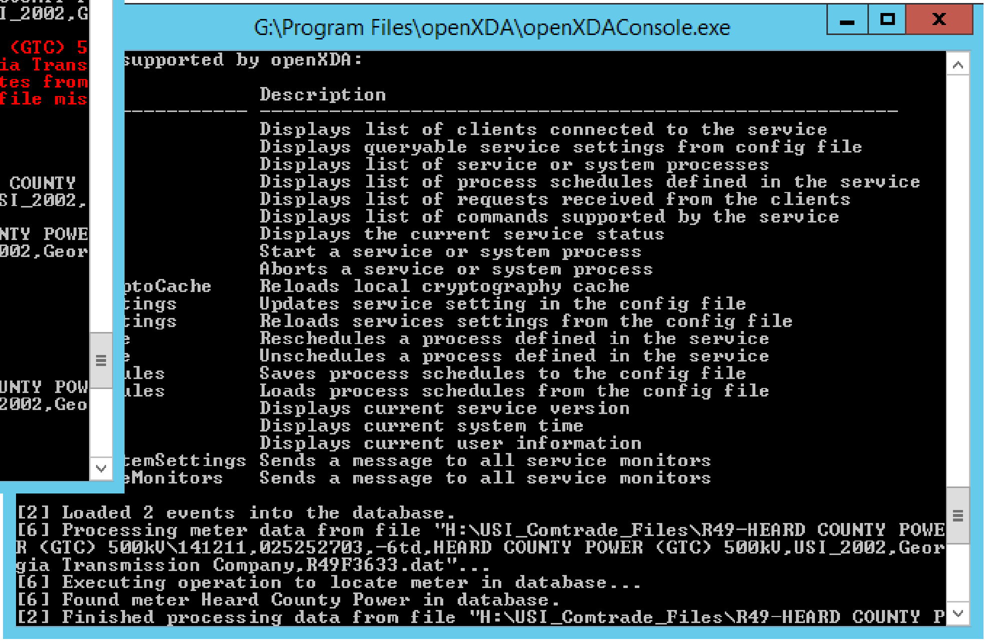
openXDA Remote Admin Console

- Real-time monitoring of status log
- Interact with service through commands



G:\Program Files\openXDA\openXDAConsole.exe

```
[13] Executing operation to load fault location data into the database...
[13] Loading event data into the database...
[5] Loaded 2 events into the database.
[16] Executing operation to load event data into the database...
[5] Finished processing data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY P
OWER (GTC) 500kV\141207,210826443,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,G
eorgia Transmission Company,R49F2655.dat".
[0] Failed to process file "H:\USI_Comtrade_Files\R49-HEARD COUNTY POWER (GTC) 5
00kV\141211,013823794,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,Georgia Trans
mission Company,R49F3337.dat" due to exception: Failed to read enough bytes from
 COMTRADE file for a record as defined by schema - possible schema/data file mis
match or file corruption.
[16] Found data for 2 events.
[16] Calculating cycle data for all 2 events.
[13] Loaded 2 events into the database.
[13] Finished processing data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY
POWER (GTC) 500kV\141207,211917675,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,
Georgia Transmission Company,R49F2662.dat".
[5] Processing meter data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY POWE
R (GTC) 500kV\141211,013829364,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,Geor
gia Transmission Company,R49F3338.dat"...
[5] Executing operation to locate meter in database...
[5] Found meter Heard County Power in database.
[5] Executing operation to load event data into the database...
[5] Found data for 2 events.
[5] Calculating cycle data for all 2 events.
[11] Executing fault location analysis on 2 events.
[13] Processing meter data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY POW
ER (GTC) 500kV\141211,013833583,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,Geo
rgia Transmission Company,R49F3339.dat"...
[13] Executing operation to locate meter in database...
[13] Found meter Heard County Power in database.
[1] Executing fault location analysis on 2 events.
```



G:\Program Files\openXDA\openXDAConsole.exe

```
supported by openXDA:

            Description
----------- ----------------------------------------------------------
            Displays list of clients connected to the service
            Displays queryable service settings from config file
            Displays list of service or system processes
            Displays list of process schedules defined in the service
            Displays list of requests received from the clients
            Displays list of commands supported by the service
            Displays the current service status
            Start a service or system process
            Aborts a service or system process
ptoCache    Reloads local cryptography cache
tings       Updates service setting in the config file
tings       Reloads services settings from the config file
e           Reschedules a process defined in the service
e           Unschedules a process defined in the service
ules        Saves process schedules to the config file
ules        Loads process schedules from the config file
            Displays current service version
            Displays current system time
            Displays current user information
temSettings Sends a message to all service monitors
eMonitors   Sends a message to all service monitors

[2] Loaded 2 events into the database.
[6] Processing meter data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY POWE
R (GTC) 500kV\141211,025252703,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,Geor
gia Transmission Company,R49F3633.dat"...
[6] Executing operation to locate meter in database...
[6] Found meter Heard County Power in database.
[2] Finished processing data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY P
```

openXDA Logging

- Text File-Based Logging
  - **Status Log** (key messages)
  - **Error Log**
    - Assembly, class, method
    - Stack trace
    - Exception type and message
  - **Debug Log** – All status, most error and copious additional messages
- DB Logging
  - For analytics, easily searchable/reportable record of files processed and errors encountered

What is the open PQ Dashboard?

- The presentation layer PQ data housed in openXDA.

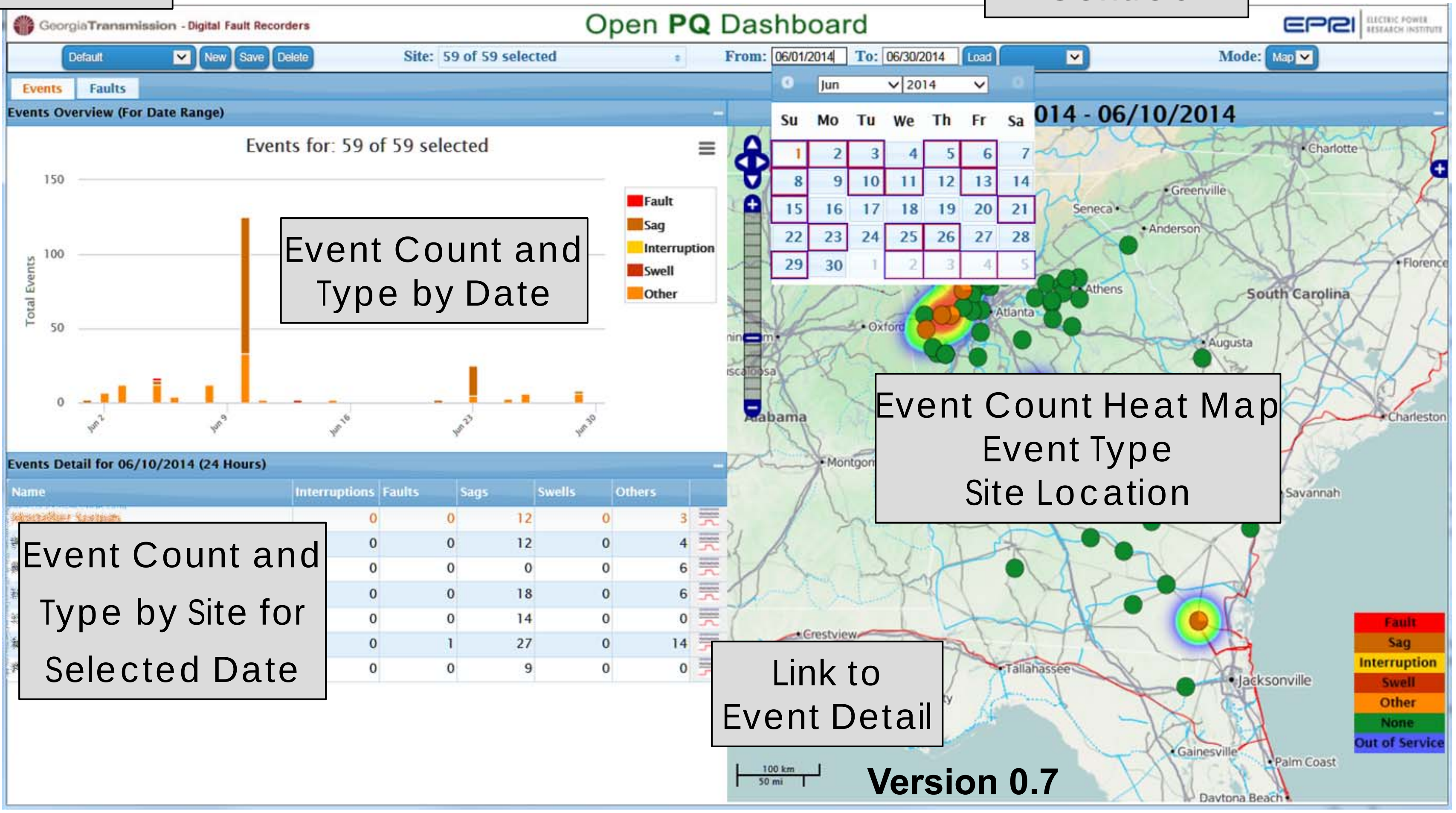
The openPQ Dashboard has been primarily funded by EPRI with extensions and contributions provided by others.

PQDashboard

Event Data Display

Selectable Views

Powerful Date Controls



GeorgiaTransmission - Digital Fault Recorders

Open PQ Dashboard

Default | New | Save | Delete

Site: 59 of 59 selected

From: 06/01/2014 To: 06/30/2014 Load

Mode: Map

Events | Faults

Events Overview (For Date Range)

Events for: 59 of 59 selected

Event Count and Type by Date

Events Detail for 06/10/2014 (24 Hours)

Name	Interruptions	Faults	Sags	Swells	Others
[illegible]	0	0	12	0	3
[illegible]	0	0	12	0	4
[illegible]	0	0	0	0	6
[illegible]	0	0	18	0	6
[illegible]	0	0	14	0	0
[illegible]	0	1	27	0	14
[illegible]	0	0	9	0	0

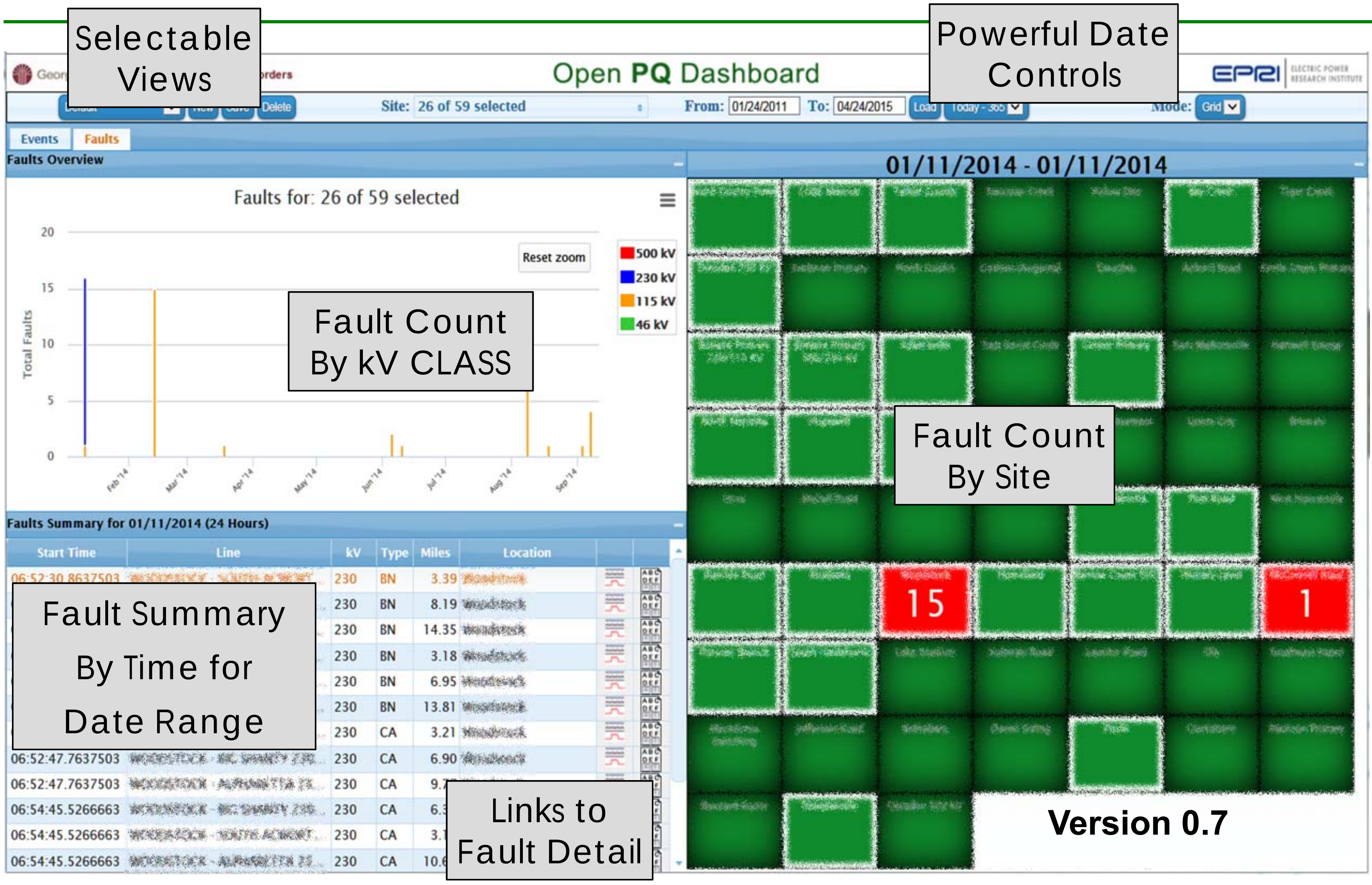
Event Count and Type by Site for Selected Date

Event Count Heat Map
Event Type
Site Location

Link to Event Detail

Version 0.7

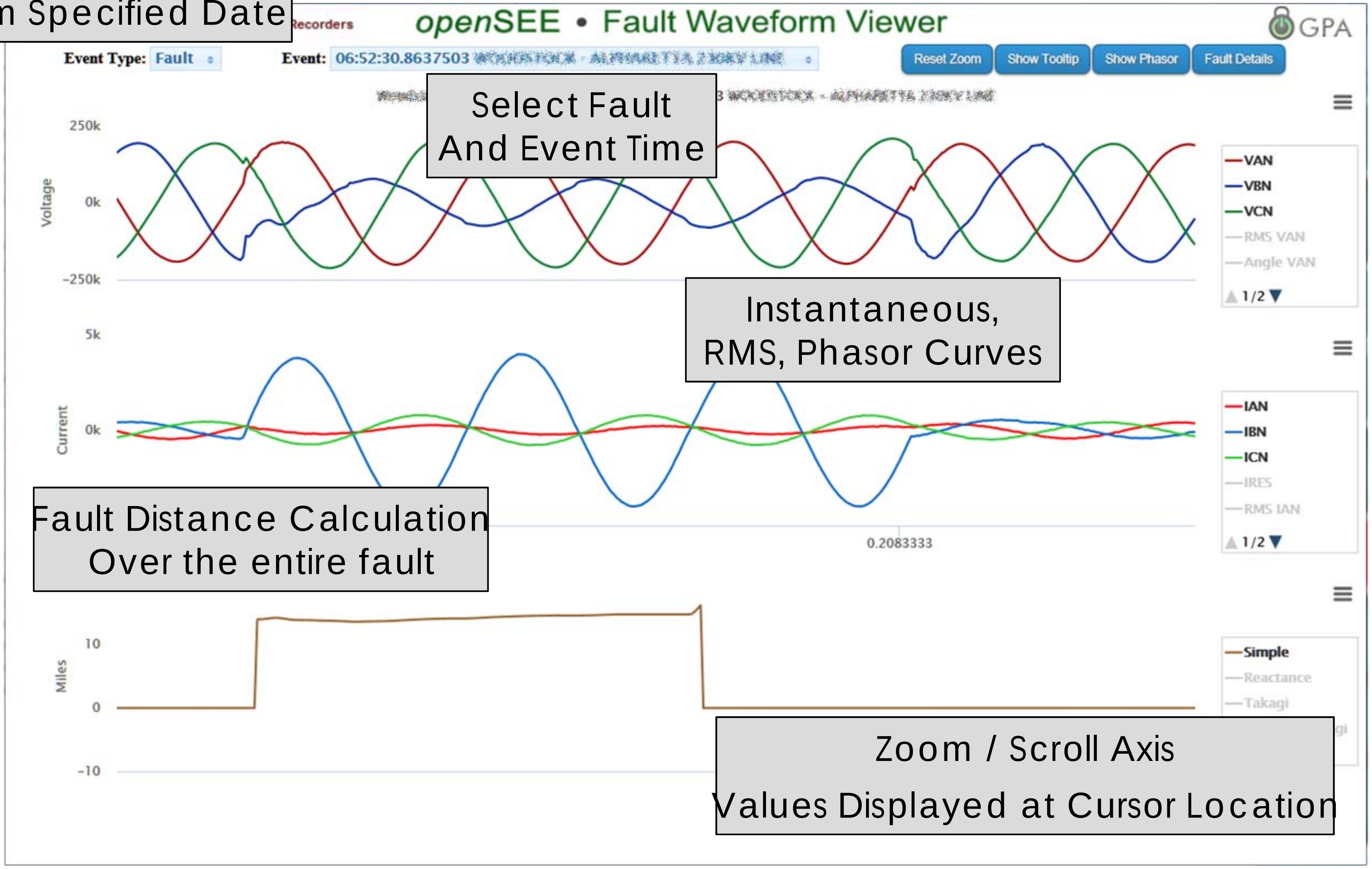
Fault Data Display



Faults Summary for 01/11/2014 (24 Hours)

Start Time	Line	kV	Type	Miles	Location
06:52:30.8637503	[redacted]	230	BN	3.39	[redacted]
		230	BN	8.19	[redacted]
		230	BN	14.35	[redacted]
		230	BN	3.18	[redacted]
		230	BN	6.95	[redacted]
		230	BN	13.81	[redacted]
		230	CA	3.21	[redacted]
06:52:47.7637503	[redacted]	230	CA	6.90	[redacted]
06:52:47.7637503	[redacted]	230	CA	9.7	
06:54:45.5266663	[redacted]	230	CA	6.3	
06:54:45.5266663	[redacted]	230	CA	3.1	
06:54:45.5266663	[redacted]	230	CA	10.6	

Fault Waveform Detail



Select Any Event From Specified Date

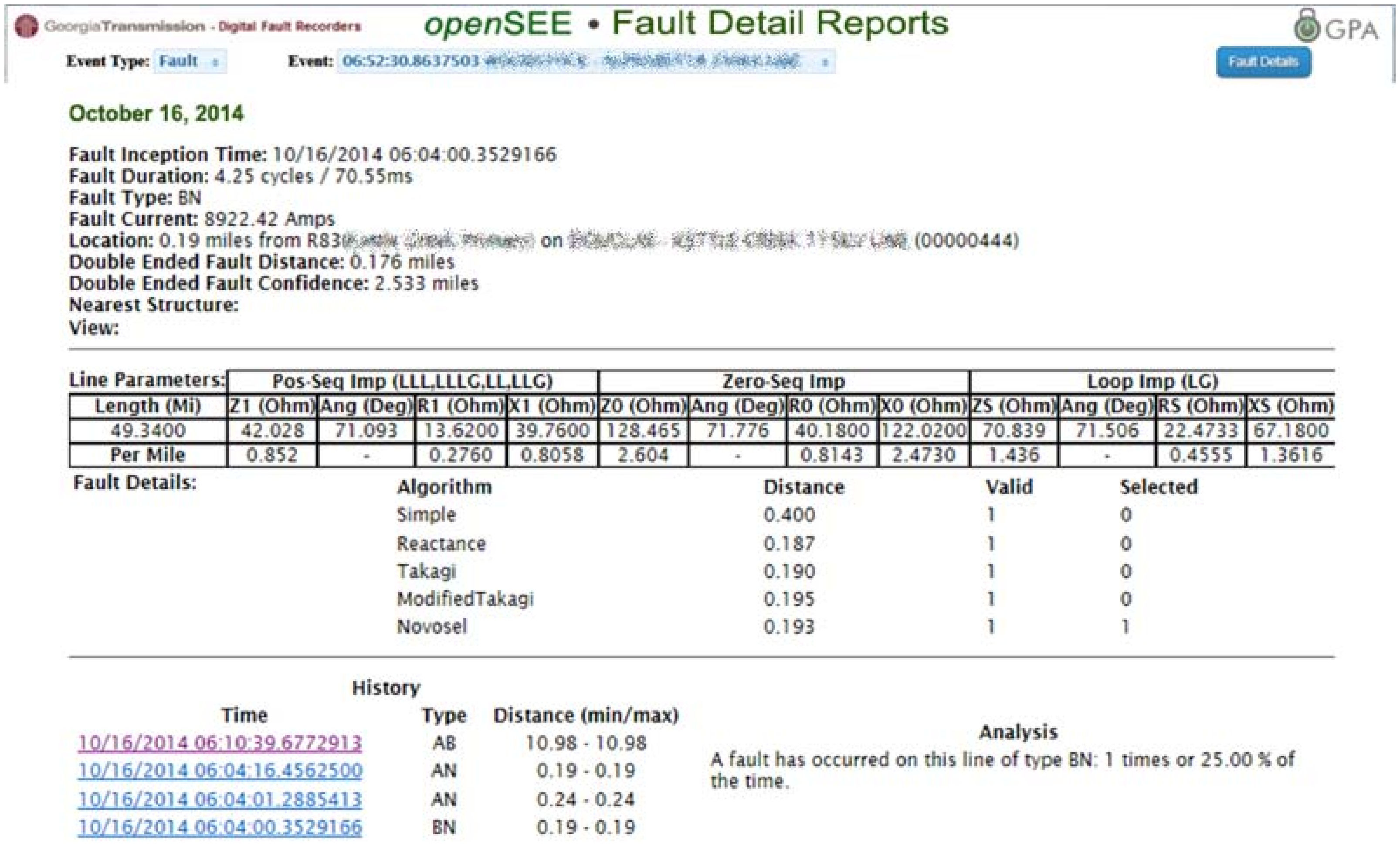
Select Fault And Event Time

Instantaneous, RMS, Phasor Curves

Fault Distance Calculation Over the entire fault

Zoom / Scroll Axis
Values Displayed at Cursor Location

Fault Detail Report



GeorgiaTransmission - Digital Fault Recorders

openSEE • Fault Detail Reports

GPA

Event Type: Fault

Event: 06:52:30.8637503 [redacted]

