https://www.halvorsen.blog

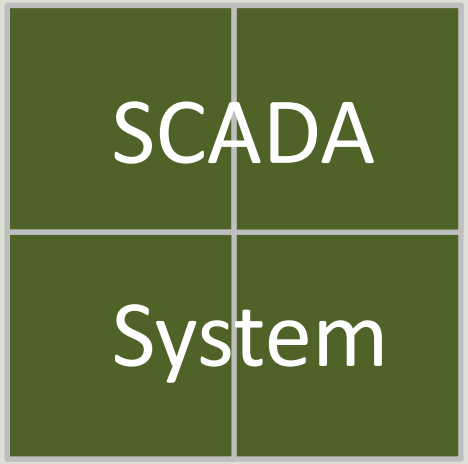
# SCADA System

## Supervisory Control and Data Acquisition

Hans-Petter Halvorsen

# SCADA System


Previous Topics will be the foundation/backbone for the SCADA System




These pieces will be built into a final SCADA System

# Background

- You work as a **System Engineer** in the R&D department in a System Engineering Company.
- Your Assignment is to **Develop a next generation SCADA System** in form of a Prototype/PoC.
- The system should be **module-based** and include a **Control Module**, a **Datalogging Module**, and an **Alarm Module**.
- **OPC** should typically be used for communication between the different Modules. You can choose between OPC DA and OPC UA.
- Data should be stored in a **SQL Server Database**.
- You need to design a general and flexible Database structure that is suitable for the system.
- To create proper and user-friendly **GUI/HMI** is an important part of the Prototype.
- **Note! You can freely choose the Programming Languages and Frameworks to use in the different parts.**
- **The delivery is a Scientific Paper** where you shall give an overview of the entire system made, including the Methods used and the Results archived.
- The Scientific Paper shall be published in an international Journal in competition with many others, so it is important that you "**Add Value**" and stand out compared to the others in order to be selected.
- The delivered Scientific Papers will be assessed by a Committee and only the best contributions will be selected to be published in the international Journal.

# System Requirements

- Design the **Database** using erwin Data Modeler software.
- Implement the Database using **SQL Server.**
- Create a **Control System** and send data to an OPC Server. Use the Air Heater System.
  - Start with a model of the system. When the Simulations works, use the the real system. USB-6008 should be used as an interface between the Application and the Air Heater.
  - You should create and use your own PI(D) controller and Lowpass Filter from Scratch.
  - Write Data to an OPC Server (e.g., MatrikonOPC Simulation Server or OPC UA Server Simulator).
- **Datalogging System**: Read Data from OPC Server and Log the Data to a SQL Server Database
- **Alarm System**: Create an Alarm Generation and Monitoring System. The Alarm System can either be a Windows Application or a Web Application.
- The different subsystems should be implemented as separate Applications because they should be able to run on different computers in a network (distributed).
- **Cyber Security**
  - Deal with and get an overview of relevant Cyber Security issues within your system.

These are the complete requirements for the assignment. The rest of this document contains resources like additional information, code examples, tips and tricks, step-by step instructions, etc. that you can use at your own discretion.

