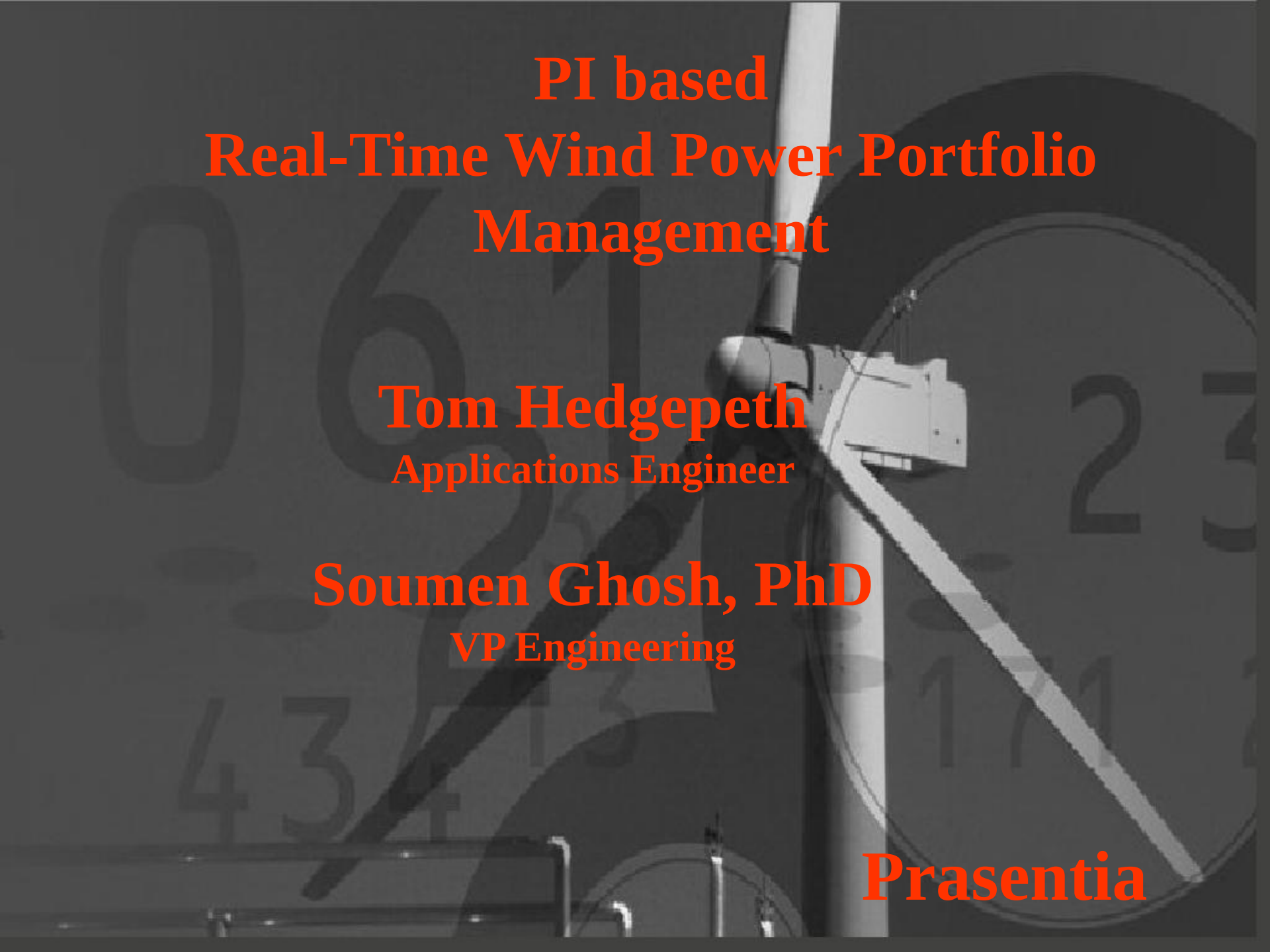




PI based Real-Time Wind Power Portfolio Management

Tom Hedgepeth
Applications Engineer

Soumen Ghosh, PhD
VP Engineering

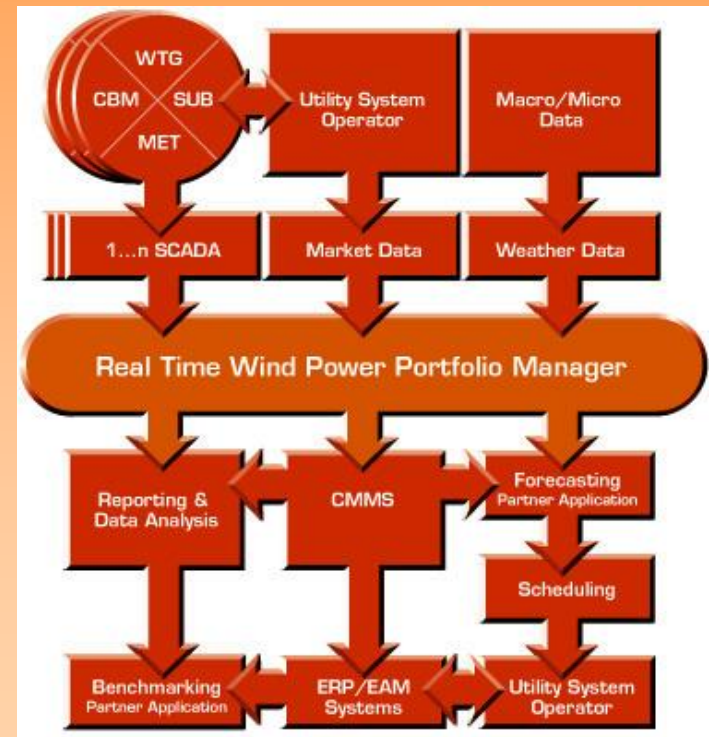
Prasentia

Agenda

- About Prasentia
- Current Configuration
- Standardizing Multiple Turbine Types
- Creating Faults – Unit Batches
- Using Visual Studio, Crystal Reports, and the PI-SDK
- Lessons Learned
- Questions