Fault Details

October 16, 2014

Fault Inception Time: 10/16/2014 06:04:00.3529166
Fault Duration: 4.25 cycles / 70.55ms
Fault Type: BN
Fault Current: 8922.42 Amps
Location: 0.19 miles from R83[redacted] on [redacted] (00000444)
Double Ended Fault Distance: 0.176 miles
Double Ended Fault Confidence: 2.533 miles
Nearest Structure:
View:

Line Parameters:	Pos-Seq Imp (LLL,LLLG,LL,LLG)				Zero-Seq Imp				Loop Imp (LG)			
Length (Mi)	Z1 (Ohm)	Ang (Deg)	R1 (Ohm)	X1 (Ohm)	Z0 (Ohm)	Ang (Deg)	R0 (Ohm)	X0 (Ohm)	ZS (Ohm)	Ang (Deg)	RS (Ohm)	XS (Ohm)
49.3400	42.028	71.093	13.6200	39.7600	128.465	71.776	40.1800	122.0200	70.839	71.506	22.4733	67.1800
Per Mile	0.852	-	0.2760	0.8058	2.604	-	0.8143	2.4730	1.436	-	0.4555	1.3616

Fault Details:

Algorithm	Distance	Valid	Selected
Simple	0.400	1	0
Reactance	0.187	1	0
Takagi	0.190	1	0
ModifiedTakagi	0.195	1	0
Novosel	0.193	1	1

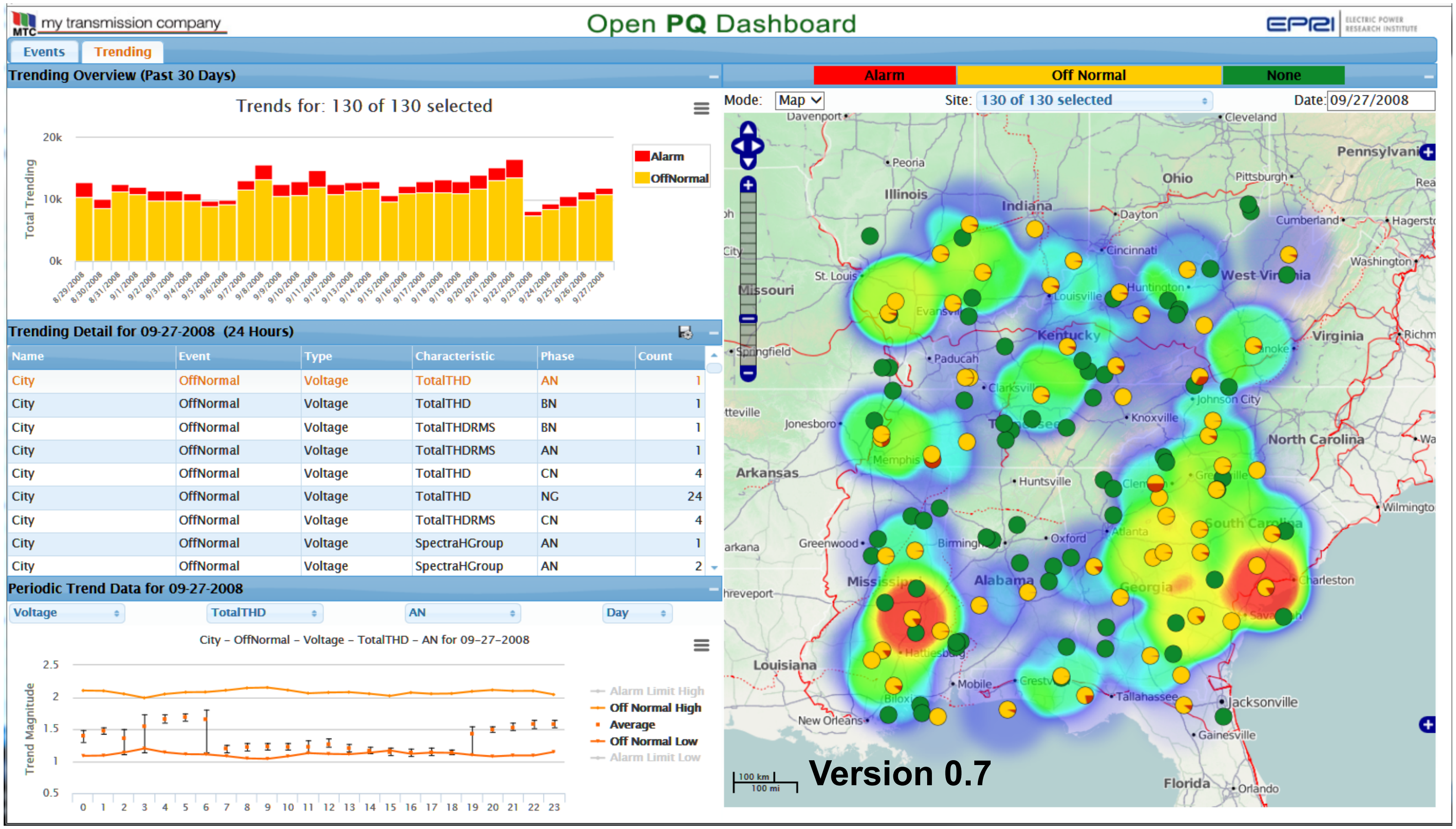
History

Time	Type	Distance (min/max)
10/16/2014 06:10:39.6772913	AB	10.98 - 10.98
10/16/2014 06:04:16.4562500	AN	0.19 - 0.19
10/16/2014 06:04:01.2885413	AN	0.24 - 0.24
10/16/2014 06:04:00.3529166	BN	0.19 - 0.19

Analysis

A fault has occurred on this line of type BN: 1 times or 25.00 % of the time.

Trending Data Display


my transmission company
MTC
Open PQ Dashboard
EPRI ELECTRIC POWER RESEARCH INSTITUTE
Events
Trending
Trending Overview (Past 30 Days)
Trends for: 130 of 130 selected
Total Trending
20k
10k
0k
Alarm
OffNormal
Alarm
Off Normal
None
Mode: Map
Site: 130 of 130 selected
Date: 09/27/2008
Trending Detail for 09-27-2008 (24 Hours)
Periodic Trend Data for 09-27-2008
Voltage
TotalTHD
AN
Day
City – OffNormal – Voltage – TotalTHD – AN for 09–27–2008
Trend Magnitude
Alarm Limit High
Off Normal High
Average
Off Normal Low
Alarm Limit Low
Version 0.7

Name	Event	Type	Characteristic	Phase	Count
City	OffNormal	Voltage	TotalTHD	AN	1
City	OffNormal	Voltage	TotalTHD	BN	1
City	OffNormal	Voltage	TotalTHDRMS	BN	1
City	OffNormal	Voltage	TotalTHDRMS	AN	1
City	OffNormal	Voltage	TotalTHD	CN	4
City	OffNormal	Voltage	TotalTHD	NG	24
City	OffNormal	Voltage	TotalTHDRMS	CN	4
City	OffNormal	Voltage	SpectraHGroup	AN	1
City	OffNormal	Voltage	SpectraHGroup	AN	2

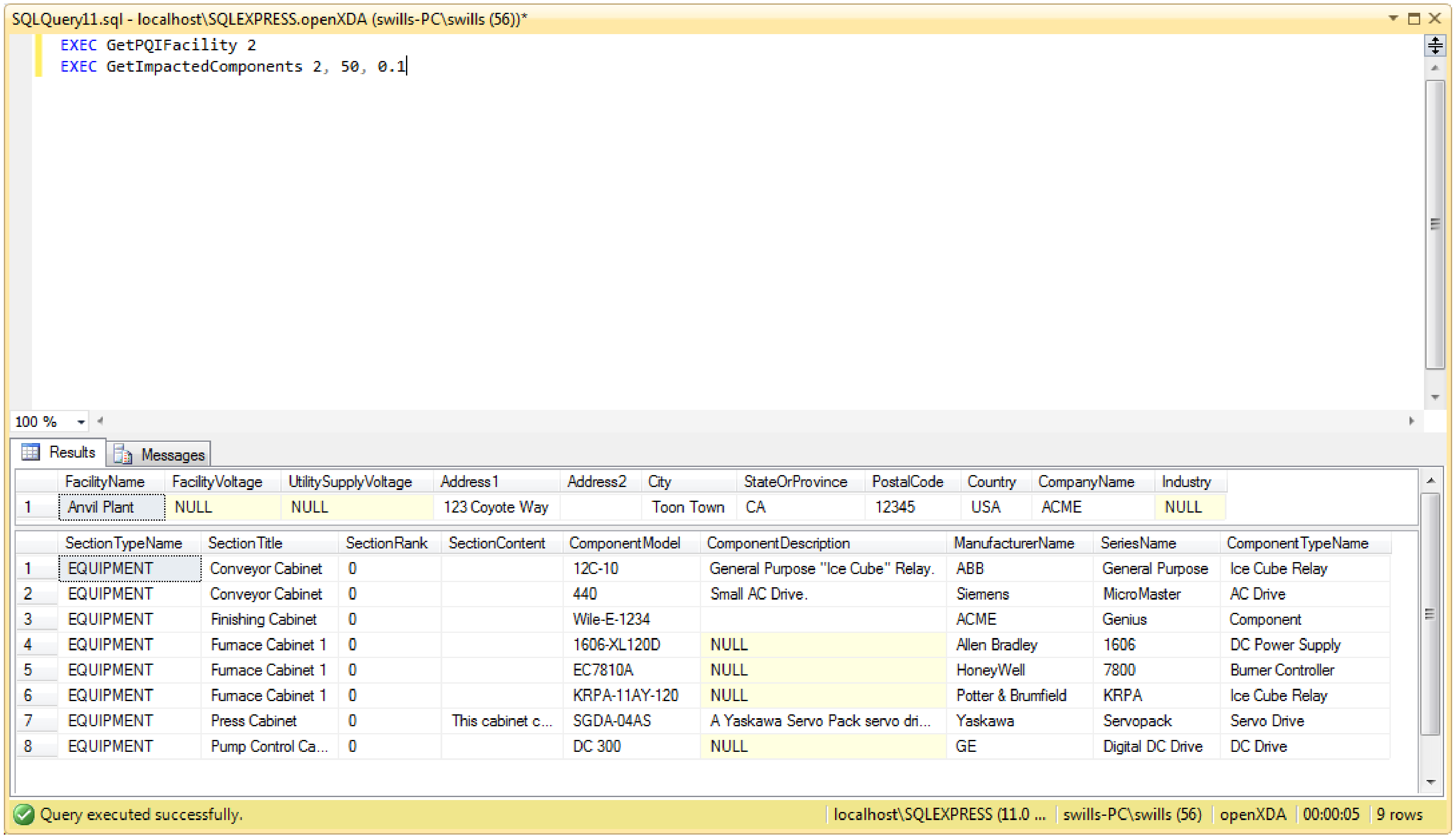
openEAS
Extended Analytic Servcies

- Provides a separate Windows service platform for analytics based on the openXDA database.
- Designed with integration in mind. Write analytics on another platform, such as Matlab, and integrate the results into openXDA.
- Distributing analytics to separate Windows services allows for sandboxing analytics to prevent potentially unsafe code from compromising the openXDA platform.

PQ Investigator Integration

- PQ Investigator tolerance curves indicate the failure points of equipment based on voltage magnitude and duration.
- Automatically determine after an event, such as a voltage sag, what equipment might have been affected by the disturbance.
- View the list of affected equipment in the PQ Dashboard.

PQ Investigator Integration



SQLQuery11.sql - localhost\SQLEXPRESS.openXDA (swills-PC\swills (56))*

```
EXEC GetPQIFacility 2
EXEC GetImpactedComponents 2, 50, 0.1
```

100 %

Results | Messages

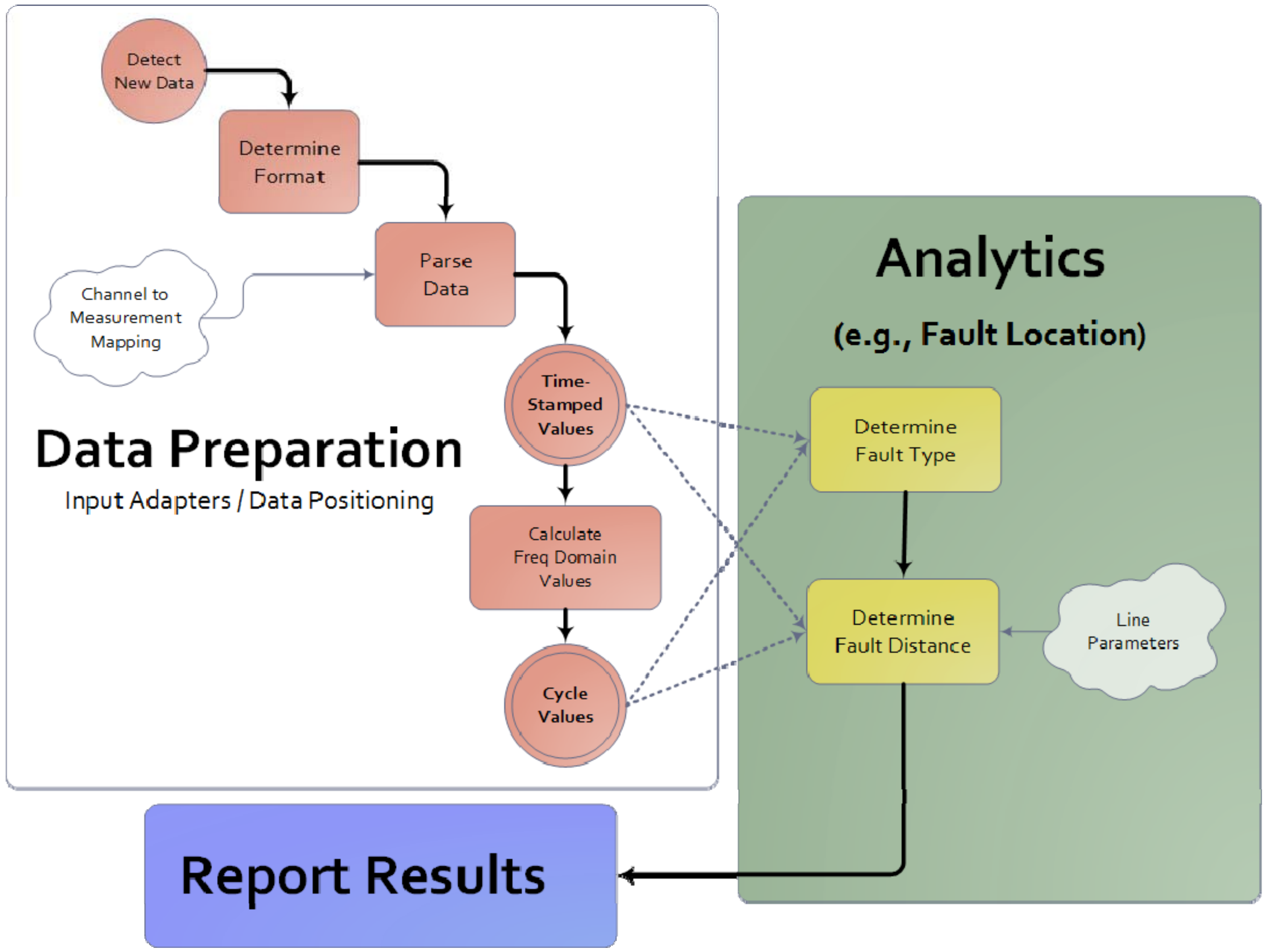
	FacilityName	FacilityVoltage	UtilitySupplyVoltage	Address1	Address2	City	StateOrProvince	PostalCode	Country	CompanyName	Industry
1	Anvil Plant	NULL	NULL	123 Coyote Way		Toon Town	CA	12345	USA	ACME	NULL

	SectionTypeName	SectionTitle	SectionRank	SectionContent	ComponentModel	ComponentDescription	ManufacturerName	SeriesName	ComponentTypeName
1	EQUIPMENT	Conveyor Cabinet	0		12C-10	General Purpose "Ice Cube" Relay.	ABB	General Purpose	Ice Cube Relay
2	EQUIPMENT	Conveyor Cabinet	0		440	Small AC Drive.	Siemens	MicroMaster	AC Drive
3	EQUIPMENT	Finishing Cabinet	0		Wile-E-1234		ACME	Genius	Component
4	EQUIPMENT	Furnace Cabinet 1	0		1606-XL120D	NULL	Allen Bradley	1606	DC Power Supply
5	EQUIPMENT	Furnace Cabinet 1	0		EC7810A	NULL	HoneyWell	7800	Burner Controller
6	EQUIPMENT	Furnace Cabinet 1	0		KRPA-11AY-120	NULL	Potter & Brumfield	KRPA	Ice Cube Relay
7	EQUIPMENT	Press Cabinet	0	This cabinet c...	SGDA-04AS	A Yaskawa Servo Pack servo dri...	Yaskawa	Servopack	Servo Drive
8	EQUIPMENT	Pump Control Ca...	0		DC 300	NULL	GE	Digital DC Drive	DC Drive

Query executed successfully. | localhost\SQLEXPRESS (11.0 ... | swills-PC\swills (56) | openXDA | 00:00:05 | 9 rows

The Deep Dive

openXDA is an Automation Platform



Highlight 1

The processes executed by openXDA are database driven.

openXDA Constructs

- ConfigurationLoader
  - Load configuration updates from configuration source
- DataReader
  - Read data from files
- DataOperation
  - Perform data analysis and load results into database
- DataWriter
  - Provide results and notifications to external systems

openXDA Inputs

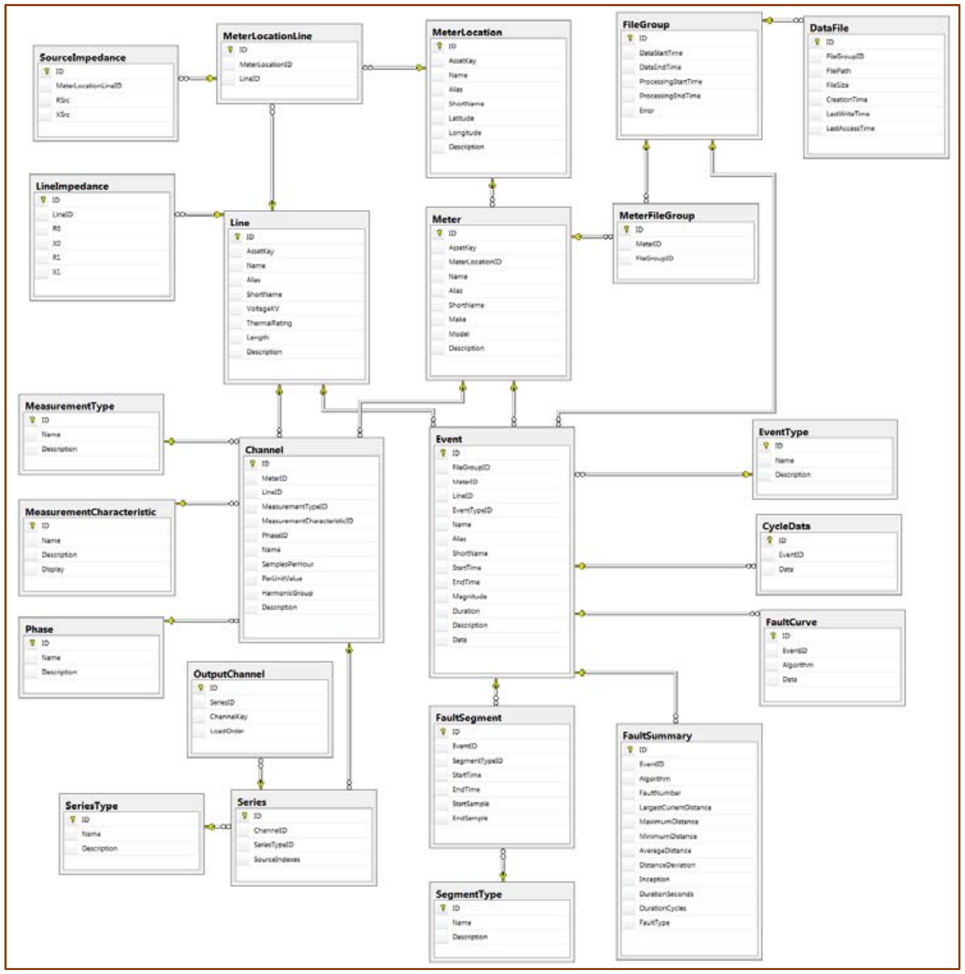
- Configuration Data
  - Meter name and location
  - Meter channel definitions
  - Line parameters
- Waveform Data
  - COMTRADE
  - PQDIF
  - EMAX (native format)
  - SEL .eve (SEL-251, SEL-351, Sel-421 relays)

openXDA Event and Soft Configuration Data

open**XDA** Data

- MS SQL server required (2012 or later)

- Easy to understand collection of tables with procedural interface layer

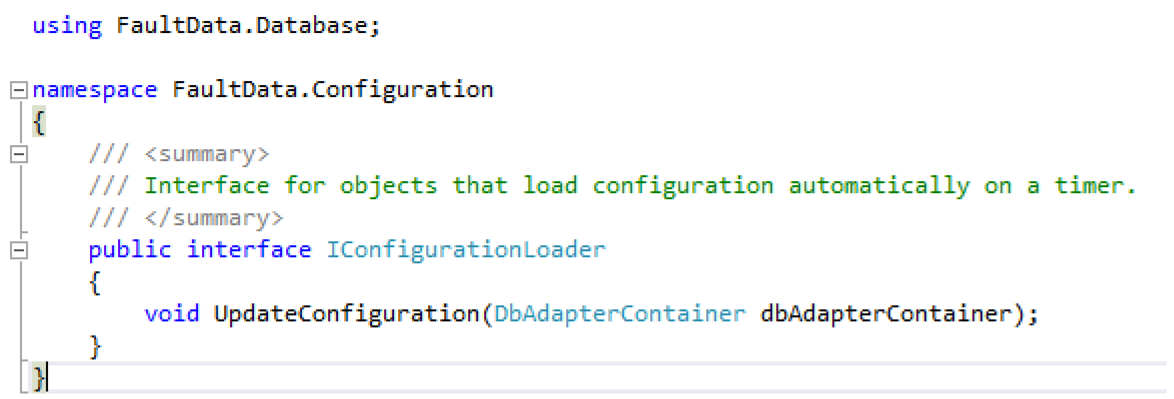


ConfigurationLoader

- The ConfigurationLoader interface allows you to…
  - Automatically load configuration from an external configuration data source on a timer
  - Manually load configuration via the remote system console