# SCADA Resources

Hans-Petter Halvorsen

Table of Contents

# Table of Contents

# Introduction

Hans-Petter Halvorsen

Table of Contents

# SCADA System Example


USB-6008

Air Heater

Process Plant

MV

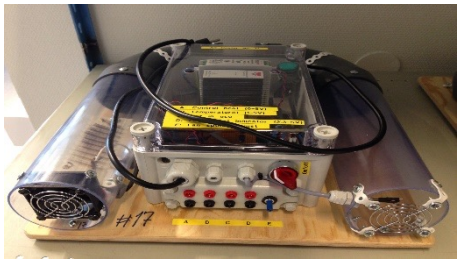
PV

Control System

PV, MV

OPC Server

Simulator/Real Process



PV, MV

Network

Datalogging System

PV, MV

SQL Database

Microsoft SQL Server

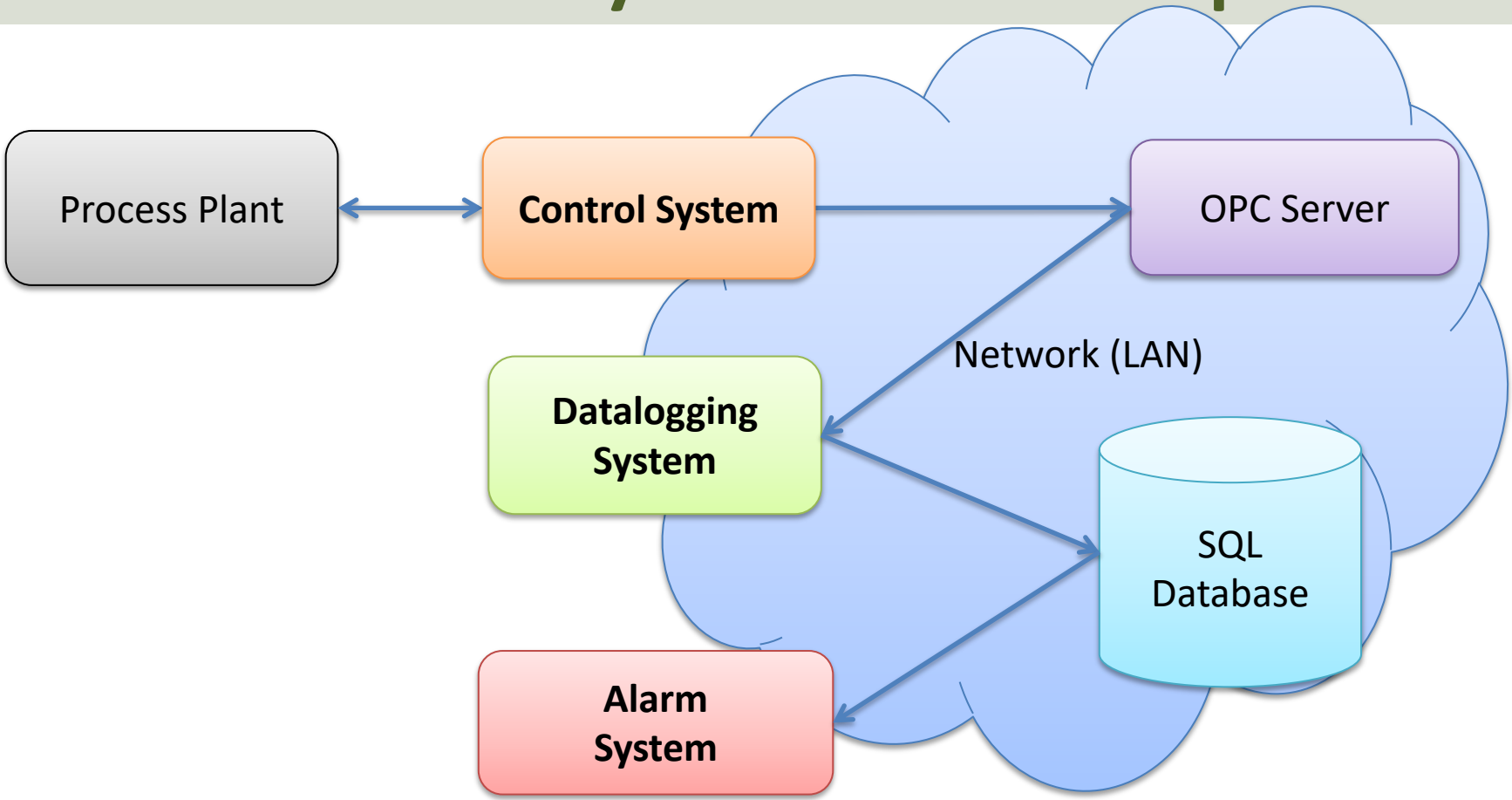
LL, L, H, HH

Alarm System

Alarm Trigger

The SCADA system should be distributed, meaning that the different components could be located on different computers in a network