About Prasentia

- Best of breed technology integrator specializing in Wind Power.
- SCADA:
 - Siemens Simatic IT Historian
 - Communication Directly from SCADA to WTG Controller, Meteorological Station, and Substation Controllers
 - OPC Compliant
- Reporting and Data Analysis:
 - OSISoft PI Historian
 - Visual Studio .Net 2003
 - Crystal Reports
 - ASP.Net
 - C#
- Visualization Tools:
 - OSISoft ICE Platform
 - ProcessBook and DataLink



Previous Wind Farm Management Tools

- An off-line information and reporting service attached to individual wind farm
- Based on home-grown database, (Microsoft Access, SQL, other). Required large data transfers to perform portfolio analysis.
- Reports available around the tenth or twelfth of the month
- Not timely enough to give decision makers the advantage to proactively steer the project in the most optimal directions.

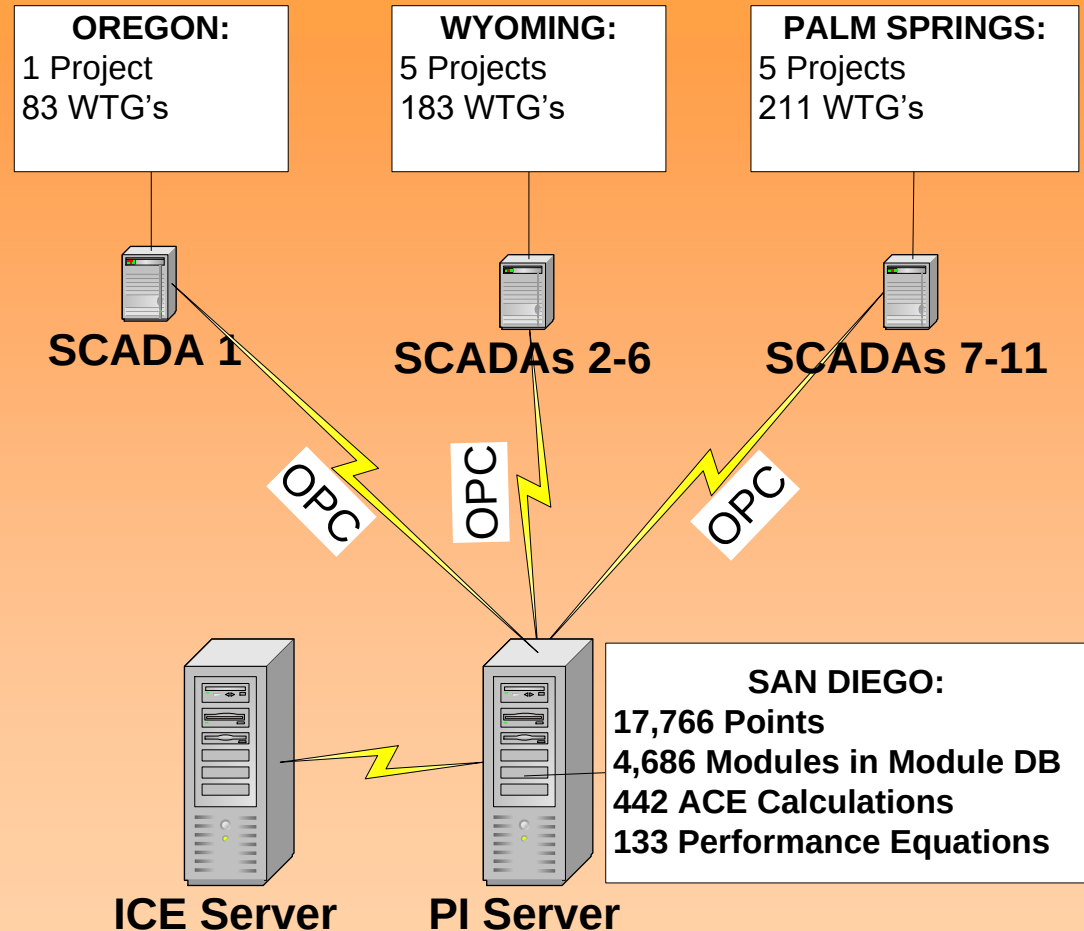
Genesis of RT-WPPM

- To provide wind industry with
 - portfolio based strategic information in real-time
 - applications to optimize wind assets and performance
- To help stakeholders
 - maximize the economics of wind projects by delivering critical information in real-time from the **same data** source to every level

Current Configuration

WTG Models

- Micon
 - 48/700
 - 750
- Mitsubishi
 - MWT 600
 - MWT 450/600
 - MWT 1000
 - MWT 600-47
- Vestas
 - V47