ConfigurationLoader

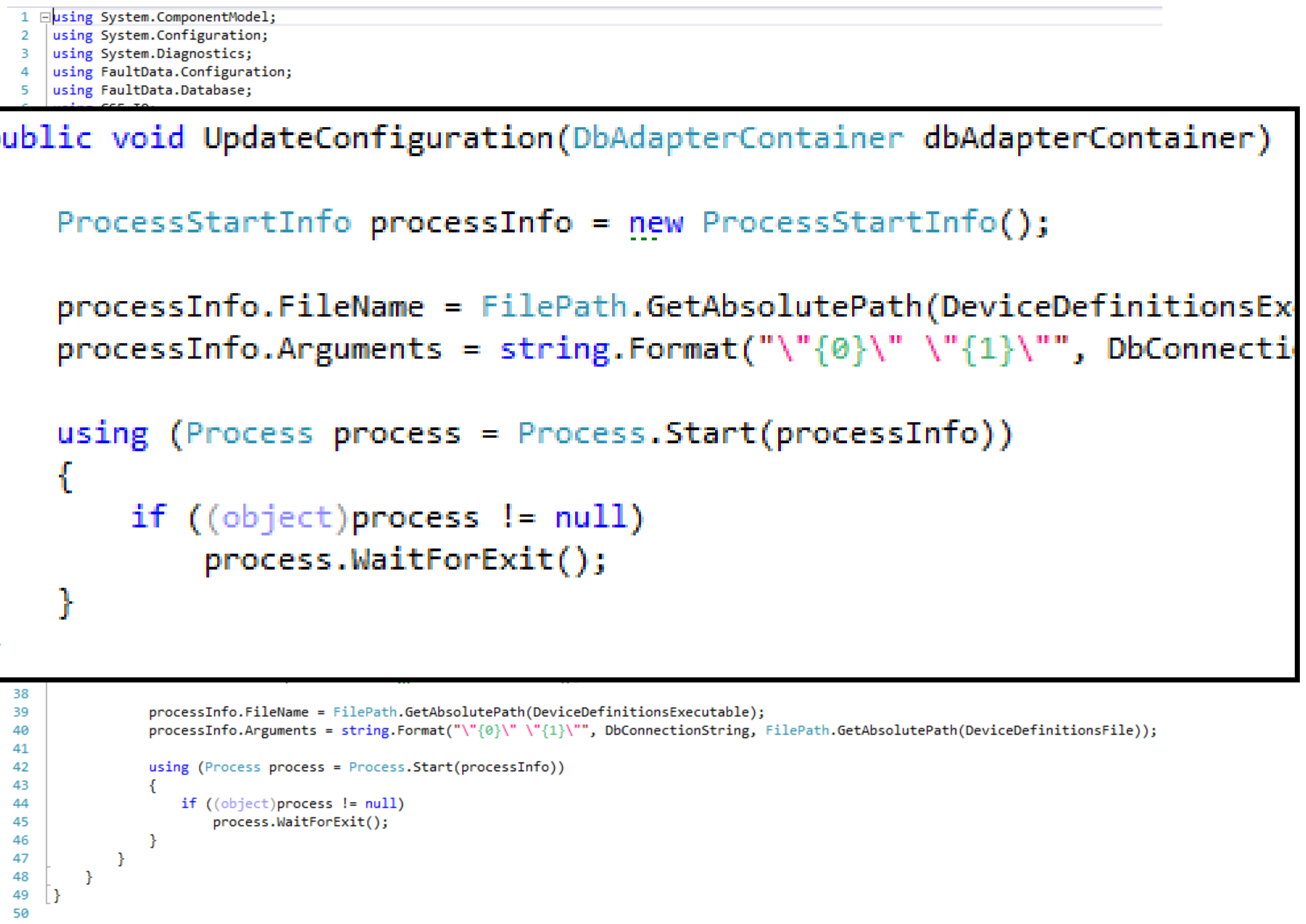
IConfigurationLoader interface



```
using FaultData.Database;

namespace FaultData.Configuration
{
    /// <summary>
    /// Interface for objects that load configuration automatically on a timer.
    /// </summary>
    public interface IConfigurationLoader
    {
        void UpdateConfiguration(DbAdapterContainer dbAdapterContainer);
    }
}
```

ConfigurationLoader

Example ConfigurationLoader

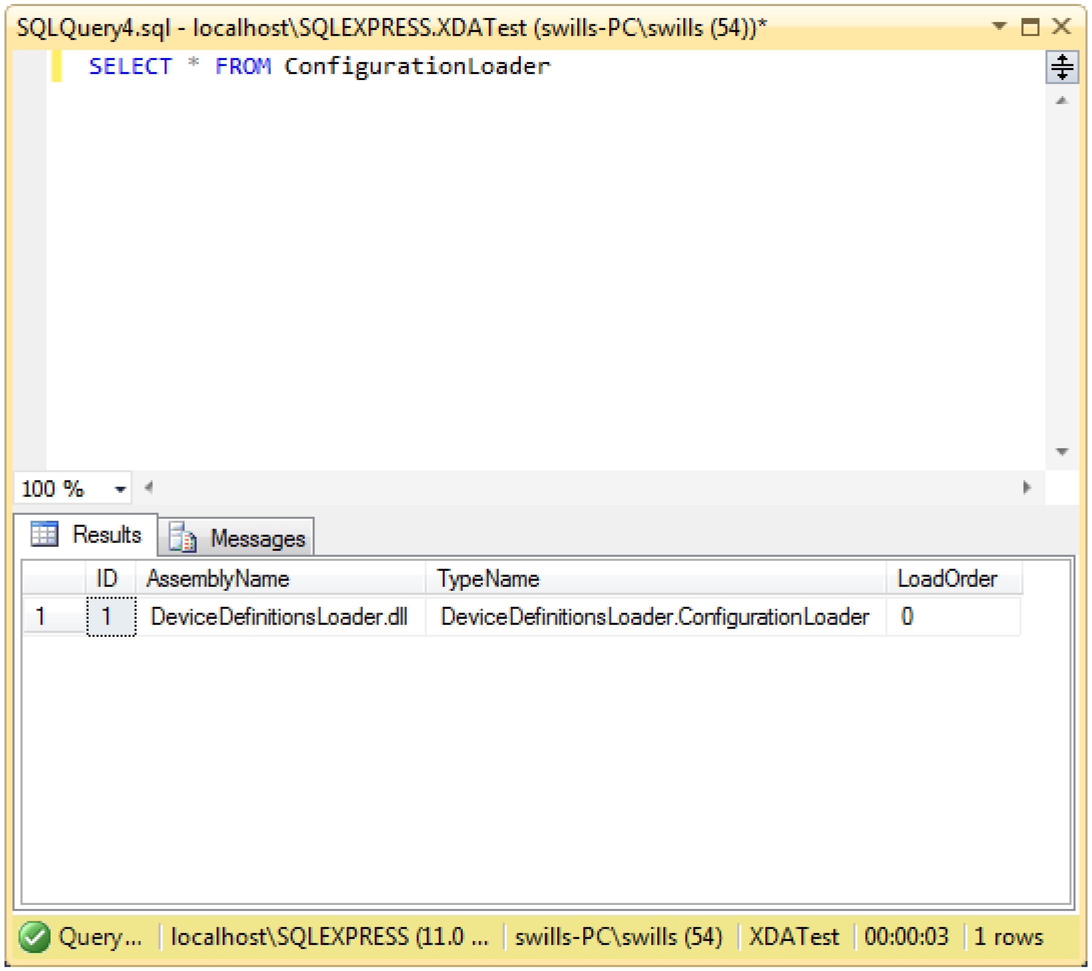


```
using System.ComponentModel;
using System.Configuration;
using System.Diagnostics;
using FaultData.Configuration;
using FaultData.Database;
```

```
public void UpdateConfiguration(DbAdapterContainer dbAdapterContainer)
{
    ProcessStartInfo processInfo = new ProcessStartInfo();

    processInfo.FileName = FilePath.GetAbsolutePath(DeviceDefinitionsEx
    processInfo.Arguments = string.Format("\"{0}\" \"{1}\"", DbConnecti

    using (Process process = Process.Start(processInfo))
    {
        if ((object)process != null)
            process.WaitForExit();
    }
}
```

```

            processInfo.FileName = FilePath.GetAbsolutePath(DeviceDefinitionsExecutable);
            processInfo.Arguments = string.Format("\"{0}\" \"{1}\"", DbConnectionString, FilePath.GetAbsolutePath(DeviceDefinitionsFile));

            using (Process process = Process.Start(processInfo))
            {
                if ((object)process != null)
                    process.WaitForExit();
            }
        }
    }
}

```

ConfigurationLoader


SQLQuery4.sql - localhost\SQLEXPRESS.XDATest (swills-PC\swills (54))*
100 %
Results
Messages

```
SELECT * FROM ConfigurationLoader
```

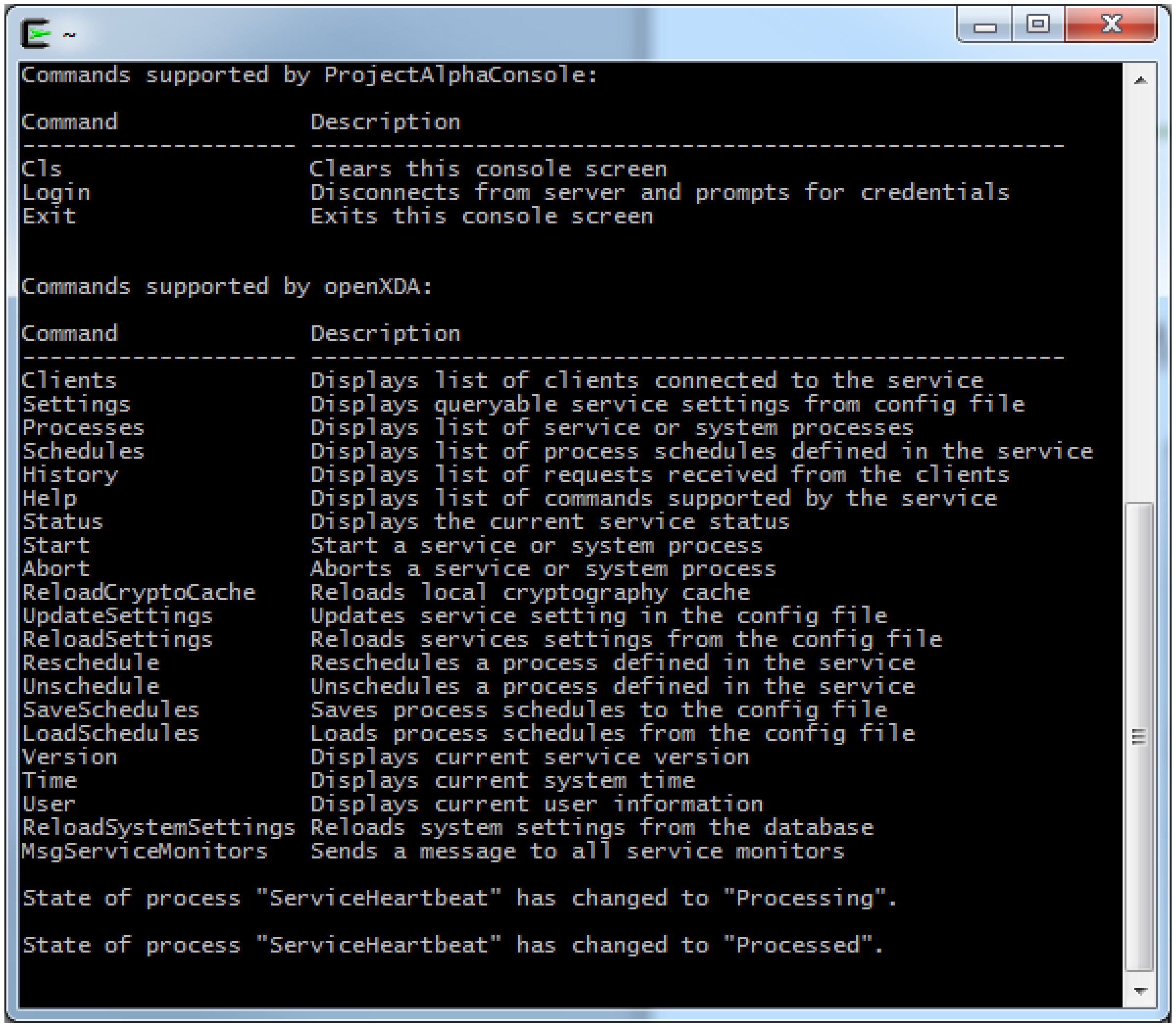
	ID	AssemblyName	TypeName	LoadOrder
1	1	DeviceDefinitionsLoader.dll	DeviceDefinitionsLoader.ConfigurationLoader	0

Query... | localhost\SQLEXPRESS (11.0 ... | swills-PC\swills (54) | XDATest | 00:00:03 | 1 rows

Highlight 2

A console application can be used to remotely monitor and control the openXDA service.

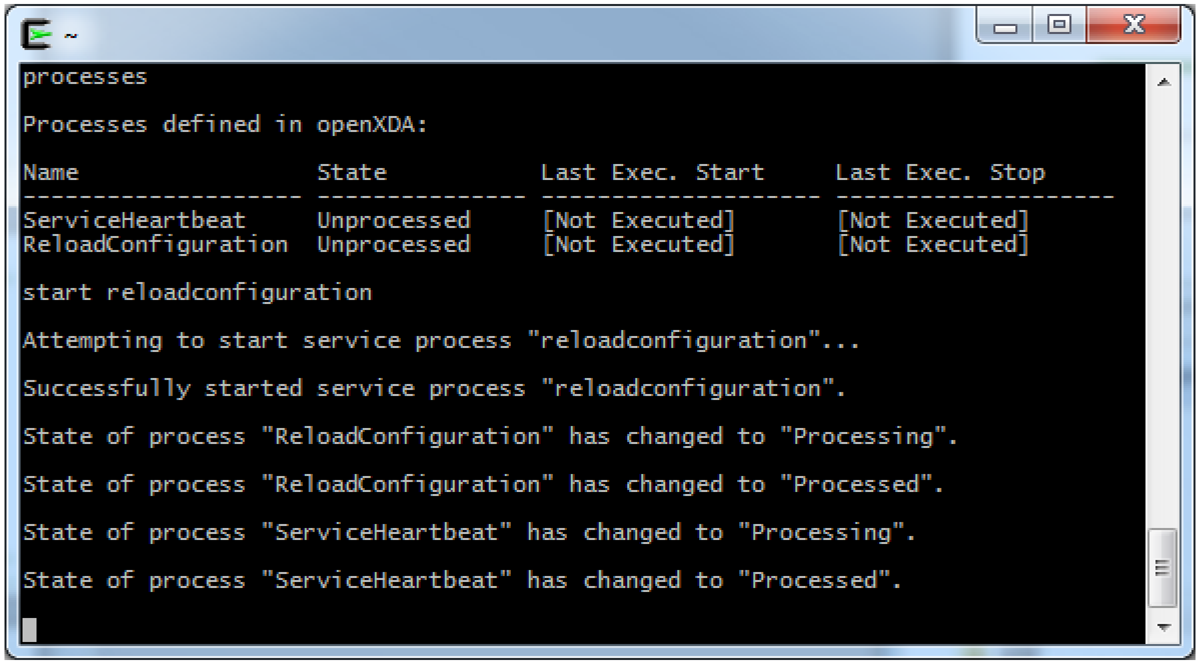
Console Commands



```
Commands supported by ProjectAlphaConsole:

Command              Description
-------------------- ------------------------------------------------------
Cls                  Clears this console screen
Login                Disconnects from server and prompts for credentials
Exit                 Exits this console screen


Commands supported by openXDA:

Command              Description
-------------------- ------------------------------------------------------
Clients              Displays list of clients connected to the service
Settings             Displays queryable service settings from config file
Processes            Displays list of service or system processes
Schedules            Displays list of process schedules defined in the service
History              Displays list of requests received from the clients
Help                 Displays list of commands supported by the service
Status               Displays the current service status
Start                Start a service or system process
Abort                Aborts a service or system process
ReloadCryptoCache    Reloads local cryptography cache
UpdateSettings       Updates service setting in the config file
ReloadSettings       Reloads services settings from the config file
Reschedule           Reschedules a process defined in the service
Unschedule           Unschedules a process defined in the service
SaveSchedules        Saves process schedules to the config file
LoadSchedules        Loads process schedules from the config file
Version              Displays current service version
Time                 Displays current system time
User                 Displays current user information
ReloadSystemSettings Reloads system settings from the database
MsgServiceMonitors   Sends a message to all service monitors

State of process "ServiceHeartbeat" has changed to "Processing".

State of process "ServiceHeartbeat" has changed to "Processed".
```

ConfigurationLoader

Manual configuration load



```
processes

Processes defined in openXDA:

Name                  State            Last Exec. Start      Last Exec. Stop
--------------------  ---------------  --------------------  --------------------
ServiceHeartbeat      Unprocessed      [Not Executed]        [Not Executed]
ReloadConfiguration   Unprocessed      [Not Executed]        [Not Executed]

start reloadconfiguration

Attempting to start service process "reloadconfiguration"...

Successfully started service process "reloadconfiguration".

State of process "ReloadConfiguration" has changed to "Processing".

State of process "ReloadConfiguration" has changed to "Processed".

State of process "ServiceHeartbeat" has changed to "Processing".

State of process "ServiceHeartbeat" has changed to "Processed".
```

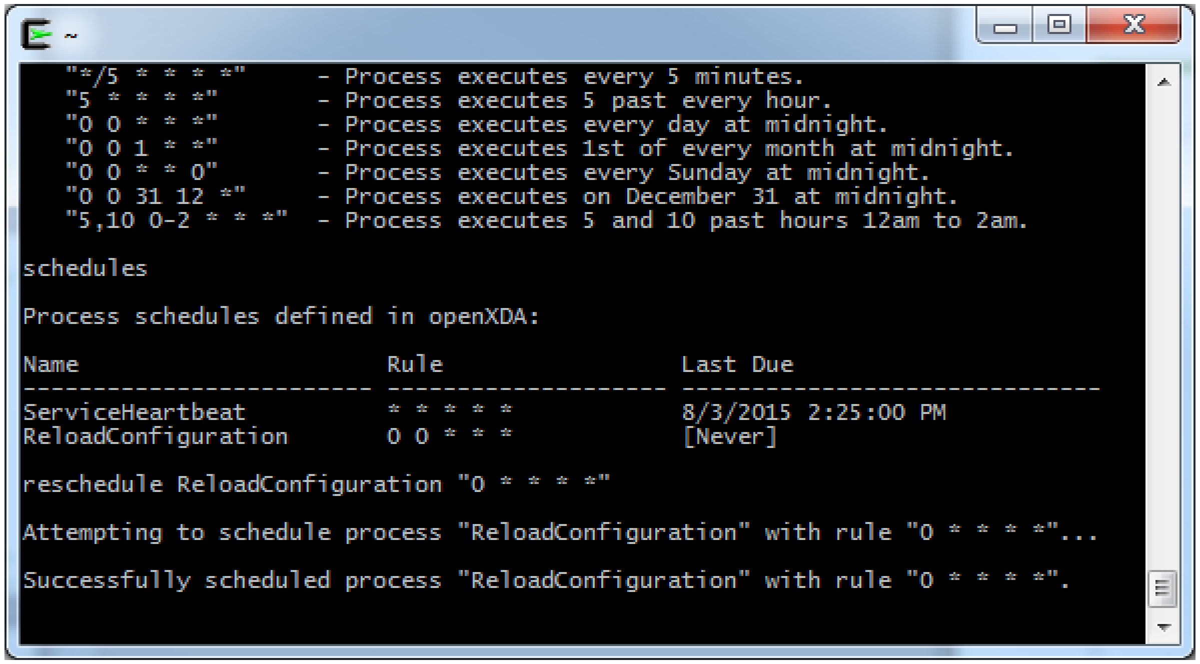
ConfigurationLoader

- By default, openXDA automatically loads configuration once per day. This behavior can be modified via the remote system console. See the following link for details about the syntax for scheduling.

https://www.gridprotectionalliance.org/NightlyBuilds/GridSolutionsFramework/Help/html/T_GSF_Scheduling_Schedule.htm

ConfigurationLoader

Reschedule automatic configuration load to once per hour instead of once per day.



```
   "*/5 * * * *"      - Process executes every 5 minutes.
   "5 * * * *"        - Process executes 5 past every hour.
   "0 0 * * *"        - Process executes every day at midnight.
   "0 0 1 * *"        - Process executes 1st of every month at midnight.
   "0 0 * * 0"        - Process executes every Sunday at midnight.
   "0 0 31 12 *"      - Process executes on December 31 at midnight.
   "5,10 0-2 * * *"   - Process executes 5 and 10 past hours 12am to 2am.

schedules

Process schedules defined in openXDA:

Name                       Rule                 Last Due
-------------------------- -------------------- ------------------------------
ServiceHeartbeat           * * * * *            8/3/2015 2:25:00 PM
ReloadConfiguration        0 0 * * *            [Never]

reschedule ReloadConfiguration "0 * * * *"

Attempting to schedule process "ReloadConfiguration" with rule "0 * * * *"...

Successfully scheduled process "ReloadConfiguration" with rule "0 * * * *".
```

GRID PROTECTION ALLIANCE

GPA User's Forum 2015
Atlanta, Georgia

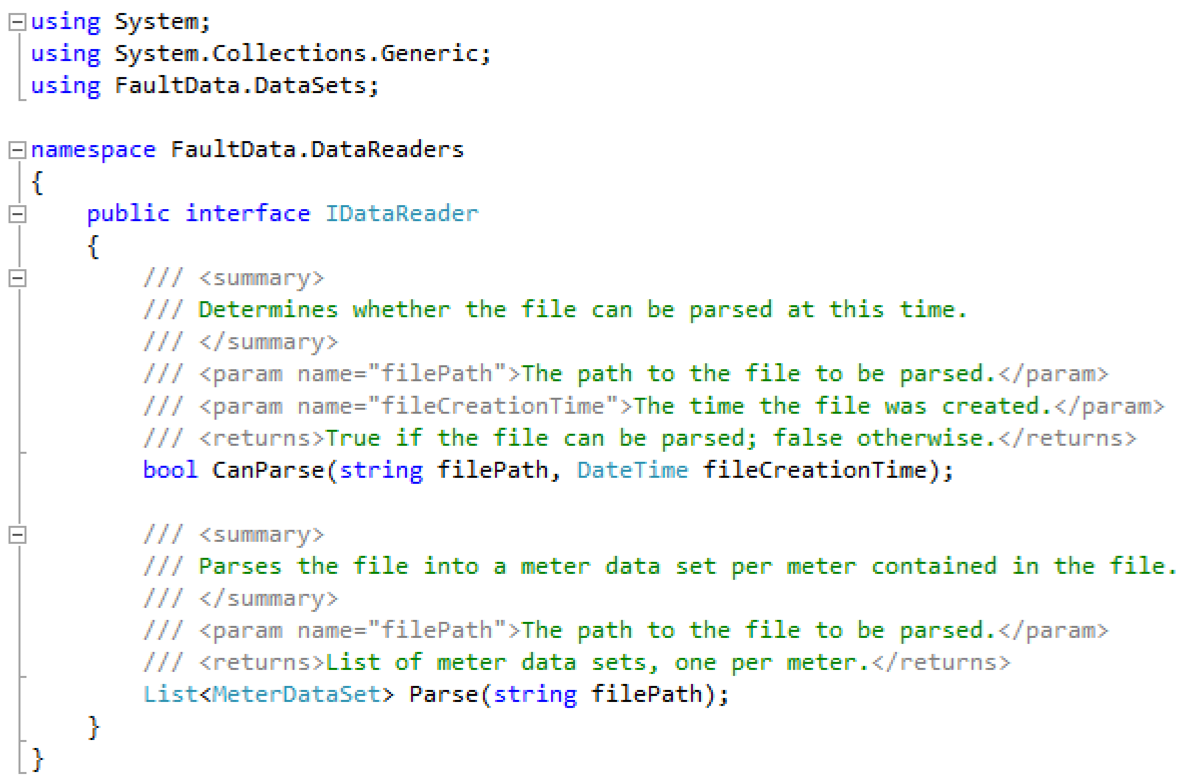
openXDA
EXTENSIBLE DISTURBANCE ANALYTICS

DataReader

- The DataReader allows you to transform data from a file into a MeterDataSet that can be used by openXDA analytics.