# SCADA System Example



# Learning Goals

- Learn key concepts within SCADA systems
- Learn more about Database Systems, DAQ and OPC (Data Communication)
- Learn practical skills and implementation of SCADA systems
- Learn more Programming
- Learn about Hardware-Software Interactions
- Learn Practical Skills and Implementations in general
- Learn Software Installation in general, which can be cumbersome with many pitfalls
- Learn to use and create Industrial Software Systems in general
- Learn to Design and Develop Software needed by a given client and follow a set of requirements given by the client
- Problem Solving: Learn to solve unexcepted Problems during Development of a given System

# Software


erwin

Visual Studio

NATIONAL INSTRUMENTS
LabVIEW

MATLAB
The Language of Technical Computing

Microsoft®
SQL Server®

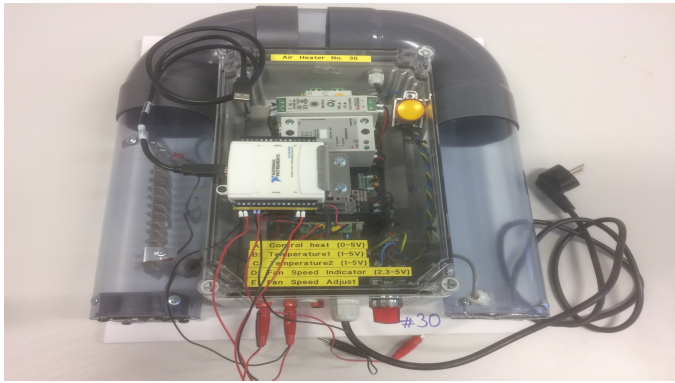
OPC Software

# Hardware


Your Personal Computer

USB-6008


Air Heater




Used as test the Control System

RFID Reader (optional)

Can be used for Login, 2FA, etc.

Industrial PID Controller (optional)


Can be integrated in the SCADA system and/or used to compare results from Control Application

The teacher have not done all the Tasks in detail, so he may not have all the answers! That's how it is in real life also!

Very often it works on one computer but not on another. You may have other versions of the software, you may have installed it in the wrong order, etc...
In these cases, Google is your best friend!


The Teacher dont have all the answers (very few actually ☹)!! Sometimes you just need to “Google” in order to solve your problems, Collaborate with other Students, etc. Thats how you Learn!

# Troubleshooting & Debugging

Use the **Debugging Tools** in your Programming IDE.
Visual Studio, LabVIEW, etc. have great Debugging Tools! Use them!!


“Google It”!

You probably will find the answer on the Internet


Another person in the world probably had a similar problem

Use available Resources such as User Guides, Datasheets, Textbooks, Tutorials, Examples, Tips & Tricks, etc.

My System is not Working??

Multimeter, etc.


Use Microsoft Teams

Check your electric circuit, electrical cables, DAQ device, etc. Check if the wires from/to the DAQ device is correct. Are you using the same I/O Channel in your Software as the wiring suggest? etc.

# Lab Assignment Guidelines

- If you miss assumptions for solving some of the problems, you may define proper assumptions yourself.
- The Tasks described in the Assignment are somewhat loosely defined and more like guidelines, so feel free to interpret the Tasks in your own way with a personalized touch.
- Feel free to **Explore**! Make sure to **Add Value** and **Creativity** to your Applications!
- Try to add some extra value and be creative compared to the simplified examples given by me, in that way you learn so much more.