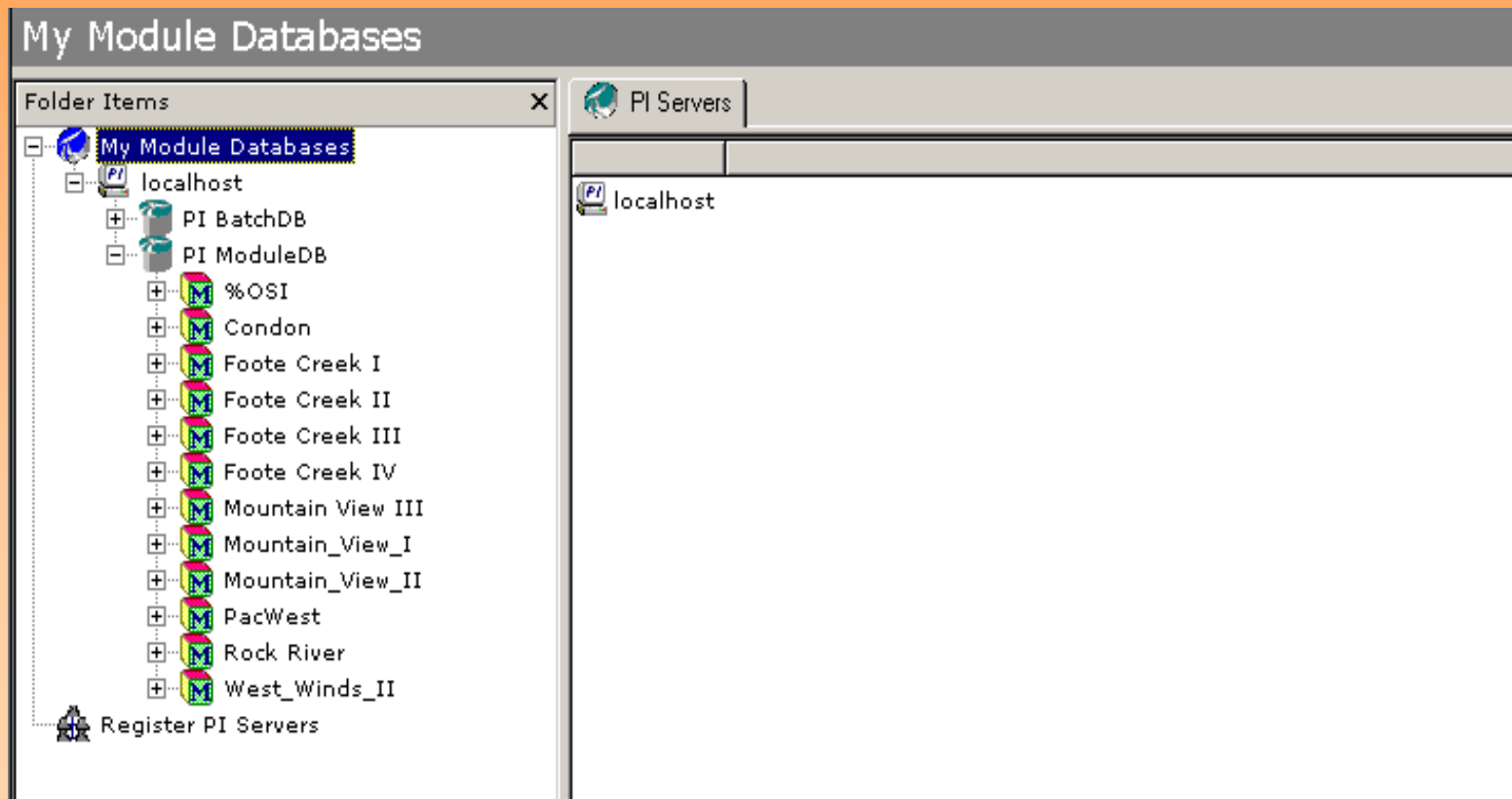


Standardizing Multiple Turbine Types

- **Goal: Provide a user friendly “Alias” to cryptic point names while standardizing multiple vendors for reporting purposes.**
- **Facts:**
 - All wind turbines have the same basic components.
 - Reporting requirements generally require the same basic inputs.

Standardizing Multiple Turbine Types

- **Solution: OsiSoft Module Database!**



WHY PI Module Data Base

- Nature of Wind Farm has a consistent hierarchy
 - Wind Farm (Project)
 - Wind Turbine Generator
 - Substation
 - Meteorological Station
- Prasentia applications use reusable elements to map PI Data for different user requirements
 - Operator, Owner, Performance Engr, Turbine Manufacturer
- Efficient access to Tags using Aliases for PI data sources.
- Manage additions and changes to the relationship of Wind Farm data over time
- Use Batch processes to track turbine status and faults