DataReader

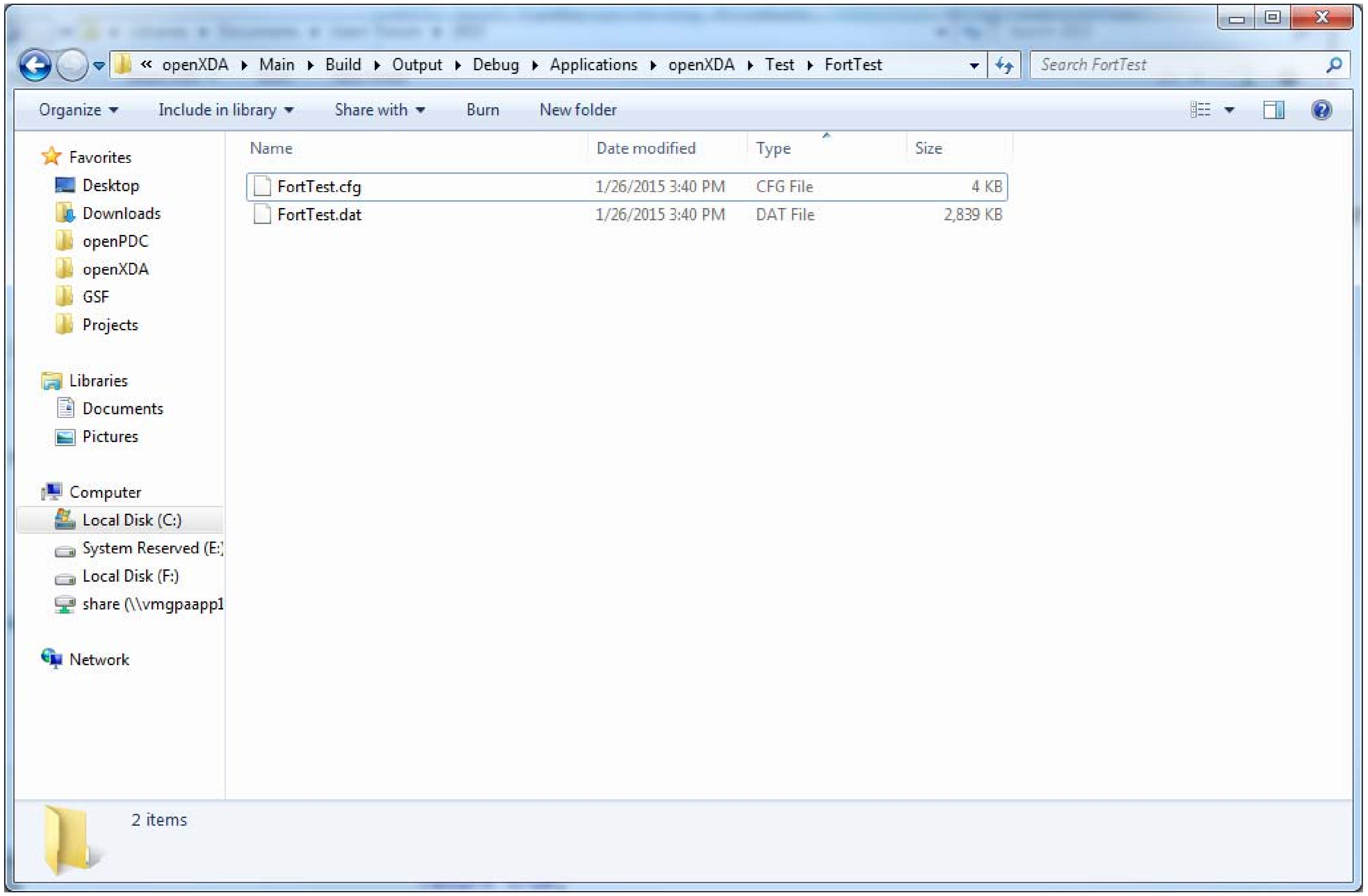
IDataReader interface



```
using System;
using System.Collections.Generic;
using FaultData.DataSets;

namespace FaultData.DataReaders
{
    public interface IDataReader
    {
        /// <summary>
        /// Determines whether the file can be parsed at this time.
        /// </summary>
        /// <param name="filePath">The path to the file to be parsed.</param>
        /// <param name="fileCreationTime">The time the file was created.</param>
        /// <returns>True if the file can be parsed; false otherwise.</returns>
        bool CanParse(string filePath, DateTime fileCreationTime);

        /// <summary>
        /// Parses the file into a meter data set per meter contained in the file.
        /// </summary>
        /// <param name="filePath">The path to the file to be parsed.</param>
        /// <returns>List of meter data sets, one per meter.</returns>
        List<MeterDataSet> Parse(string filePath);
    }
}
```

DataReader

Example COMTRADE file



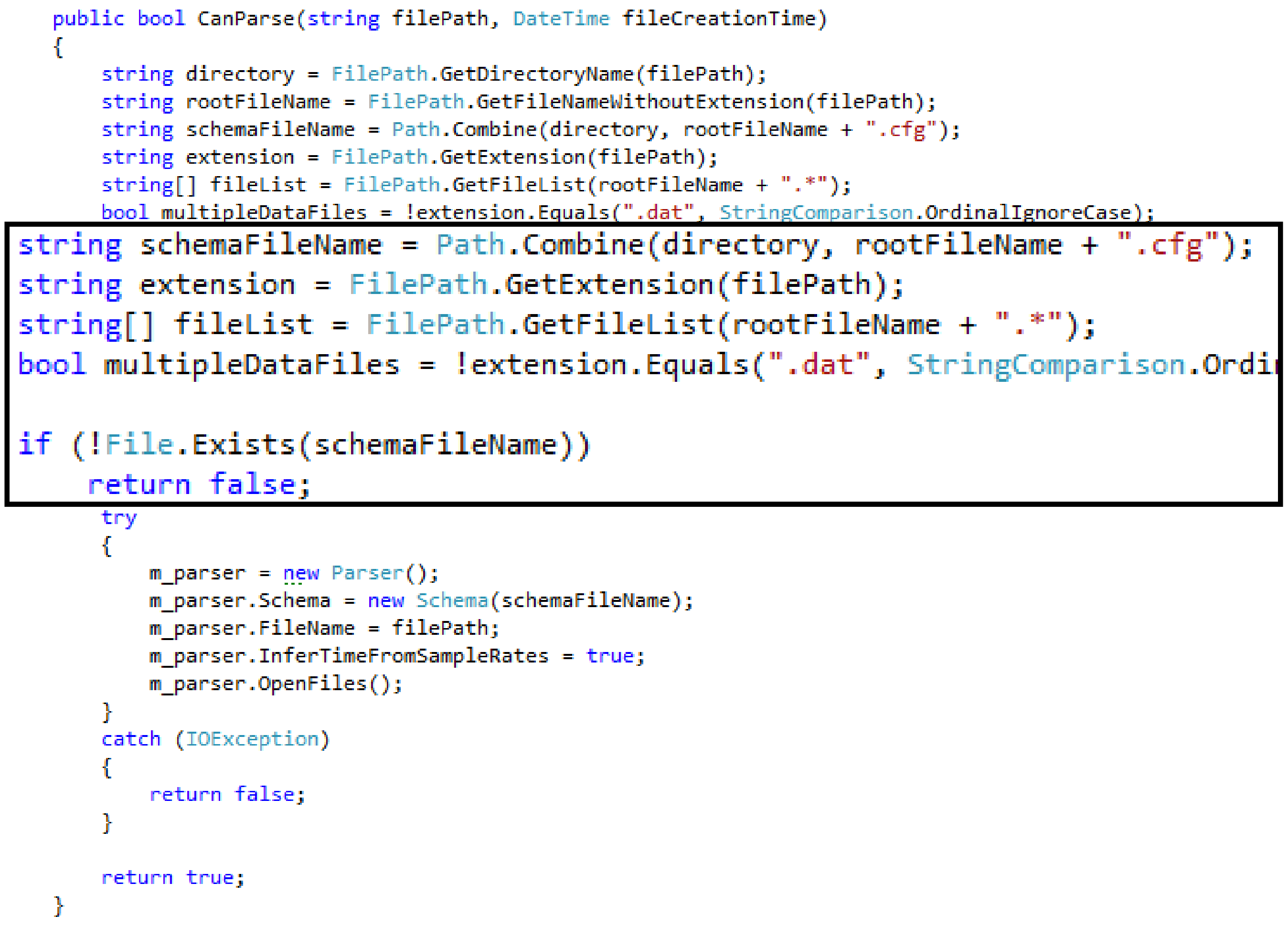
« openXDA ▸ Main ▸ Build ▸ Output ▸ Debug ▸ Applications ▸ openXDA ▸ Test ▸ FortTest

Name	Date modified	Type	Size
FortTest.cfg	1/26/2015 3:40 PM	CFG File	4 KB
FortTest.dat	1/26/2015 3:40 PM	DAT File	2,839 KB

2 items

DataReader

CanParse() method (COMTRADEReader)



```
public bool CanParse(string filePath, DateTime fileCreationTime)
{
    string directory = FilePath.GetDirectoryName(filePath);
    string rootFileName = FilePath.GetFileNameWithoutExtension(filePath);
    string schemaFileName = Path.Combine(directory, rootFileName + ".cfg");
    string extension = FilePath.GetExtension(filePath);
    string[] fileList = FilePath.GetFileList(rootFileName + ".*");
    bool multipleDataFiles = !extension.Equals(".dat", StringComparison.OrdinalIgnoreCase);
```

```
string schemaFileName = Path.Combine(directory, rootFileName + ".cfg");
string extension = FilePath.GetExtension(filePath);
string[] fileList = FilePath.GetFileList(rootFileName + ".*");
bool multipleDataFiles = !extension.Equals(".dat", StringComparison.Ordi
```

```
if (!File.Exists(schemaFileName))
    return false;
```

```
    try
    {
        m_parser = new Parser();
        m_parser.Schema = new Schema(schemaFileName);
        m_parser.FileName = filePath;
        m_parser.InferTimeFromSampleRates = true;
        m_parser.OpenFiles();
    }
    catch (IOException)
    {
        return false;
    }

    return true;
}
```

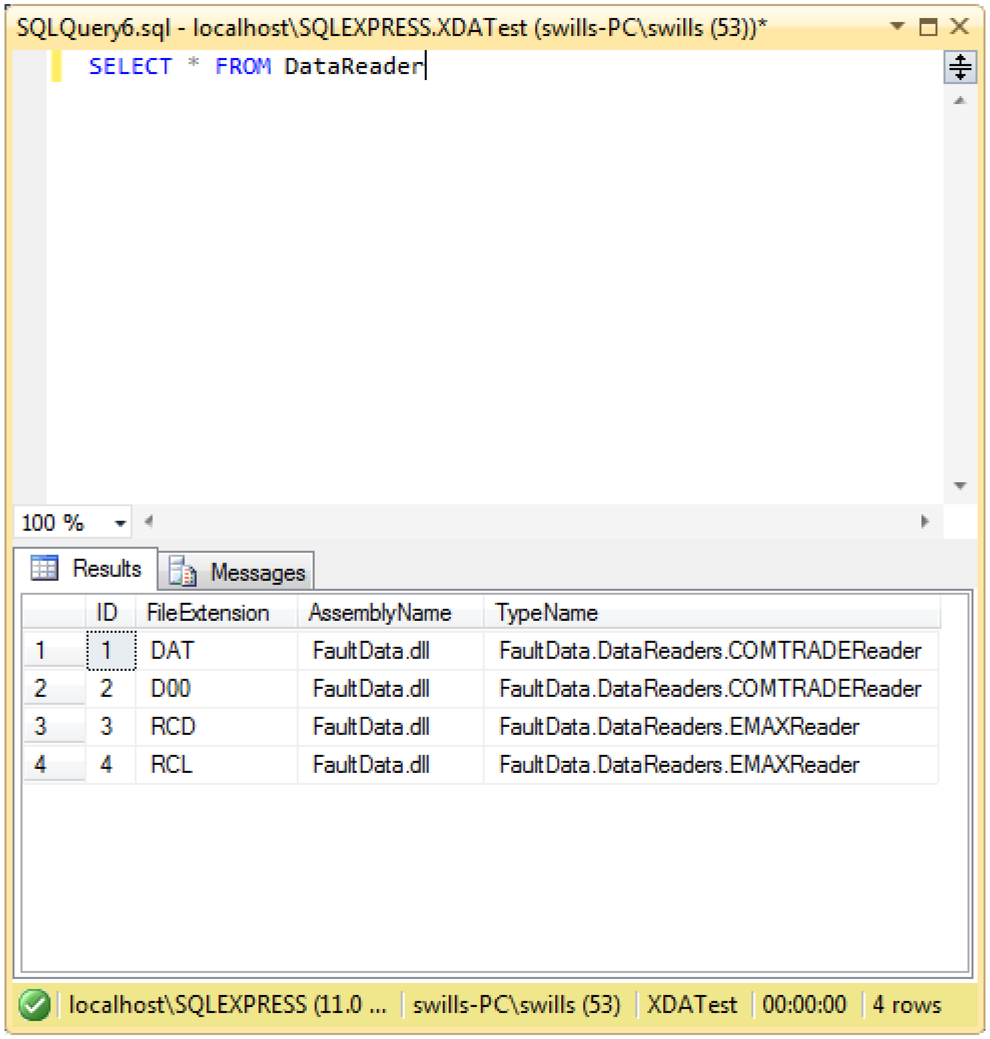
DataReader

Parse() method (COMTRADEReader)

```
public List<MeterDataSet> Parse(string filePath)
{
    MeterDataSet meterDataSet;
    Schema schema;
    Channel channel;
    DataSeries series;

    meterDataSet = new MeterDataSet();
    schema = m_parser.Schema;

    meterDataSet.Meter = new Meter();
    meterDataSet.Meter.AssetKey = schema.DeviceID;
    meterDataSet.Meter.Name = schema.DeviceID;
    meterDataSet.Meter.ShortName = schema.DeviceID.Substring(0, Math.Min(schema.DeviceID.Length, 50));

    meterDataSet.Meter.MeterLocation = new MeterLocation();
    meterDataSet.Meter.MeterLocation.AssetKey = schema.StationName;
    meterDataSet.Meter.MeterLocation.Name = schema.StationName;
    meterDataSet.Meter.MeterLocation.ShortName = schema.StationName.Substring(0, Math.Min(schema.StationName.Length, 50));
    meterDataSet.Meter.MeterLocation.Description = schema.StationName;

    foreach (AnalogChannel analogChannel in schema.AnalogChannels)
    {
        channel = ParseSeries(analogChannel);

        series = new DataSeries();
        series.SeriesInfo = channel.Series[0];

        meterDataSet.Meter.Channels.Add(channel);

        while (meterDataSet.DataSeries.Count <= analogChannel.Index)
            meterDataSet.DataSeries.Add(new DataSeries());

        meterDataSet.DataSeries[analogChannel.Index] = series;
    }
```

snip...

DataReader


SQLQuery6.sql - localhost\SQLEXPRESS.XDATest (swills-PC\swills (53))*

```
SELECT * FROM DataReader
```

	ID	FileExtension	AssemblyName	TypeName
1	1	DAT	FaultData.dll	FaultData.DataReaders.COMTRADEReader
2	2	D00	FaultData.dll	FaultData.DataReaders.COMTRADEReader
3	3	RCD	FaultData.dll	FaultData.DataReaders.EMAXReader
4	4	RCL	FaultData.dll	FaultData.DataReaders.EMAXReader

localhost\SQLEXPRESS (11.0 ... | swills-PC\swills (53) | XDATest | 00:00:00 | 4 rows

Highlight 3

Robust file watcher detects new files dropped by source devices as soon as they are available.

DataReader

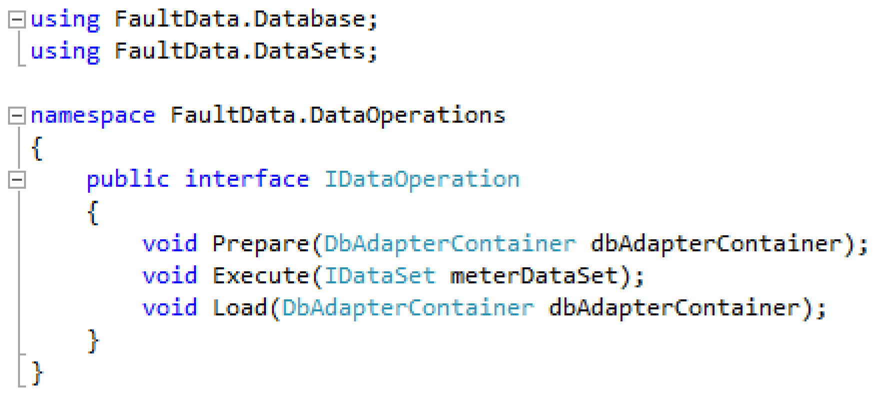
- FileExtension
    - DataReaders are created and invoked when the file watcher detects that a file has appeared on the file system. The type of the DataReader that is created is determined by the extension of the file that is detected by the file watcher.

DataOperation

- The DataOperation allows you to…
  - Analyze data from a MeterDataSet to produce meaningful results.
  - Load the results of analysis into the database.