# Lab Assignment Guidelines

- Think about the Lab Assignment as a small real-life industrial Project, and not a set of tasks or exercises.
- What does the company that hire you expect from you when you deliver this project? What kind of Quality is expected?
- Try to see your work in a larger context than just a Lab Assignment or a set of exercises.
- Try to see the big picture. The tasks within the assignment are just just small building blocks that ends up with a fully working system.
- It is recommended that you make a Work Plan and a System Sketch that gives you an overview of what YOU should do

# Lab Work Requirements

- Make sure to see the “**Big picture**” – you don’t need to document every single step you have made. Focus on what’s important (your final system).
- Your GUIs is important! - make sure to make them user friendly and intuitive. You create this on behalf of someone that are going to use your applications.
- Make sure to always add **Units** in your GUI, charts, documentation, etc.
- **Presenting values with 4+ decimals makes no sense!** E.g., a temperature sensor is not that accurate. You can easily change number of decimals that you present in your GUI in LabVIEW, C#, etc.
- The **Quality** of the LabVIEW code is important. Make sure to use "straight lines" in your LabVIEW code, etc. The code should also flow from left to right, not opposite direction. You create this on behalf of someone that are going to use your applications. Neat code makes it easier to develop, maintain, find code errors, etc.
- In general, make sure that you take some pride in your applications and the work that you do. It's not about getting finished as soon as possible. The mission is to learn as much as possible within a given timeframe. Try to change the mindset.
- Add Value