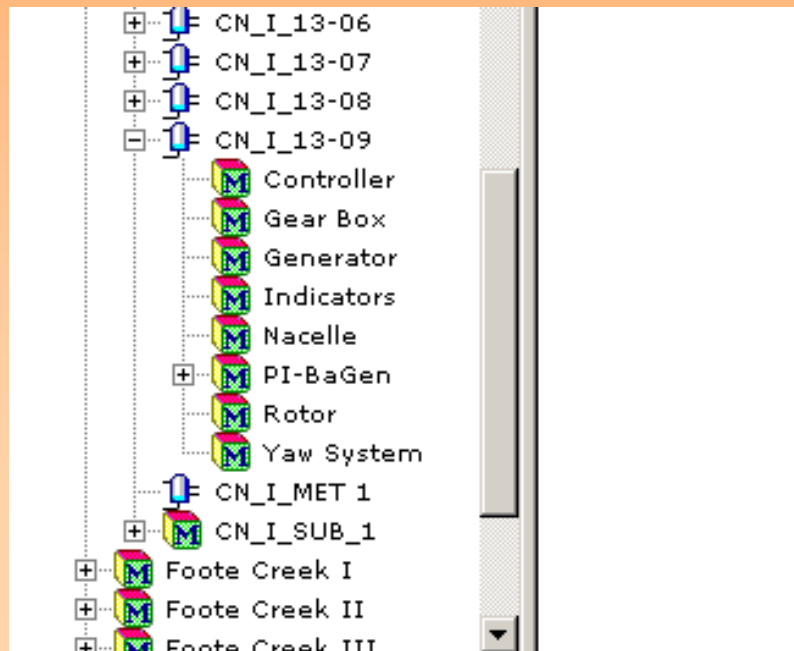
Our Module Database Structure

- Project 1
 - WTG 1
 - Controller
 - Nacelle
 - etc
 - WTG 2
 - Substation
 - Met 1
- Project 2

Has Properties

Has Aliases

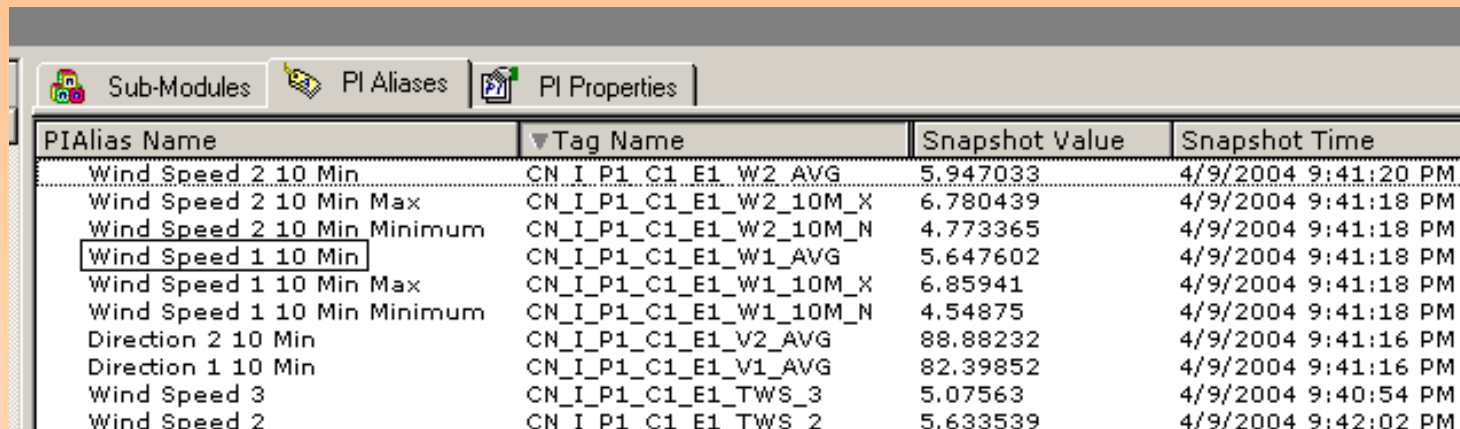
Is
PiUnit





Module Database & Batch Terminology

- Alias: “Aliases provide a translation from site-specific plant instrument tags to more generic, process-oriented names, which are more intuitive to users” (OSISoft Devnet).
 - CN_I_P1_C1_E1_TWS1 = Project \ Met 1 \ Wind Speed 1
 - CN_I_P1_C1_M001_VGEN = Project \ WTG 1 \ Generator \ RPM