DataOperation

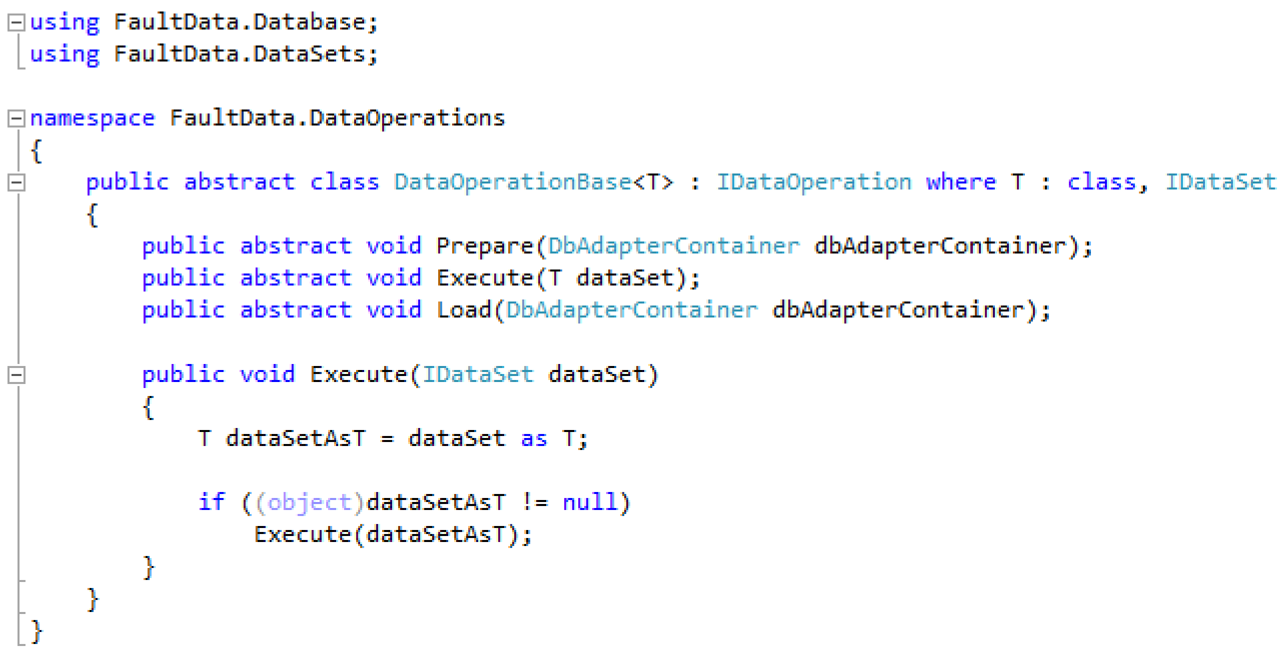
IDataOperation interface



```
using FaultData.Database;
using FaultData.DataSets;

namespace FaultData.DataOperations
{
    public interface IDataOperation
    {
        void Prepare(DbAdapterContainer dbAdapterContainer);
        void Execute(IDataSet meterDataSet);
        void Load(DbAdapterContainer dbAdapterContainer);
    }
}
```

DataOperation

DataOperationBase



```
using FaultData.Database;
using FaultData.DataSets;

namespace FaultData.DataOperations
{
    public abstract class DataOperationBase<T> : IDataOperation where T : class, IDataSet
    {
        public abstract void Prepare(DbAdapterContainer dbAdapterContainer);
        public abstract void Execute(T dataSet);
        public abstract void Load(DbAdapterContainer dbAdapterContainer);

        public void Execute(IDataSet dataSet)
        {
            T dataSetAsT = dataSet as T;

            if ((object)dataSetAsT != null)
                Execute(dataSetAsT);
        }
    }
}
```

DataOperation

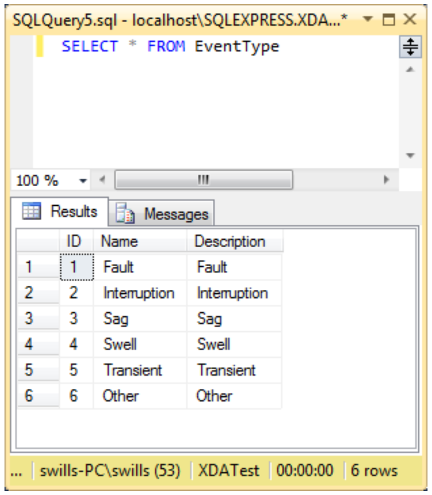
- Prepare
  - Database work that needs to be done in preparation for the data operation to execute. An example would be validation of supporting database tables, such as the EventType table.
  - Executed outside of any database transactions so that locking of database objects can be kept to a minimum. Excessive database locking can reduce performance and increase the risk of deadlocks.

DataOperation

EventOperation prepares by loading event types into the EventType table, ensuring that all the necessary event types exist before executing the operation.

```
public override void Prepare(DbAdapterContainer dbAdapterContainer)
{
    m_dbAdapterContainer = dbAdapterContainer;
    LoadEventTypes(dbAdapterContainer);
}
```

EventType table



```
SELECT * FROM EventType
```

	ID	Name	Description
1	1	Fault	Fault
2	2	Interruption	Interruption
3	3	Sag	Sag
4	4	Swell	Swell
5	5	Transient	Transient
6	6	Other	Other

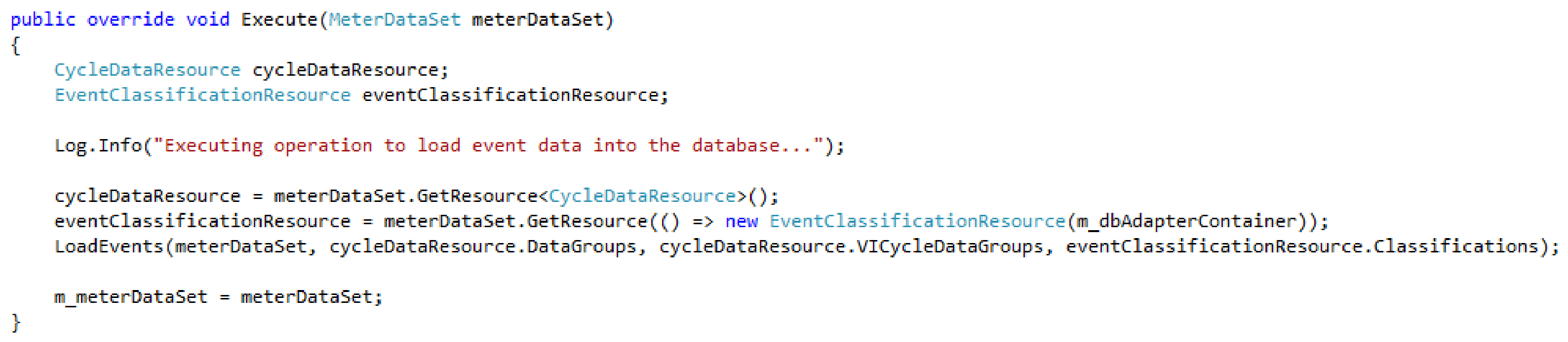
... swills-PC\swills (53) | XDATest | 00:00:00 | 6 rows

DataOperation

- Execute
  - This is where data analysis is performed, transforming the input data into results that can be loaded into the database.
  - No database work is actually done here. Analysis is performed and results are stored in memory in preparation for the Load method.

DataOperation

In the EventOperation, Events are loaded into m_eventTable, an in-memory data table that models the Event table in the database.



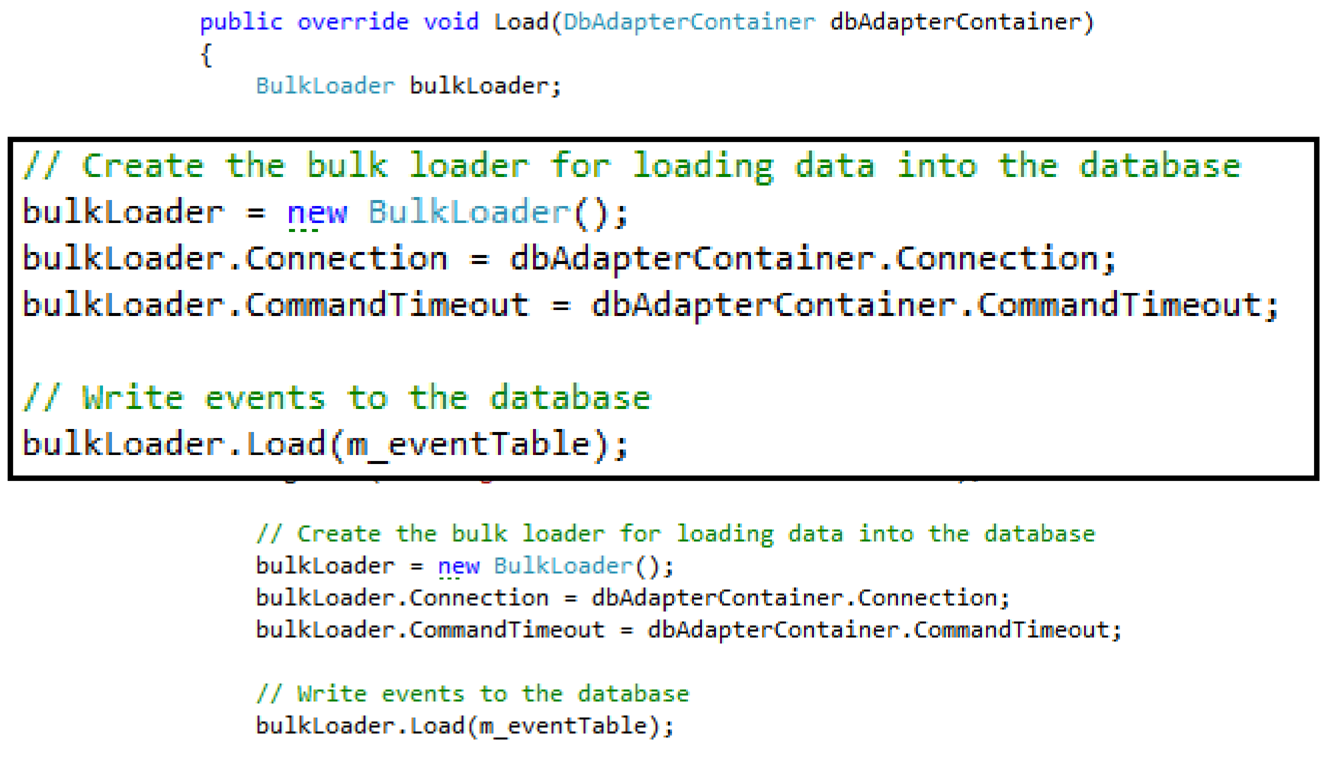
```
public override void Execute(MeterDataSet meterDataSet)
{
    CycleDataResource cycleDataResource;
    EventClassificationResource eventClassificationResource;

    Log.Info("Executing operation to load event data into the database...");

    cycleDataResource = meterDataSet.GetResource<CycleDataResource>();
    eventClassificationResource = meterDataSet.GetResource(() => new EventClassificationResource(m_dbAdapterContainer));
    LoadEvents(meterDataSet, cycleDataResource.DataGroups, cycleDataResource.VICycleDataGroups, eventClassificationResource.Classifications);

    m_meterDataSet = meterDataSet;
}
```

DataOperation

- Load
  - Loads the results of analysis into the database.
  - This method is executed in a transaction that may or may not span multiple separate DataOperations. Keep work here to a minimum in order to reduce database locking.

DataOperation

EventOperation uses a BulkLoader object to load Event records from m_eventTable into the database.



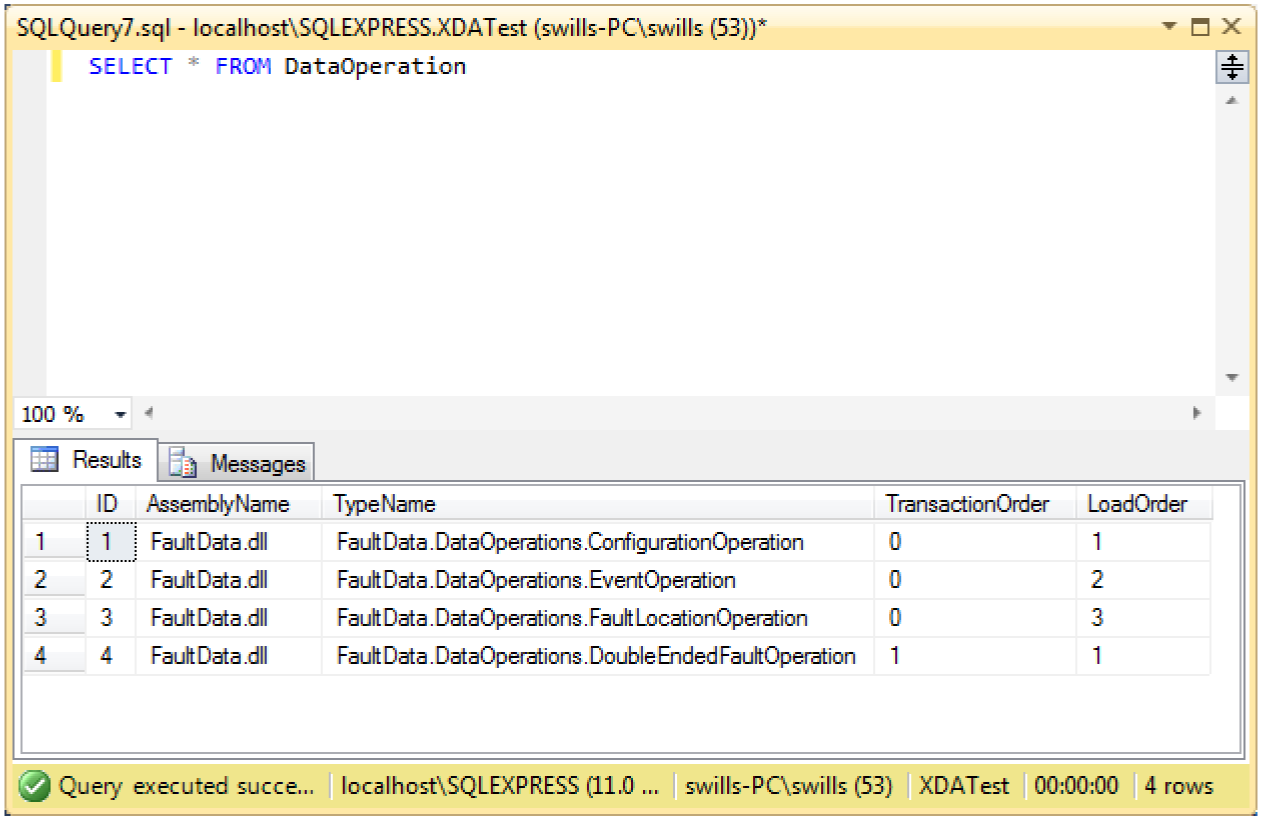
```
public override void Load(DbAdapterContainer dbAdapterContainer)
{
    BulkLoader bulkLoader;
```

```
// Create the bulk loader for loading data into the database
bulkLoader = new BulkLoader();
bulkLoader.Connection = dbAdapterContainer.Connection;
bulkLoader.CommandTimeout = dbAdapterContainer.CommandTimeout;

// Write events to the database
bulkLoader.Load(m_eventTable);
```

```
    // Create the bulk loader for loading data into the database
    bulkLoader = new BulkLoader();
    bulkLoader.Connection = dbAdapterContainer.Connection;
    bulkLoader.CommandTimeout = dbAdapterContainer.CommandTimeout;

    // Write events to the database
    bulkLoader.Load(m_eventTable);
```

snip…

DataOperation



SQLQuery7.sql - localhost\SQLEXPRESS.XDATest (swills-PC\swills (53))*
100 %
Results
Messages

```
SELECT * FROM DataOperation
```

	ID	AssemblyName	TypeName	TransactionOrder	LoadOrder
1	1	FaultData.dll	FaultData.DataOperations.ConfigurationOperation	0	1
2	2	FaultData.dll	FaultData.DataOperations.EventOperation	0	2
3	3	FaultData.dll	FaultData.DataOperations.FaultLocationOperation	0	3
4	4	FaultData.dll	FaultData.DataOperations.DoubleEndedFaultOperation	1	1

Query executed succe... | localhost\SQLEXPRESS (11.0 ... | swills-PC\swills (53) | XDATest | 00:00:00 | 4 rows

TransactionOrder

- Determines the order in which groups of DataOperations will be executed.
  - DataOperations with the same transaction order will be executed as a group.
  - Each group of DataOperations shares a transaction with each other (Load method).
  - If one DataOperation in the group fails, all DataOperations fail!

LoadOrder

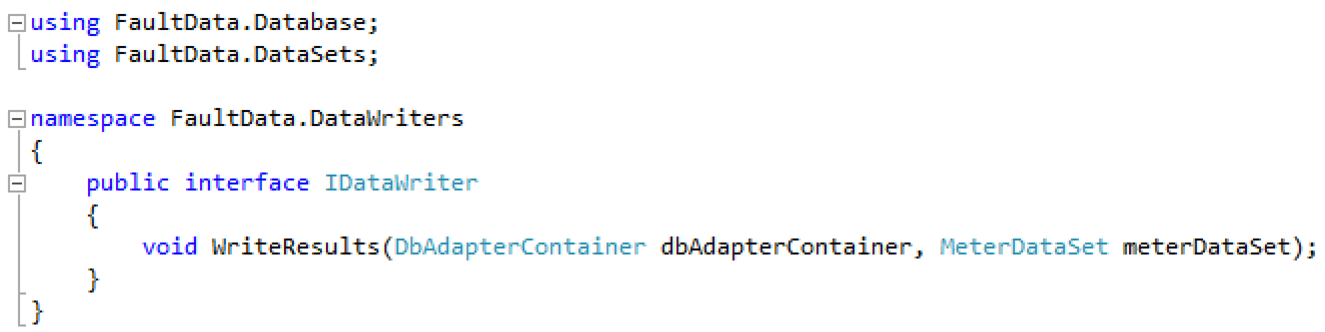
- Determines the order in which DataOperations in the same group will be executed.

DataWriter

- The DataWriter allows you to send the results of your analysis to external systems apart from openXDA.

DataWriter

IDataWriter interface



```
using FaultData.Database;
using FaultData.DataSets;

namespace FaultData.DataWriters
{
    public interface IDataWriter
    {
        void WriteResults(DbAdapterContainer dbAdapterContainer, MeterDataSet meterDataSet);
    }
}
```

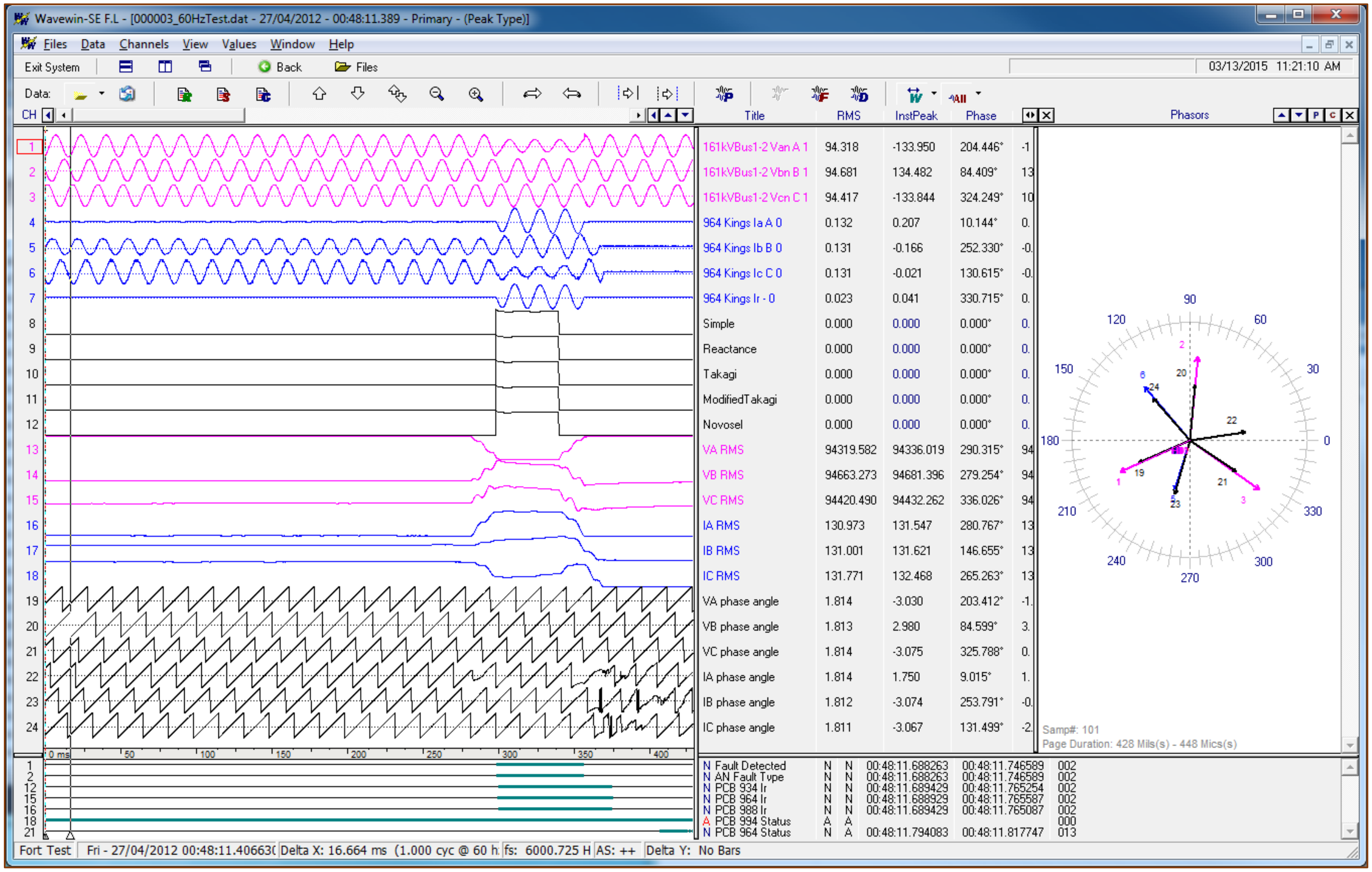
DataWriter

COMTRADEWriter

```
public void WriteResults(DbAdapterContainer dbAdapterContainer, MeterDataSet meterDataSet)
{
    CycleDataResource cycleDataResource;
    FaultDataResource faultDataResource;

    DataGroup dataGroup;
    List<Fault> faults;
    List<int> seriesIDs;
    EventDataSet eventDataSet;

    string rootFileName;
    string fileName;

    cycleDataResource = meterDataSet.GetResource<CycleDataResource>();
    faultDataResource = meterDataSet.GetResource(() => new FaultDataResource(dbAdapterContainer));

    if (!Directory.Exists(m_resultsPath))
        Directory.CreateDirectory(m_resultsPath);

    for (int i = 0; i < cycleDataResource.DataGroups.Count; i++)
    {
        dataGroup = cycleDataResource.DataGroups[i];

        if (faultDataResource.FaultLookup.TryGetValue(dataGroup, out faults))
        {
            rootFileName = FilePath.GetFileNameWithoutExtension(meterDataSet.FilePath);
            fileName = string.Format("{0},{1:000},Line{2}.dat", rootFileName, i, dataGroup.Line.AssetKey);
```

snip...

COMTRADE Output


Wavewin-SE F.L - [000003_60HzTest.dat - 27/04/2012 - 00:48:11.389 - Primary - (Peak Type)]
Files Data Channels View Values Window Help
Exit System
Back
Files
03/13/2015 11:21:10 AM
Data:
CH
Phasors
90
60
30
0
330
300
270
240
210
180
150
120
0 ms
50
100
150
200
250
300
350
400

Title	RMS	InstPeak	Phase	
161kVBus1-2 Van A 1	94.318	-133.950	204.446°	-1
161kVBus1-2 Vbn B 1	94.681	134.482	84.409°	13
161kVBus1-2 Vcn C 1	94.417	-133.844	324.249°	10
964 Kings Ia A 0	0.132	0.207	10.144°	0.
964 Kings Ib B 0	0.131	-0.166	252.330°	-0.
964 Kings Ic C 0	0.131	-0.021	130.615°	-0.
964 Kings Ir - 0	0.023	0.041	330.715°	0.
Simple	0.000	0.000	0.000°	0.
Reactance	0.000	0.000	0.000°	0.
Takagi	0.000	0.000	0.000°	0.
ModifiedTakagi	0.000	0.000	0.000°	0.
Novosel	0.000	0.000	0.000°	0.
VA RMS	94319.582	94336.019	290.315°	94
VB RMS	94663.273	94681.396	279.254°	94
VC RMS	94420.490	94432.262	336.026°	94
IA RMS	130.973	131.547	280.767°	13
IB RMS	131.001	131.621	146.655°	13
IC RMS	131.771	132.468	265.263°	13
VA phase angle	1.814	-3.030	203.412°	-1.
VB phase angle	1.813	2.980	84.599°	3.
VC phase angle	1.814	-3.075	325.788°	0.
IA phase angle	1.814	1.750	9.015°	1.
IB phase angle	1.812	-3.074	253.791°	-0.
IC phase angle	1.811	-3.067	131.499°	-2.