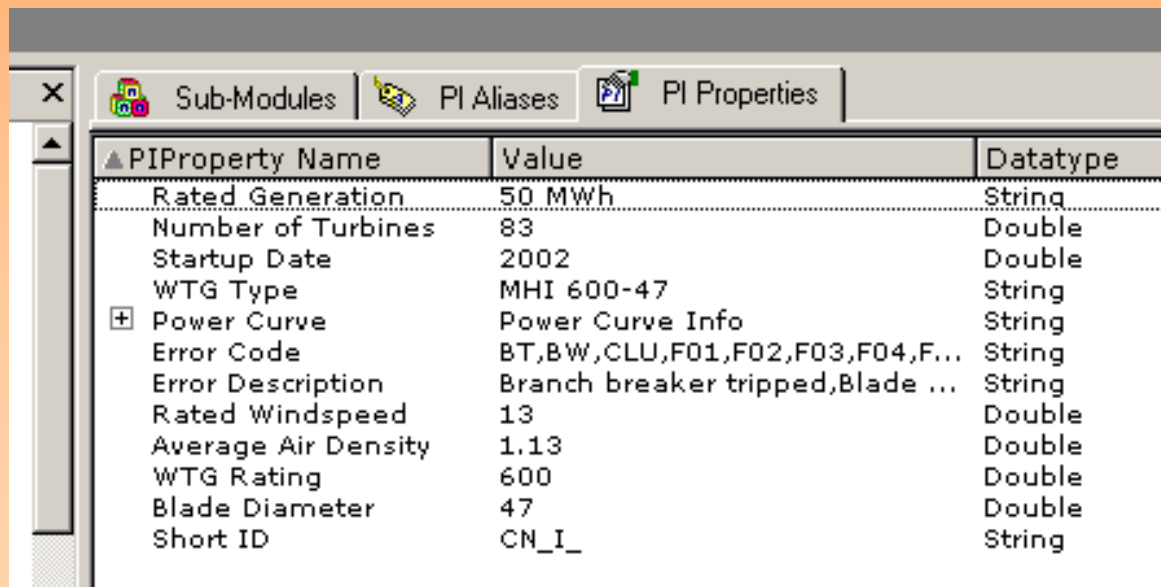


The screenshot shows a software window titled 'PI Properties' with a tab labeled 'PI Aliases'. The window contains a table with four columns: 'PIAlias Name', 'Tag Name', 'Snapshot Value', and 'Snapshot Time'. The table lists various wind speed and direction aliases for a specific plant.

PIAlias Name	Tag Name	Snapshot Value	Snapshot Time
Wind Speed 2 10 Min	CN_I_P1_C1_E1_W2_AVG	5.947033	4/9/2004 9:41:20 PM
Wind Speed 2 10 Min Max	CN_I_P1_C1_E1_W2_10M_X	6.780439	4/9/2004 9:41:18 PM
Wind Speed 2 10 Min Minimum	CN_I_P1_C1_E1_W2_10M_N	4.773365	4/9/2004 9:41:18 PM
Wind Speed 1 10 Min	CN_I_P1_C1_E1_W1_AVG	5.647602	4/9/2004 9:41:18 PM
Wind Speed 1 10 Min Max	CN_I_P1_C1_E1_W1_10M_X	6.85941	4/9/2004 9:41:18 PM
Wind Speed 1 10 Min Minimum	CN_I_P1_C1_E1_W1_10M_N	4.54875	4/9/2004 9:41:18 PM
Direction 2 10 Min	CN_I_P1_C1_E1_V2_AVG	88.88232	4/9/2004 9:41:16 PM
Direction 1 10 Min	CN_I_P1_C1_E1_V1_AVG	82.39852	4/9/2004 9:41:16 PM
Wind Speed 3	CN_I_P1_C1_E1_TWS_3	5.07563	4/9/2004 9:40:54 PM
Wind Speed 2	CN_I_P1_C1_E1_TWS_2	5.633539	4/9/2004 9:42:02 PM

Other Terms

- Property: “Used to store static information about a module, like equipment specifications” (OSISoft Devnet)
- PIUnit: Classifies the Module as a Batch Unit.



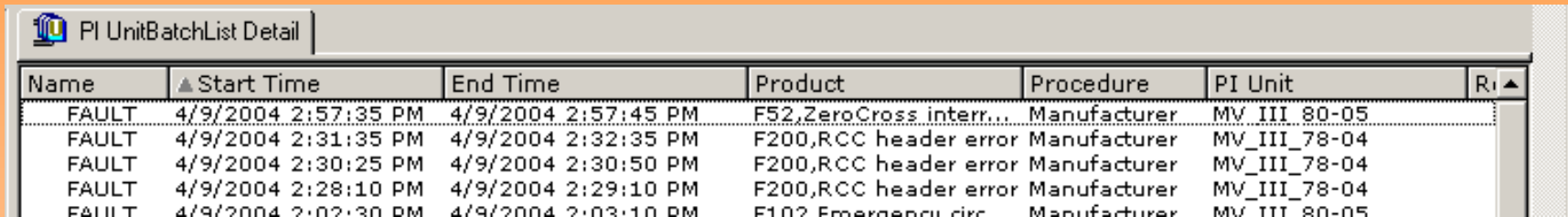
PIProperty Name	Value	Datatype
Rated Generation	50 MWh	String
Number of Turbines	83	Double
Startup Date	2002	Double
WTG Type	MHI 600-47	String
Power Curve	Power Curve Info	String
Error Code	BT,BW,CLU,F01,F02,F03,F04,F...	String
Error Description	Branch breaker tripped,Blade ...	String
Rated Windspeed	13	Double
Average Air Density	1.13	Double
WTG Rating	600	Double
Blade Diameter	47	Double
Short ID	CN_I_	String

Using Unit Batches to track Down Time

- PI Batch Generator Interface:
 - Used to automatically create UnitBatches and SubBatches.
 - Monitors 1 active point to start/stop UnitBatches.
 - Allows assignment of tags for associating values to the Product and Procedure fields.

Using Unit Batches to track Downtime

- Prasentia's Standard Configuration:



The screenshot shows a software window titled "PI UnitBatchList Detail". It contains a table with the following columns: Name, Start Time, End Time, Product, Procedure, and PI Unit. The table lists several fault events from April 9, 2004.

Name	Start Time	End Time	Product	Procedure	PI Unit
FAULT	4/9/2004 2:57:35 PM	4/9/2004 2:57:45 PM	F52,ZeroCross interr...	Manufacturer	MV_III_80-05
FAULT	4/9/2004 2:31:35 PM	4/9/2004 2:32:35 PM	F200,RCC header error	Manufacturer	MV_III_78-04
FAULT	4/9/2004 2:30:25 PM	4/9/2004 2:30:50 PM	F200,RCC header error	Manufacturer	MV_III_78-04
FAULT	4/9/2004 2:28:10 PM	4/9/2004 2:29:10 PM	F200,RCC header error	Manufacturer	MV_III_78-04
FAULT	4/9/2004 2:02:30 PM	4/9/2004 2:03:10 PM	F102 Emergency circ	Manufacturer	MV_III_80-05

- WTG Status Point is used as the “Active Point” and “Batch ID”.
- Procedure is standardized as a Downtime Category.
- Product is used for a description of the fault.