Samp#: 101
Page Duration: 428 Mils(s) - 448 Mics(s)

```
N Fault Detected   N  N  00:48:11.688263  00:48:11.746589  002
N AN Fault Type    N  N  00:48:11.688263  00:48:11.746589  002
N PCB 934 Ir       N  N  00:48:11.689429  00:48:11.765254  002
N PCB 964 Ir       N  N  00:48:11.688929  00:48:11.765587  002
N PCB 988 Ir       N  N  00:48:11.689429  00:48:11.765087  002
A PCB 994 Status   A  A                                    000
N PCB 964 Status   N  A  00:48:11.794083  00:48:11.817747  013
```

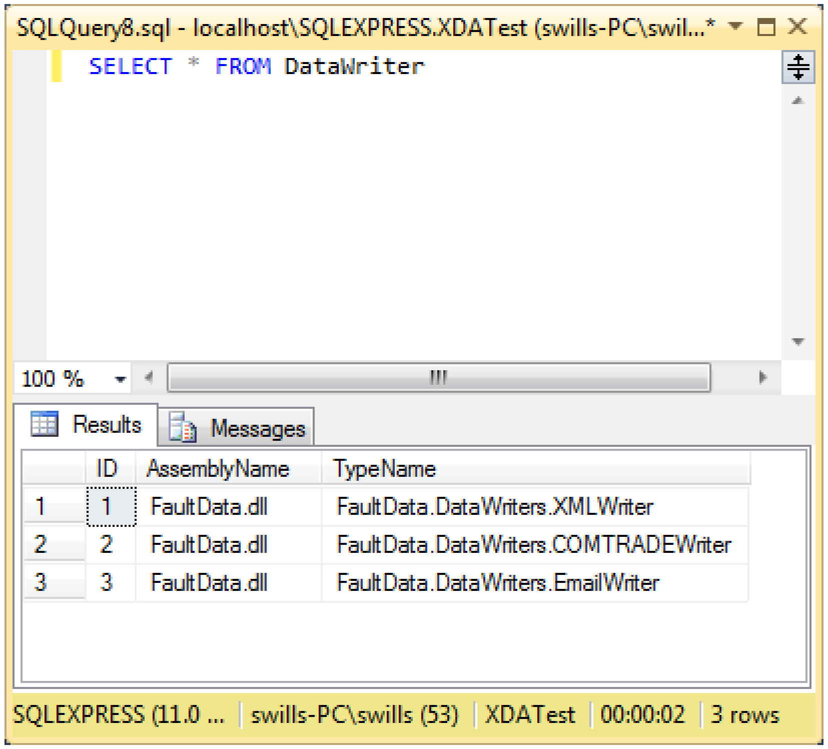
Fort Test | Fri - 27/04/2012 00:48:11.40663(| Delta X: 16.664 ms (1.000 cyc @ 60 h | fs: 6000.725 H | AS: ++ | Delta Y: No Bars

GRID PROTECTION ALLIANCE

GPA User's Forum 2015
Atlanta, Georgia

openXDA EXTENSIBLE DISTURBANCE ANALYTICS

DataWriter



SQLQuery8.sql - localhost\SQLEXPRESS.XDATest (swills-PC\swil...*

```
SELECT * FROM DataWriter
```

	ID	AssemblyName	TypeName
1	1	FaultData.dll	FaultData.DataWriters.XMLWriter
2	2	FaultData.dll	FaultData.DataWriters.COMTRADEWriter
3	3	FaultData.dll	FaultData.DataWriters.EmailWriter

SQLEXPRESS (11.0 ... | swills-PC\swills (53) | XDATest | 00:00:02 | 3 rows

openXDA Outputs

DataOperation vs DataWriter?

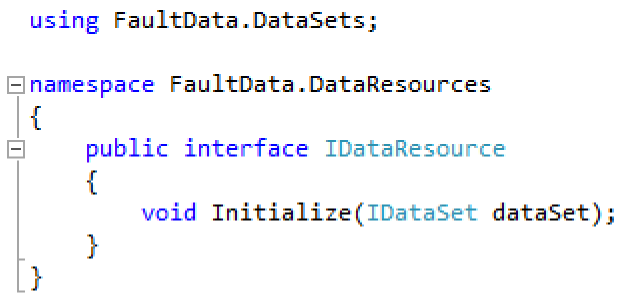
- DataOperation
  - Performs analysis
  - Analytic results saved in database
- DataWriter
  - Automated notifications (e-mail, …)
  - File output consumable by other systems (COMTRADE, …)
  - …

DataResource

- The DataResource allows you to share analytic results as well as analysis routines between DataOperations. Sharing analysis routines allows separate DataOperations to use the results of the analysis regardless of which DataOperations exist in the system.

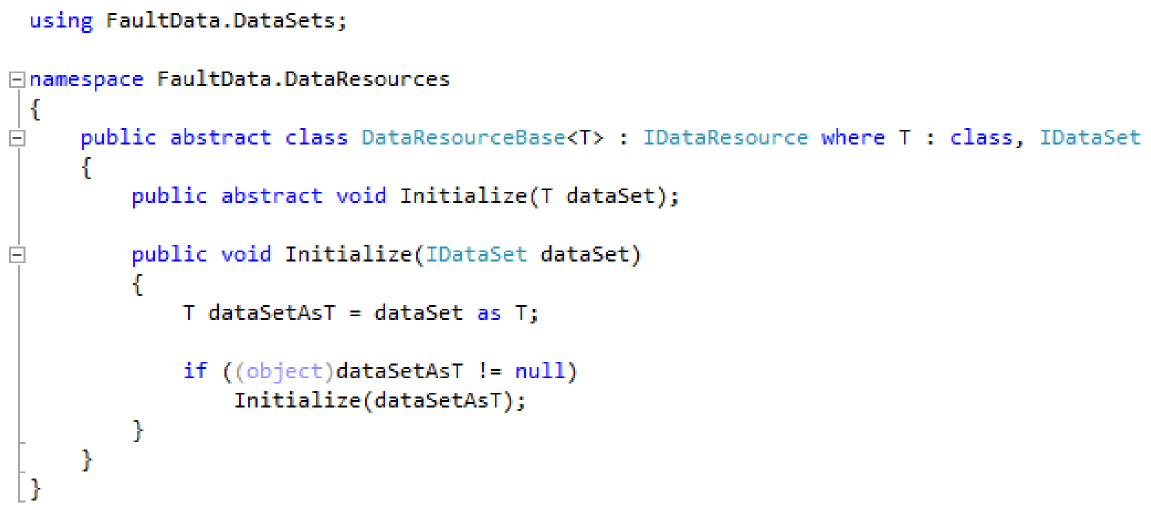
DataResource

IDataResource interface



```
using FaultData.DataSets;

namespace FaultData.DataResources
{
    public interface IDataResource
    {
        void Initialize(IDataSet dataSet);
    }
}
```

DataResourceBase



```
using FaultData.DataSets;

namespace FaultData.DataResources
{
    public abstract class DataResourceBase<T> : IDataResource where T : class, IDataSet
    {
        public abstract void Initialize(T dataSet);

        public void Initialize(IDataSet dataSet)
        {
            T dataSetAsT = dataSet as T;

            if ((object)dataSetAsT != null)
                Initialize(dataSetAsT);
        }
    }
}
```

DataResource

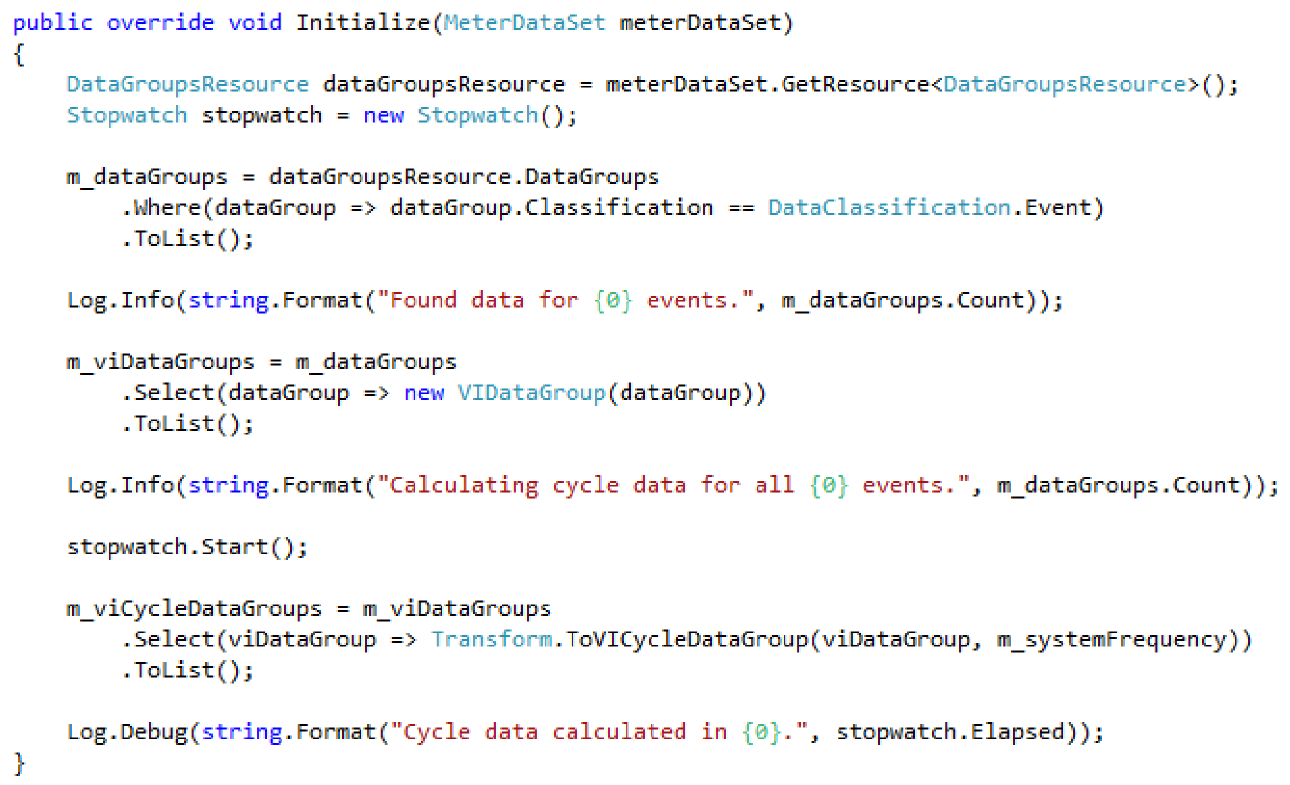
- How it works:
  - DataOperation calls `meterDataSet.GetResource()`.
  - MeterDataSet creates an instance of the DataResource and calls `IDataResource.Initialize()`.
  - MeterDataSet stores the DataResource in a lookup table by type so that subsequent calls to `meterDataSet.GetResource()` will not run the analysis again.

Event Data

- **Time Domain**
  - Event attributes
  - Event segments attributes
  - Waveform
- **Frequency Domain**
  - Cycle data – Full-set of RMS and synchronous component values for each full cycle of data on the waveform

DataResource

CycleDataResource transforms the data from the frequency domain to the time domain. Any DataOperation can use the transformed data, and the analysis will only be performed once.



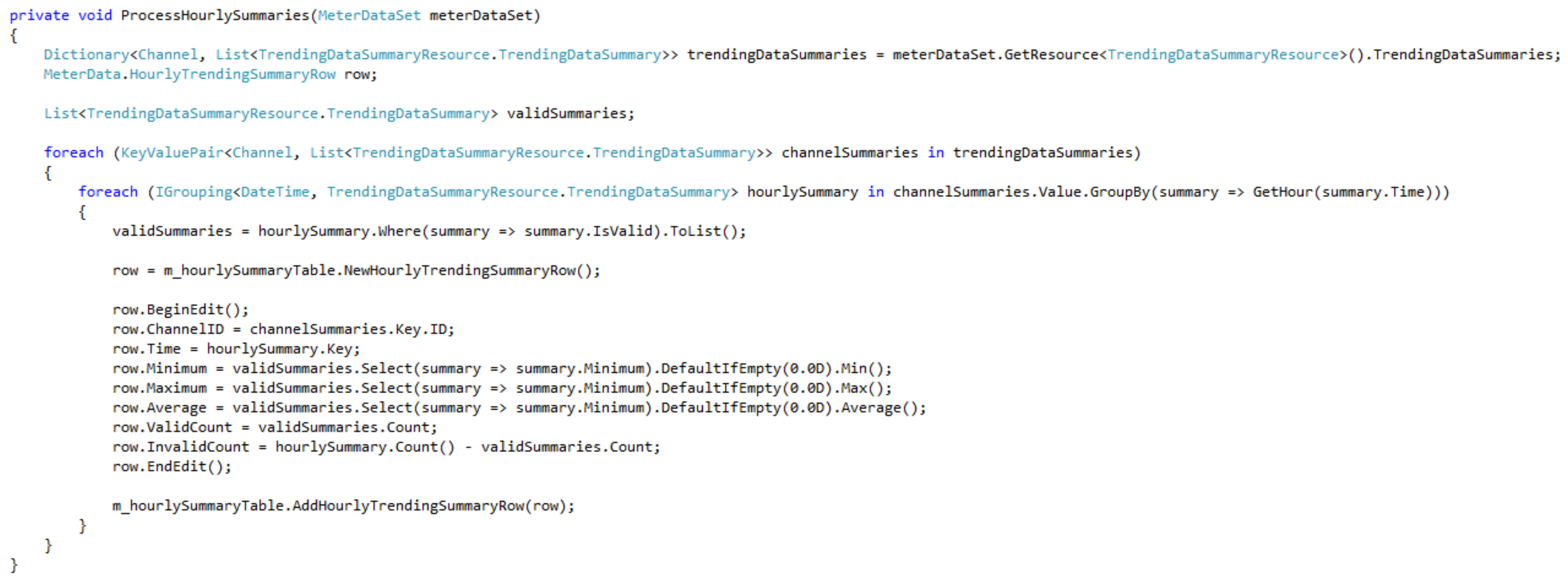
```
public override void Initialize(MeterDataSet meterDataSet)
{
    DataGroupsResource dataGroupsResource = meterDataSet.GetResource<DataGroupsResource>();
    Stopwatch stopwatch = new Stopwatch();

    m_dataGroups = dataGroupsResource.DataGroups
        .Where(dataGroup => dataGroup.Classification == DataClassification.Event)
        .ToList();

    Log.Info(string.Format("Found data for {0} events.", m_dataGroups.Count));

    m_viDataGroups = m_dataGroups
        .Select(dataGroup => new VIDataGroup(dataGroup))
        .ToList();

    Log.Info(string.Format("Calculating cycle data for all {0} events.", m_dataGroups.Count));

    stopwatch.Start();

    m_viCycleDataGroups = m_viDataGroups
        .Select(viDataGroup => Transform.ToVICycleDataGroup(viDataGroup, m_systemFrequency))
        .ToList();

    Log.Debug(string.Format("Cycle data calculated in {0}.", stopwatch.Elapsed));
}
```

Trending Data

- **Daily Values**
  - Min, Max, Average
- **Hourly Values**
  - Min, Max, Average
- **Full Resolution Values**
  - Min, Max, Average

DataResource

The HourlySummaryOperation uses the TrendingDataSummaryResource to group minimum, maximum, and average values by time so that it can further group them by hour. The DailySummaryOperation does the same thing, but groups the trending data by day.



```
private void ProcessHourlySummaries(MeterDataSet meterDataSet)
{
    Dictionary<Channel, List<TrendingDataSummaryResource.TrendingDataSummary>> trendingDataSummaries = meterDataSet.GetResource<TrendingDataSummaryResource>().TrendingDataSummaries;
    MeterData.HourlyTrendingSummaryRow row;

    List<TrendingDataSummaryResource.TrendingDataSummary> validSummaries;

    foreach (KeyValuePair<Channel, List<TrendingDataSummaryResource.TrendingDataSummary>> channelSummaries in trendingDataSummaries)
    {
        foreach (IGrouping<DateTime, TrendingDataSummaryResource.TrendingDataSummary> hourlySummary in channelSummaries.Value.GroupBy(summary => GetHour(summary.Time)))
        {
            validSummaries = hourlySummary.Where(summary => summary.IsValid).ToList();

            row = m_hourlySummaryTable.NewHourlyTrendingSummaryRow();

            row.BeginEdit();
            row.ChannelID = channelSummaries.Key.ID;
            row.Time = hourlySummary.Key;
            row.Minimum = validSummaries.Select(summary => summary.Minimum).DefaultIfEmpty(0.0D).Min();
            row.Maximum = validSummaries.Select(summary => summary.Minimum).DefaultIfEmpty(0.0D).Max();
            row.Average = validSummaries.Select(summary => summary.Minimum).DefaultIfEmpty(0.0D).Average();
            row.ValidCount = validSummaries.Count;
            row.InvalidCount = hourlySummary.Count() - validSummaries.Count;
            row.EndEdit();

            m_hourlySummaryTable.AddHourlyTrendingSummaryRow(row);
        }
    }
}
```

Highlight 4

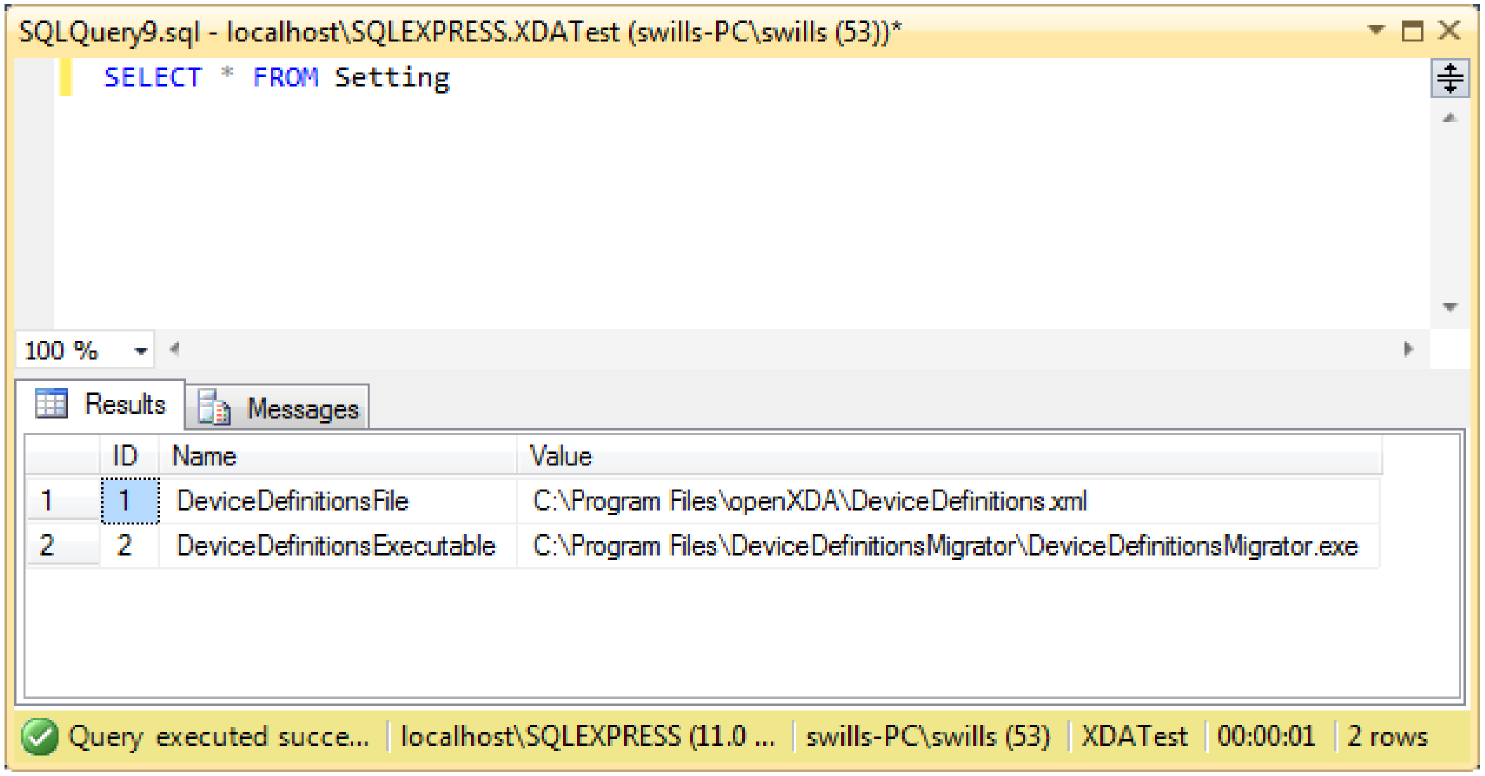
A system settings and logging pattern makes it easy to create and configure new openXDA modules.

System Settings

- System settings allow you to define settings in the database to configure your custom modules. System settings in openXDA are easy to use and can be changed at runtime.

System Settings

The Setting table contains simple name/value pairs.