Visual Studio 2003

Projects
from
Module
Database

Select Report
File Help

PRASENTIA
OPTIMIZING WINDPOWER PERFORMANCE

PROJECT PORTFOLIO

- Condon
- Foote Creek I
- Foote Creek II
- Foote Creek III
- Foote Creek IV
- Mountain View III
- Mountain_View_I
- Mountain_View_II
- PacWest

REPORT PARAMETERS

Start Date/Time: 04/09/2004 4:19 PM

End Date/Time: 04/09/2004 4:19 PM

☒ 1 Report per Project
☐ All on 1 Report

☒ All WTGs
☐ Selected WTGs

REPORTS

☐ Met Summary Sectors: 16
☐ Sub Summary
☐ Downtime Detail
☐ Production Detail
☐ Custom Report

☐ Warranty Detail
☐ Fault Detail
☐ Fault Summary
☐ Power Curve
☐ Scatter Plot

☐ Vestas Report

Retrieve Reports

SPECIAL SELECTIONS

Visual Studio 2003

Select Report

File Help

PRASENTIA
OPTIMIZING WINDPOWER PERFORMANCE

PROJECT PORTFOLIO

Condon
Foote Creek I
Foote Creek II
Foote Creek III
Foote Creek IV
Mountain View III
Mountain_View_I
Mountain_View_II
PacWest

REPORT PARAMETERS

Start Date/Time 04/09/2004 4:19 PM

End Date/Time 04/09/2004 4:19 PM

☒ 1 Report per Project
☐ All on 1 Report

☐ All WTGs
☒ Selected WTGs

REPORTS

☐ Met Summary ☐ Sub Summary ☐ Downtime Detail ☐ Production Detail ☐ Custom Report

Sectors 16

☐ Warranty Detail ☐ Fault Detail ☐ Fault Summary ☐ Power Curve ☐ Scatter Plot

☐ Vestas Report

Retrieve Reports

SPECIAL SELECTIONS

Condon

Add selected WTG

CN_I_03-02
CN_I_03-03
CN_I_03-04
CN_I_04-01
CN_I_04-02
CN_I_04-03
CN_I_04-04

	Project	WTG
*		

Turbines from Module Database

Crystal Reports

Fault Detail

File

MainReport

Sub-Total	High Wind	2.50	2.50	1,200	1,200	1	High Wind	shutdown - AVERAGE	Related Faults
Summary for: CN_I_06-08								1 Total Faults	
CN_I_06-08									
03/18/04 09:56	03/18/04 11:54	2.17	2.17	1,100	1,100	S02	High Wind	High Wind shutdown - AVERAGE	
Sub-Total	High Wind	2.17	2.17	1,100	1,100	1	High Wind	Related Faults	
Summary for: CN_I_06-08								1 Total Faults	
CN_I_06-09									
03/18/04 09:57	03/18/04 11:56	2.33	2.33	1,100	1,100	S02	High Wind	High Wind shutdown - AVERAGE	
Sub-Total	High Wind	2.17	2.17	1,200	1,200	1	High Wind	Related Faults	
Summary for: CN_I_06-09								1 Total Faults	
CN_I_06-10									
03/18/04 09:57	03/18/04 11:56	2.33	2.33	1,100	1,100	S02	High Wind	High Wind shutdown - AVERAGE	
Sub-Total	High Wind	2.17	2.33	1,100	1,100	1	High Wind	Related Faults	
03/17/04 17:11	03/17/04 19:29	2.17		139	F28	Manufacturer		Yaw motor slow or high speed running	

Date Printed: 4/9/2004

Page 7 of 18

Current Page No: 7 Total Page No: 18 Zoom Factor: 100%

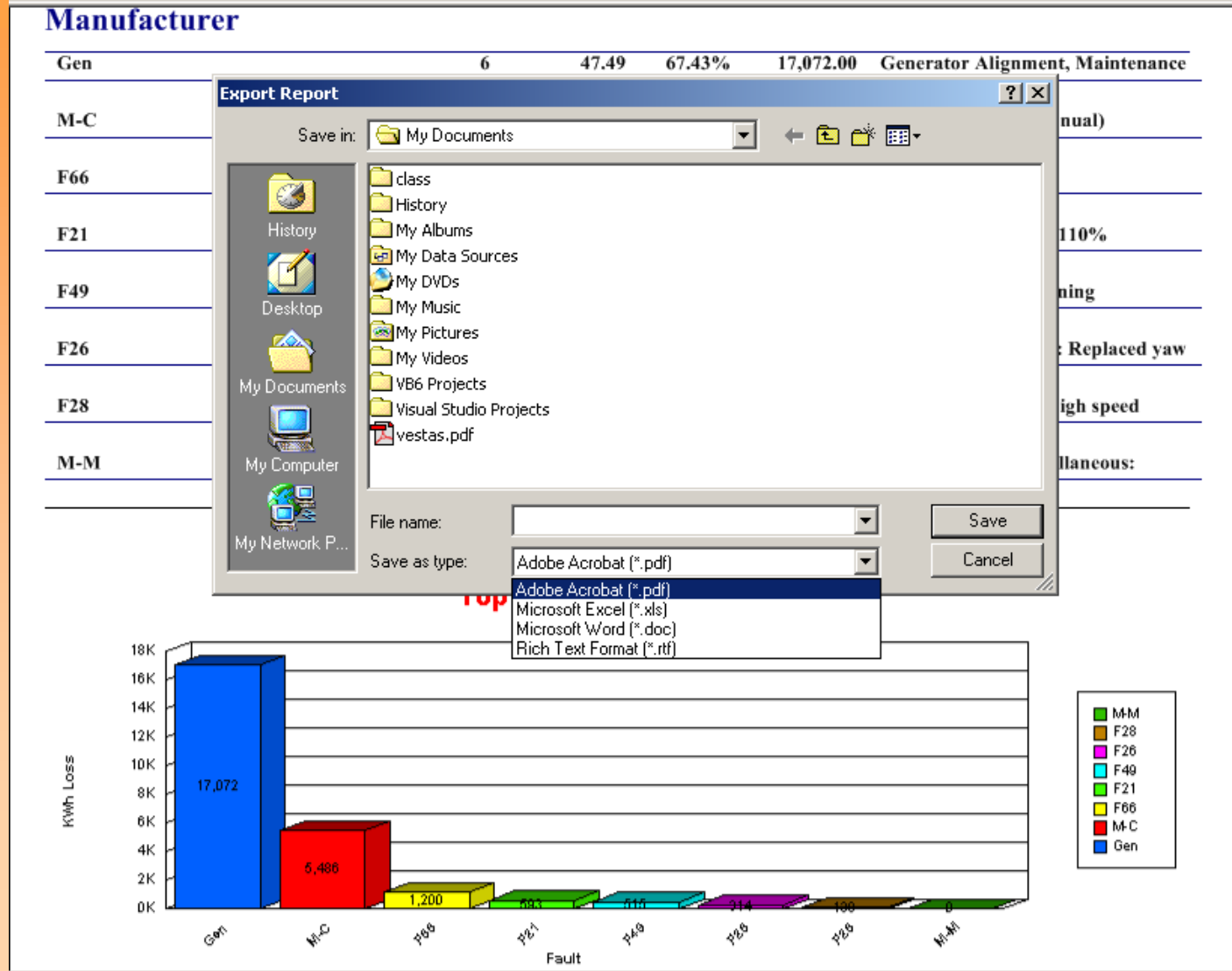
Module Name

Procedure

Product

Aliased Tags

Crystal Reports



Editing Turbine Faults

fAlarmEdit

PRASENTIA™
OPTIMIZING WINDPOWER PERFORMANCE

Project Portfolio:

- Condon
- Foote Creek I
- Foote Creek II
- Foote Creek III
- Foote Creek IV
- Mountain View III

Report Parameters:

Start Date: 03/23/2004 12:00 AM

End Date: 03/24/2004 12:00 AM

Retrieve Values

Print Pre-Edit Report

WTG Alarms | Communication Errors

WTG	From	To	Hours	Down	Error	Notes
CN_I_08-02	3/23/2004 7:11:17 AM	3/23/2004 8:15:09 AM	01:03:52	Manufacturer	M-M	Maintenance - Miscellaneous
CN_I_09-01	3/23/2004 9:13:03 AM	3/23/2004 2:18:11 PM	05:05:08	Manufacturer	M-C	Maintenance - C (2nd year)
CN_I_09-02	3/23/2004 2:58:39 PM	3/23/2004 3:17:48 PM	00:19:09	Manufacturer	M-C	Maintenance - C (2nd year)
CN_I_09-09	3/23/2004 12:40:57 PM	3/23/2004 1:10:55 PM	00:29:58	Manufacturer	F01	Controller failure at Tower ground
CN_I_10-04	3/23/2004 8:59:19 AM	3/23/2004 9:08:05 AM	00:08:46	Manufacturer	M-C	Maintenance - C (2nd year)
CN_I_08-03	3/23/2004 8:20:33 AM	3/23/2004 12:55:00 PM	04:34:27	Manufacturer	M-M	Maintenance - Miscellaneous: Pitch Ins

WTG Specific List from Project "Properties"