SQLQuery9.sql - localhost\SQLEXPRESS.XDATest (swills-PC\swills (53))*

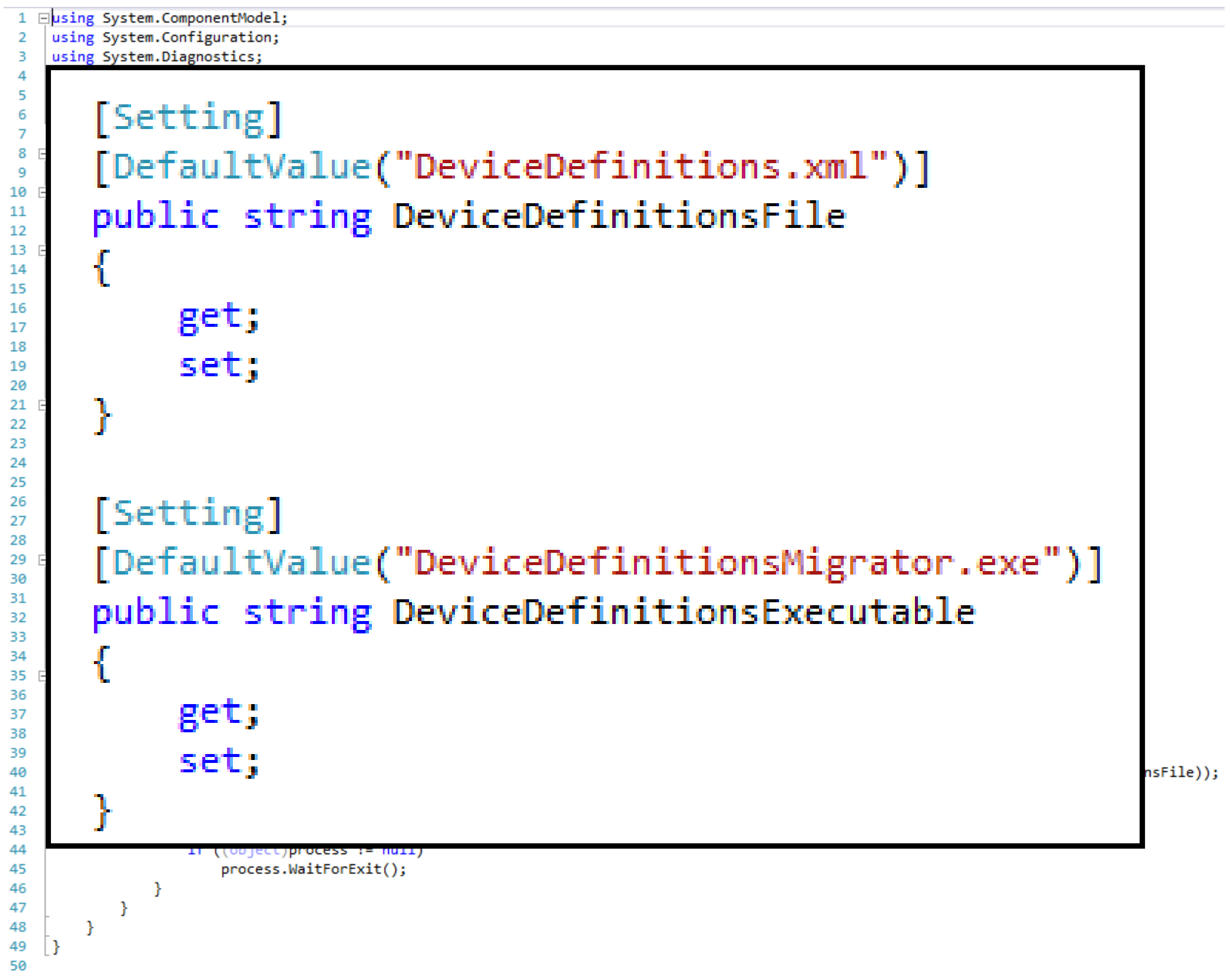
```
SELECT * FROM Setting
```

	ID	Name	Value
1	1	DeviceDefinitionsFile	C:\Program Files\openXDA\DeviceDefinitions.xml
2	2	DeviceDefinitionsExecutable	C:\Program Files\DeviceDefinitionsMigrator\DeviceDefinitionsMigrator.exe

Query executed succe... | localhost\SQLEXPRESS (11.0 ... | swills-PC\swills (53) | XDATest | 00:00:01 | 2 rows

System Settings

DeviceDefinitionsLoader defines system settings via annotated properties.



```
using System.ComponentModel;
using System.Configuration;
using System.Diagnostics;
```

```
[Setting]
[DefaultValue("DeviceDefinitions.xml")]
public string DeviceDefinitionsFile
{
    get;
    set;
}

[Setting]
[DefaultValue("DeviceDefinitionsMigrator.exe")]
public string DeviceDefinitionsExecutable
{
    get;
    set;
}
```

```
nsFile));

            if ((object)process != null)
                process.WaitForExit();
        }
    }
}
}
```

System Settings

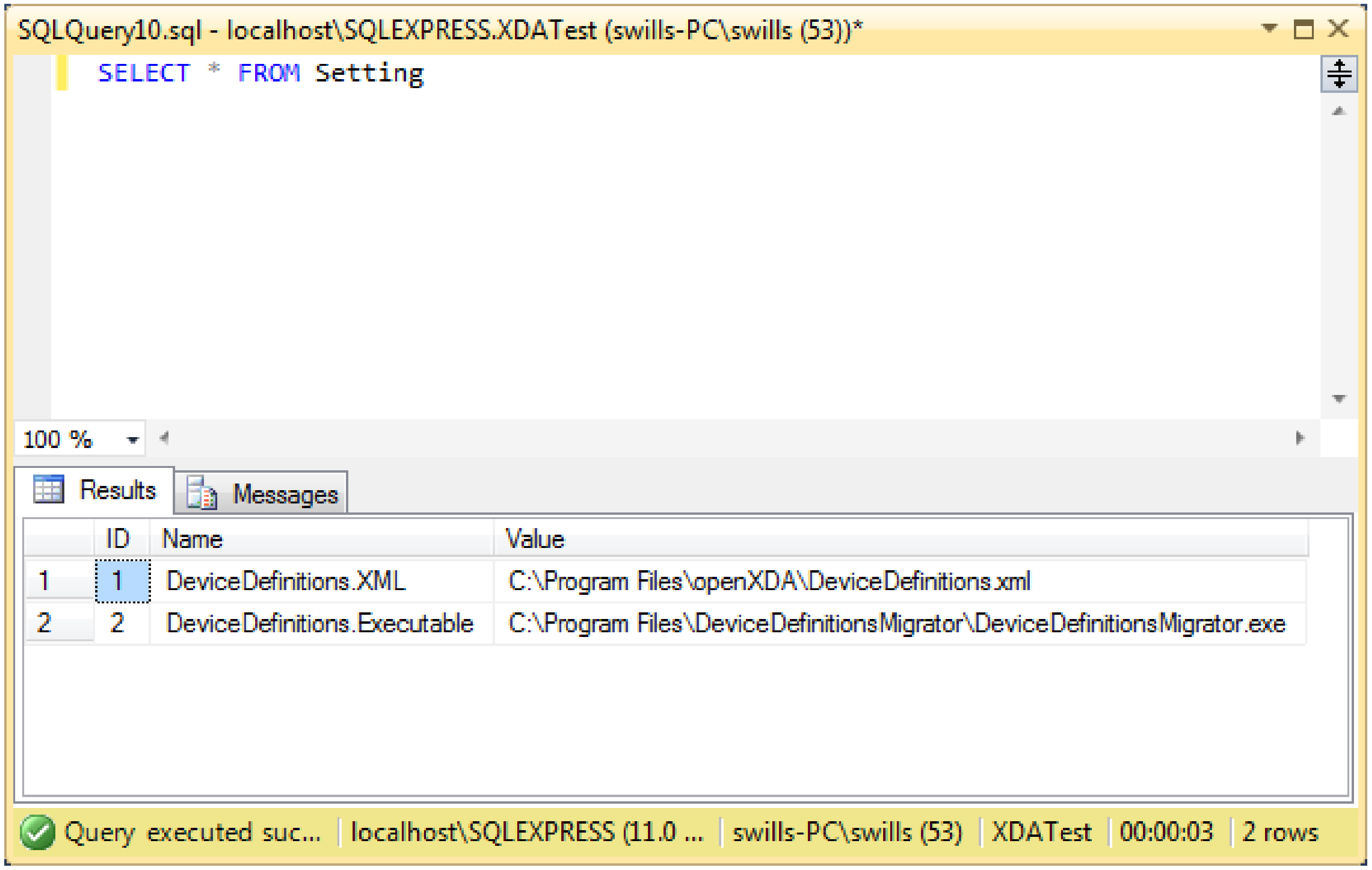
- And that’s it!
- The properties defined by the DeviceDefinitionsLoader are automatically populated based on the names of the properties, the annotations, and the data in the Setting table. This approach works with all the following constructs.
  - ConfigurationLoader
  - DataReader
  - DataOperation
  - DataWriter
  - DataResource

System Settings

- Categorized settings
  - Settings can be placed into categories for better organization. This can help to find settings via database queries, and it can also help to consolidate settings definitions in the source code.

System Settings

Setting names are prefixed by their categories, separated by a '.'



SQLQuery10.sql - localhost\SQLEXPRESS.XDATest (swills-PC\swills (53))*

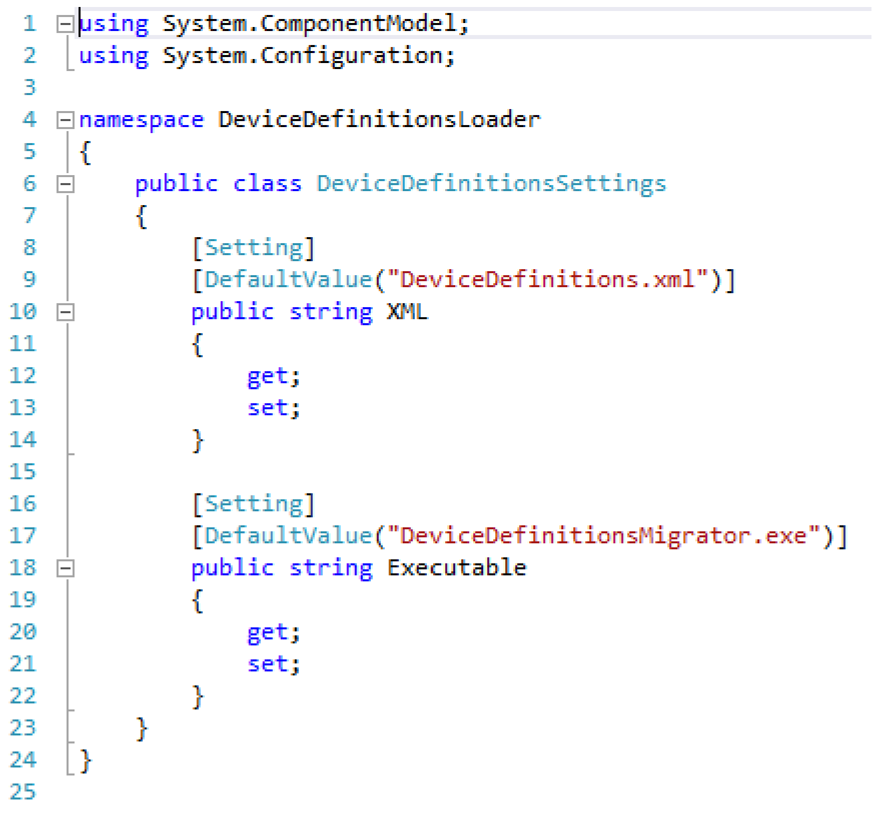
```
SELECT * FROM Setting
```

	ID	Name	Value
1	1	DeviceDefinitions.XML	C:\Program Files\openXDA\DeviceDefinitions.xml
2	2	DeviceDefinitions.Executable	C:\Program Files\DeviceDefinitionsMigrator\DeviceDefinitionsMigrator.exe

Query executed suc... | localhost\SQLEXPRESS (11.0 ... | swills-PC\swills (53) | XDATest | 00:00:03 | 2 rows

System Settings

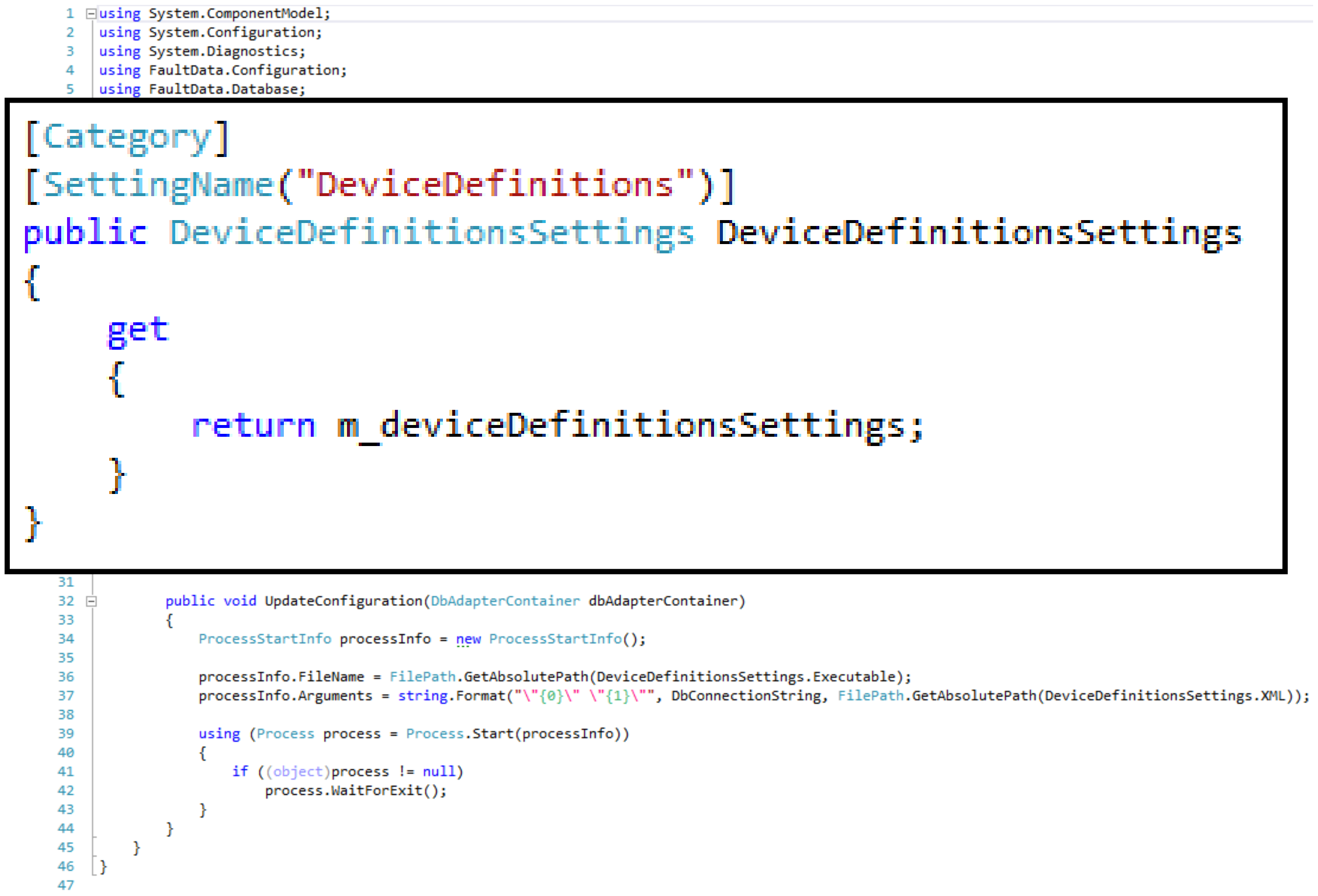
Create a class that defines the settings category. Annotate the properties like you would with normal system settings.



```
using System.ComponentModel;
using System.Configuration;

namespace DeviceDefinitionsLoader
{
    public class DeviceDefinitionsSettings
    {
        [Setting]
        [DefaultValue("DeviceDefinitions.xml")]
        public string XML
        {
            get;
            set;
        }

        [Setting]
        [DefaultValue("DeviceDefinitionsMigrator.exe")]
        public string Executable
        {
            get;
            set;
        }
    }
}

```

System Settings

Modify the configuration loader to annotate a property as a Category. Instantiate a member variable of the type that defines your settings category.



```
using System.ComponentModel;
using System.Configuration;
using System.Diagnostics;
using FaultData.Configuration;
using FaultData.Database;
```

```
[Category]
[SettingName("DeviceDefinitions")]
public DeviceDefinitionsSettings DeviceDefinitionsSettings
{
    get
    {
        return m_deviceDefinitionsSettings;
    }
}
```

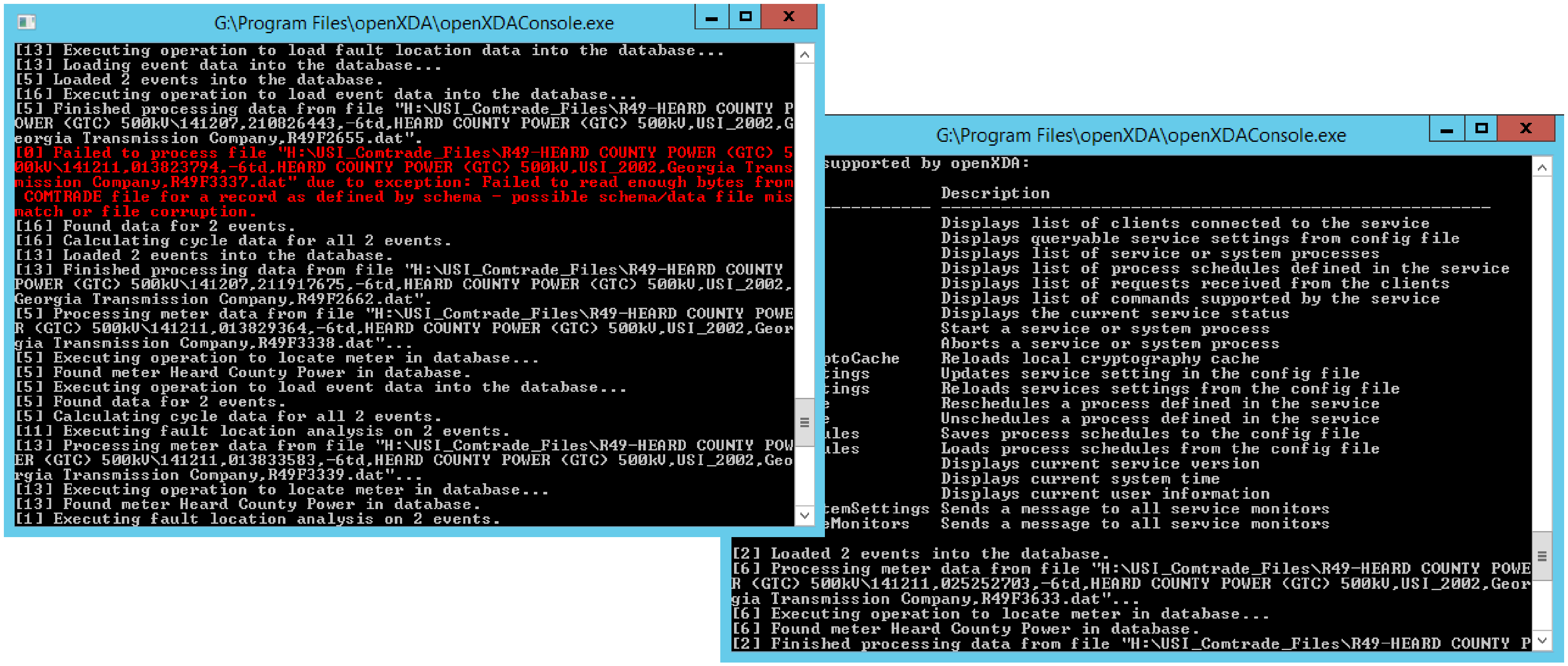
```

        public void UpdateConfiguration(DbAdapterContainer dbAdapterContainer)
        {
            ProcessStartInfo processInfo = new ProcessStartInfo();

            processInfo.FileName = FilePath.GetAbsolutePath(DeviceDefinitionsSettings.Executable);
            processInfo.Arguments = string.Format("\"{0}\" \"{1}\"", DbConnectionString, FilePath.GetAbsolutePath(DeviceDefinitionsSettings.XML));

            using (Process process = Process.Start(processInfo))
            {
                if ((object)process != null)
                    process.WaitForExit();
            }
        }
    }
}
```

openXDA Logging

- Logging allows you to provide messages back to the user for introspection into your modules' activities.
- openXDA uses log4net as its framework for generating log messages. Messages go straight to the remote system console as well as the various log files produced by openXDA.

openXDA Remote Admin Console

- Real-time monitoring of status log
- Interact with service through commands



```
[13] Executing operation to load fault location data into the database...
[13] Loading event data into the database...
[5] Loaded 2 events into the database.
[16] Executing operation to load event data into the database...
[5] Finished processing data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY P
OWER (GTC) 500kV\141207,210826443,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,G
eorgia Transmission Company,R49F2655.dat".
[0] Failed to process file "H:\USI_Comtrade_Files\R49-HEARD COUNTY POWER (GTC) 5
00kV\141211,013823794,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,Georgia Trans
mission Company,R49F3337.dat" due to exception: Failed to read enough bytes from
 COMTRADE file for a record as defined by schema - possible schema/data file mis
match or file corruption.
[16] Found data for 2 events.
[16] Calculating cycle data for all 2 events.
[13] Loaded 2 events into the database.
[13] Finished processing data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY
POWER (GTC) 500kV\141207,211917675,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,
Georgia Transmission Company,R49F2662.dat".
[5] Processing meter data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY POWE
R (GTC) 500kV\141211,013829364,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,Geor
gia Transmission Company,R49F3338.dat"...
[5] Executing operation to locate meter in database...
[5] Found meter Heard County Power in database.
[5] Executing operation to load event data into the database...
[5] Found data for 2 events.
[5] Calculating cycle data for all 2 events.
[11] Executing fault location analysis on 2 events.
[13] Processing meter data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY POW
ER (GTC) 500kV\141211,013833583,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,Geo
rgia Transmission Company,R49F3339.dat"...
[13] Executing operation to locate meter in database...
[13] Found meter Heard County Power in database.
[1] Executing fault location analysis on 2 events.
```