M-M

M-SR

None

OF

O-O

PART

PL

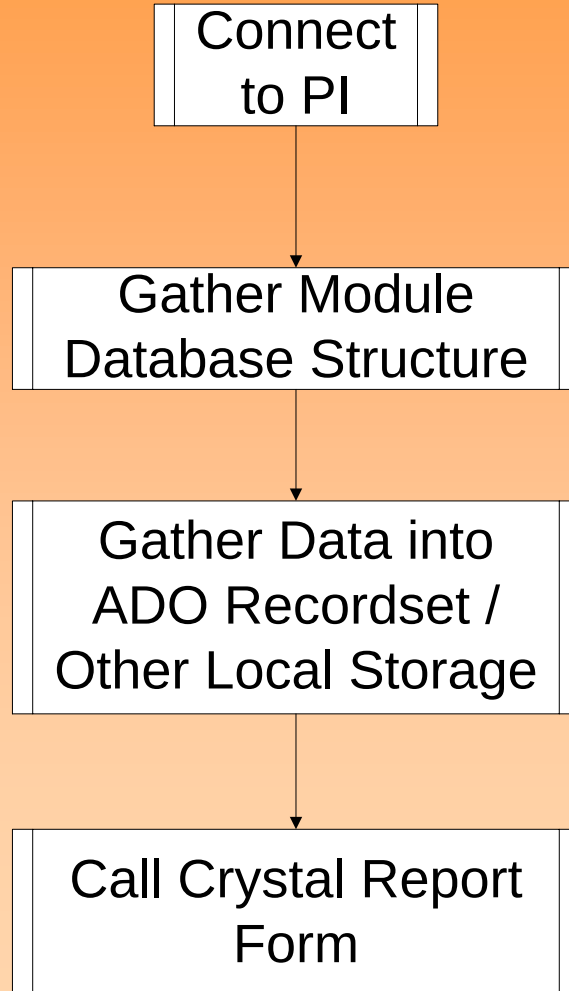
PLD

Status Flags Description

*

Add Alarm Edit Alarm Join Alarm Delete Alarm Close

Visual Studio 2003 in a Nutshell



Visual Studio 2003: How To

Connect to PI

```
PISDK.PISDK mySDK;  
mySDK = new PISDK.PISDKClass();  
g_PIServer = mySDK.Servers[strServer];  
  
if (!(g_PIServer.Connected))  
{  
    PISDKDlg.Connections cnxn = new PISDKDlg.ConnectionsClass();  
    cnxn.Login(ref g_PIServer,ref strUN,ref strPW, true, false);  
}
```

Visual Studio 2003: How To

Gather Module Database Structure

1. Create a PIModule “WarrModule”
2. Set WarrModule equal to the desired project in the Database
3. Create a PIProperty “propRating”
4. Set a PIProperty equal to the desired Property

```
PISDK.PIModule WarrModule;  
WarrModule = g_PIServer.PIModuleDB.PIModules[strProject];  
PISDK.PIProperty propRating;  
propRating = WarrModule.PIProperties["WTG Rating"];
```

Visual Studio 2003: How To

Gather Data into
ADO Recordset /
Other Local Storage

1. Set the “GeneratorMod” to a Turbines sub module.
2. Retrieve the desired Alias
3. Retrieve the point associated with the alias
4. Retrieve the data from the archive, this could also be a call for multiple values

```
PISDK.PIModule GeneratorMod;  
GeneratorMod = WarrModule.PIModules[“CN_I_01-01”].PIModules[“Generator”];
```

```
PISDK.PIAlias pa;  
pa = GeneratorMod.PIAliases[“RPM”];
```

```
PISDK.PIPoint pt;  
pt = mySDK.GetPoint( @"\" + @strServer + @"\" + @pa.DataSource.Name);  
  
DataEnd = pt.Data.ArcValue(dtEnd,PISDK.RetrievalTypeConstants.rtInterpolated,null);
```


Visual Studio 2003: Crystal Reports

FaultLossReport - Microsoft Visual C#.NET [design] - FaultSummary.rpt*

File Edit View Project Build Debug Format Tools Window Help

Debug WarrModule

100%

(Setup1) | cReporter.cs* | VestasFaultFin.rpt | VestasFaultSummary.rpt | **FaultSummary.rpt*** | ReportChooser.cs | FaultFin.rpt | Substation.rpt | Form1.cs

Field Explorer

- Database Fields
 - FaultLossDetail
 - ID
 - WTG
 - StartTime
 - EndTime
 - TotFaultHou
 - IntervalHou
 - TotFault
 - IntervalLoss
 - PrimaryFault
 - DowntimeCat
 - Description
 - AdjStart
 - AdjEnd
 - IntervalStar
 - IntervalEnd
 - Project
 - ReportStart
 - ReportEnd
- Formula Fields
- Parameter Fields
- Group Name Fields
- Running Total Fields
- Special Fields
- Unbound Fields

Grouped Data

PrimaryFault	Number of Occurences	Hours	% of Total Loss	KWh Loss	Description
Group Header #1a: FaultLossDetail.DowntimeCat - A (Section27)					
Group Header #1b: FaultLossDetail.DowntimeCat - A (Section27)					
Group #1 Name					
Group Header #2: FaultLossDetail.PrimaryFault - A (Section15)					
Group #2 Name	fail.PrimaryFault	ervalHours	intervalLoss	fail.IntervalLoss	Max of FaultLossDetail.Description
Details (Section3)					
WTG	StartTime	EndTime	IntervalHours	IntervalLoss	
Group Footer #2: FaultLossDetail.PrimaryFault - A (Section15)					
Group Footer #1a: FaultLossDetail.DowntimeCat - A (Section27)					

Zoom Data

Top 10 KWh Loss / Fault
For @DowntimeCat

Year	KWh Loss
1993	100
1994	200
1995	300

Output

Ready

Visual Studio 2003: How To

Call Crystal Report
Form

```
public dsFD ds4FaultReport
{
    set
    {
        ds = value;
        // Just set the report source, everything else is done from the
        //report chooser form.
        crReportDocument.SetDataSource(ds);
        //Set the viewer to the report object to be previewed.
        crystalReportViewer1.ReportSource = crReportDocument;
    }
}
```

Lessons Learned

- **Visual Studio, Dots, & the Module Database:**

- Traversing the module database can be slow if not done properly.

- Slow Method:

```
for (int i = 1; i <= modProj.PIModules.Count; i++)  
{  
    lbWTGs.Items.Add(modProj.PIModules[i].Name);  
}
```

- Faster Method:

```
PISDK.PIModules mods = modProj.PIModules;  
foreach(PISDK.PIModule mod in mods)  
{  
    lbWTGs.Items.Add(mod.Name);  
}
```

Questions?