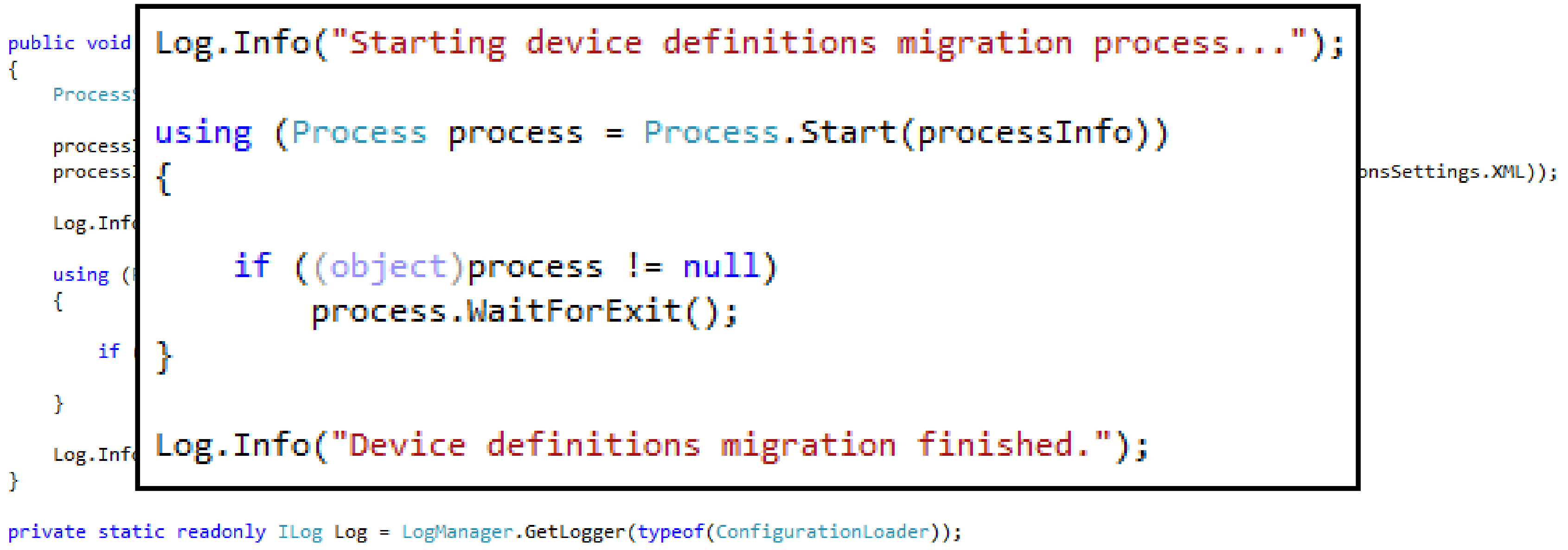
```
supported by openXDA:

               Description
-------------  -----------------------------------------------------------
               Displays list of clients connected to the service
               Displays queryable service settings from config file
               Displays list of service or system processes
               Displays list of process schedules defined in the service
               Displays list of requests received from the clients
               Displays list of commands supported by the service
               Displays the current service status
               Start a service or system process
               Aborts a service or system process
ptoCache       Reloads local cryptography cache
tings          Updates service setting in the config file
tings          Reloads services settings from the config file
e              Reschedules a process defined in the service
e              Unschedules a process defined in the service
ules           Saves process schedules to the config file
ules           Loads process schedules from the config file
               Displays current service version
               Displays current system time
               Displays current user information
temSettings    Sends a message to all service monitors
eMonitors      Sends a message to all service monitors

[2] Loaded 2 events into the database.
[6] Processing meter data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY POWE
R (GTC) 500kV\141211,025252703,-6td,HEARD COUNTY POWER (GTC) 500kV,USI_2002,Geor
gia Transmission Company,R49F3633.dat"...
[6] Executing operation to locate meter in database...
[6] Found meter Heard County Power in database.
[2] Finished processing data from file "H:\USI_Comtrade_Files\R49-HEARD COUNTY P
```

openXDA Logging

- Text File-Based Logging
  - **Status Log** (key messages)
  - **Error Log**
    - Assembly, class, method
    - Stack trace
    - Exception type and message
  - **Debug Log** – All status, most error and copious additional messages

openXDA Logging

Create an ILog object through which to log your messages, then call Log.Info() to log a message.



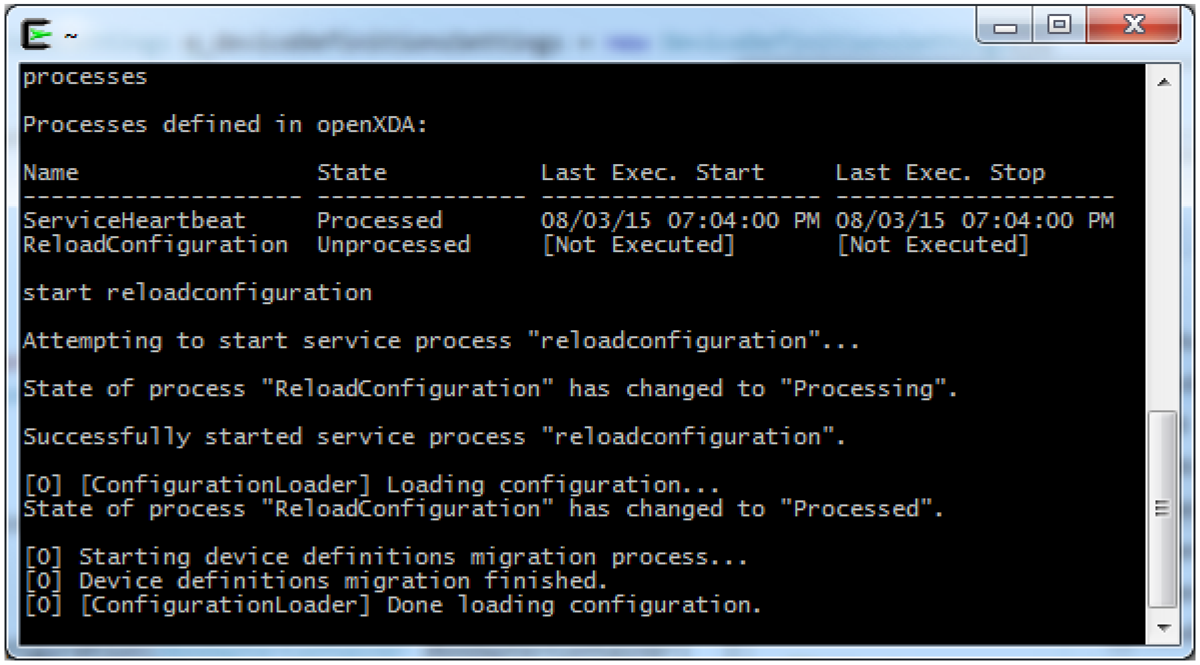
```
public void
{
    Process

    process                                                                    onsSettings.XML));
    process

    Log.Inf

    using (
    {

        if (

    }

    Log.Inf
}
```

```
Log.Info("Starting device definitions migration process...");

using (Process process = Process.Start(processInfo))
{

    if ((object)process != null)
        process.WaitForExit();
}

Log.Info("Device definitions migration finished.");
```

```
private static readonly ILog Log = LogManager.GetLogger(typeof(ConfigurationLoader));
```

openXDA Logging

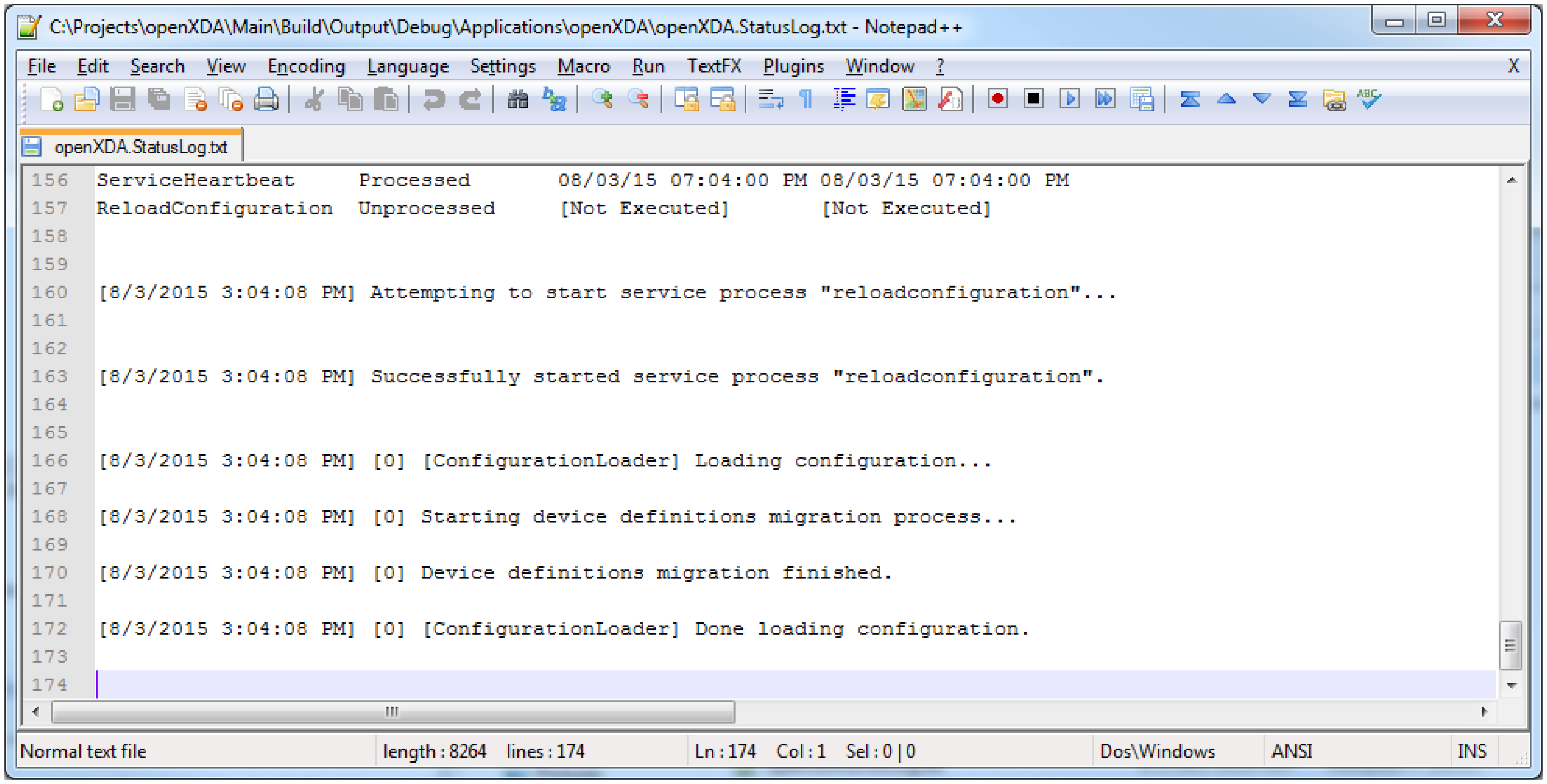
Remote system console



```
processes

Processes defined in openXDA:

Name                  State           Last Exec. Start     Last Exec. Stop
--------------------  --------------  -------------------- --------------------
ServiceHeartbeat      Processed       08/03/15 07:04:00 PM 08/03/15 07:04:00 PM
ReloadConfiguration   Unprocessed     [Not Executed]       [Not Executed]

start reloadconfiguration

Attempting to start service process "reloadconfiguration"...

State of process "ReloadConfiguration" has changed to "Processing".

Successfully started service process "reloadconfiguration".

[0] [ConfigurationLoader] Loading configuration...
State of process "ReloadConfiguration" has changed to "Processed".

[0] Starting device definitions migration process...
[0] Device definitions migration finished.
[0] [ConfigurationLoader] Done loading configuration.
```

openXDA Logging

The status log logs the messages that are printed to the remote system console.

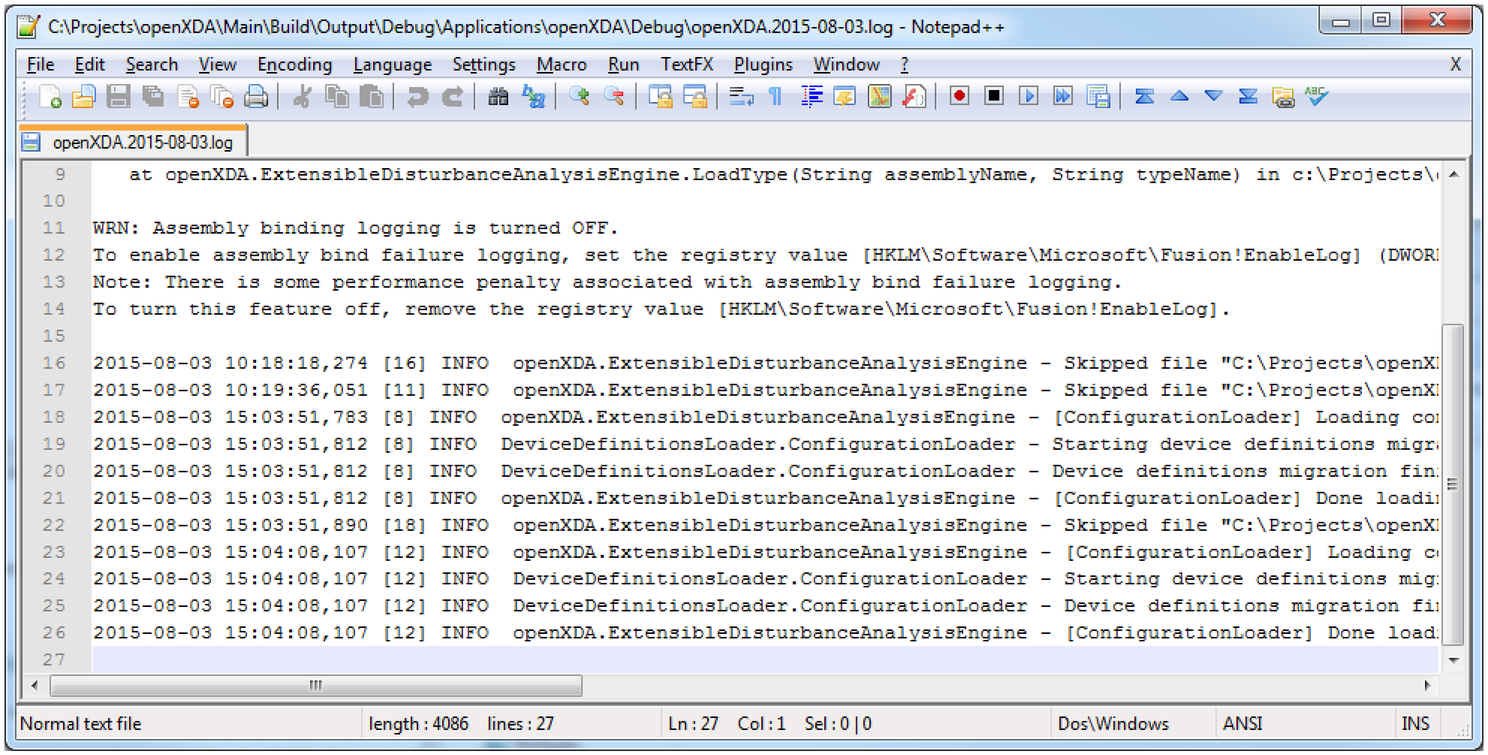

C:\Projects\openXDA\Main\Build\Output\Debug\Applications\openXDA\openXDA.StatusLog.txt - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
openXDA.StatusLog.txt

```
ServiceHeartbeat     Processed        08/03/15 07:04:00 PM 08/03/15 07:04:00 PM
ReloadConfiguration  Unprocessed      [Not Executed]       [Not Executed]


[8/3/2015 3:04:08 PM] Attempting to start service process "reloadconfiguration"...


[8/3/2015 3:04:08 PM] Successfully started service process "reloadconfiguration".


[8/3/2015 3:04:08 PM] [0] [ConfigurationLoader] Loading configuration...

[8/3/2015 3:04:08 PM] [0] Starting device definitions migration process...

[8/3/2015 3:04:08 PM] [0] Device definitions migration finished.

[8/3/2015 3:04:08 PM] [0] [ConfigurationLoader] Done loading configuration.

```

Normal text file | length : 8264 lines : 174 | Ln : 174 Col : 1 Sel : 0 | 0 | Dos\Windows | ANSI | INS

openXDA Logging

The debug log provides additional information about log messages as well as more verbose logging.

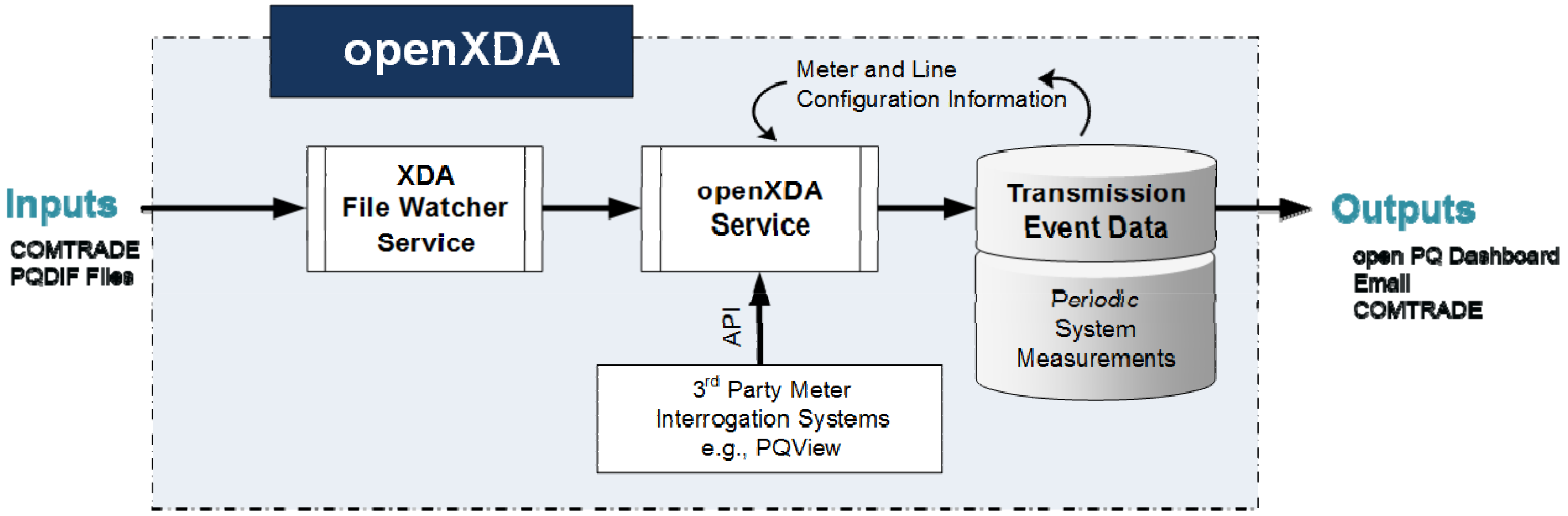

C:\Projects\openXDA\Main\Build\Output\Debug\Applications\openXDA\Debug\openXDA.2015-08-03.log - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
openXDA.2015-08-03.log
Normal text file
length : 4086 lines : 27
Ln : 27 Col : 1 Sel : 0 | 0
Dos\Windows
ANSI
INS

```
   at openXDA.ExtensibleDisturbanceAnalysisEngine.LoadType(String assemblyName, String typeName) in c:\Projects\

WRN: Assembly binding logging is turned OFF.
To enable assembly bind failure logging, set the registry value [HKLM\Software\Microsoft\Fusion!EnableLog] (DWORI
Note: There is some performance penalty associated with assembly bind failure logging.
To turn this feature off, remove the registry value [HKLM\Software\Microsoft\Fusion!EnableLog].

2015-08-03 10:18:18,274 [16] INFO  openXDA.ExtensibleDisturbanceAnalysisEngine - Skipped file "C:\Projects\openXI
2015-08-03 10:19:36,051 [11] INFO  openXDA.ExtensibleDisturbanceAnalysisEngine - Skipped file "C:\Projects\openXI
2015-08-03 15:03:51,783 [8] INFO  openXDA.ExtensibleDisturbanceAnalysisEngine - [ConfigurationLoader] Loading co
2015-08-03 15:03:51,812 [8] INFO  DeviceDefinitionsLoader.ConfigurationLoader - Starting device definitions migr
2015-08-03 15:03:51,812 [8] INFO  DeviceDefinitionsLoader.ConfigurationLoader - Device definitions migration fin
2015-08-03 15:03:51,812 [8] INFO  openXDA.ExtensibleDisturbanceAnalysisEngine - [ConfigurationLoader] Done loadi
2015-08-03 15:03:51,890 [18] INFO  openXDA.ExtensibleDisturbanceAnalysisEngine - Skipped file "C:\Projects\openXI
2015-08-03 15:04:08,107 [12] INFO  openXDA.ExtensibleDisturbanceAnalysisEngine - [ConfigurationLoader] Loading c
2015-08-03 15:04:08,107 [12] INFO  DeviceDefinitionsLoader.ConfigurationLoader - Starting device definitions mig
2015-08-03 15:04:08,107 [12] INFO  DeviceDefinitionsLoader.ConfigurationLoader - Device definitions migration fi
2015-08-03 15:04:08,107 [12] INFO  openXDA.ExtensibleDisturbanceAnalysisEngine - [ConfigurationLoader] Done load
```

openXDA Logging

- Log levels
  - log4net provides a number of log levels that can be used to log messages. Four levels are typically used by openXDA.
    - Log.Error() – Errors and exception handling
    - Log.Warn() – Warning messages
    - Log.Info() – Regular status messages
    - Log.Debug() – Debug messages
  - The remote system console displays different colors based on the log level.
    - Error = Red
    - Warning = Yellow
    - Status = White
    - Debug = Hidden
  - Debug messages only appear in the debug log.

Questions ??


openXDA
Inputs
COMTRADE
PQDIF Files
XDA File Watcher Service
openXDA Service
Meter and Line Configuration Information
Transmission Event Data
Periodic System Measurements
API
3rd Party Meter Interrogation Systems e.g., PQView
Outputs
open PQ Dashboard
Email
COMTRADE