https://www.halvorsen.blog

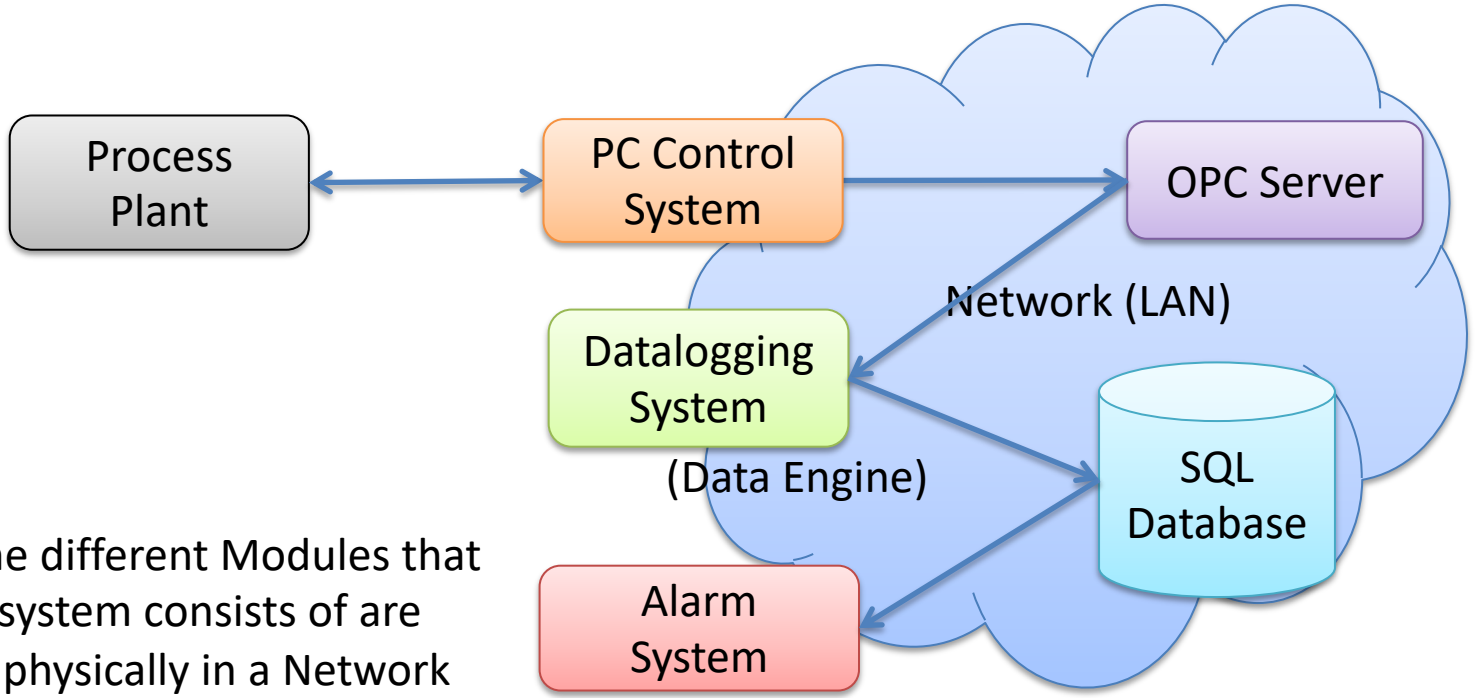
# Introduction to SCADA Systems

Supervisory Control and Data Acquisition

Hans-Petter Halvorsen

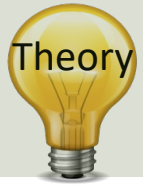
Table of Contents

# SCADA System Overview



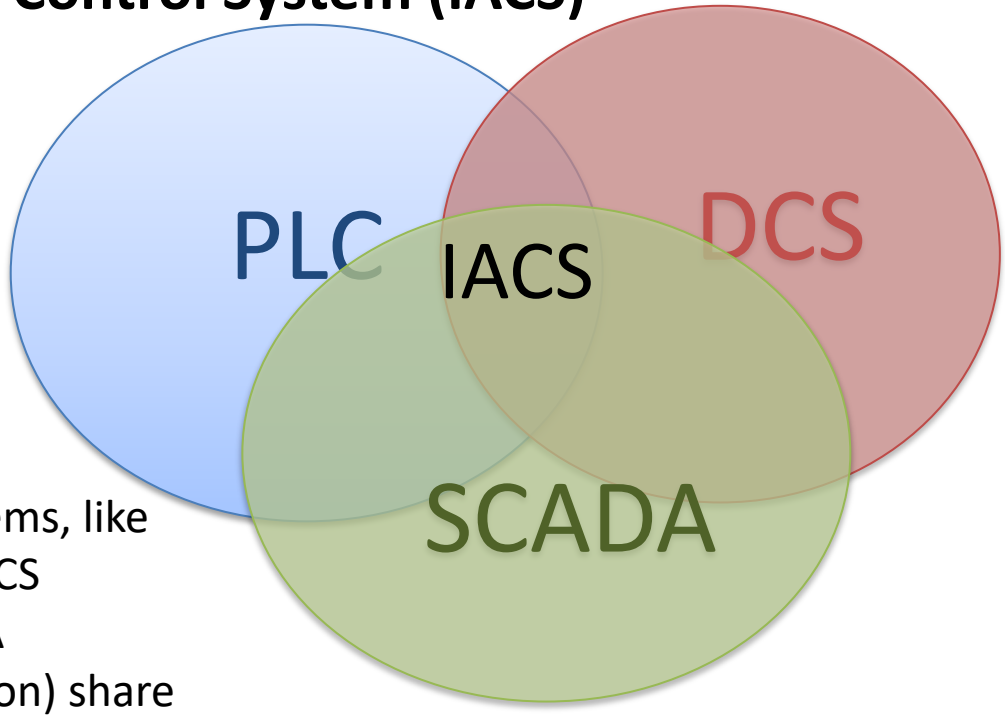

Typically, the different Modules that the SCADA system consists of are distributed physically in a Network (local network or over Internet)

# SCADA System




SCADA (Supervisory Control And Data Acquisition) is a type of **Industrial Automation and Control System (IACS)**

Industrial Automation and Control Systems (IACS) are computer-controlled systems that monitor and control industrial processes that exist in the physical world.




Industrial Automation and Control Systems, like PLC (Programmable Logic Controller), DCS (Distributed Control System) and SCADA (Supervisory Control And Data Acquisition) share many of the same features

# Industrial Control Systems (ICS)

Industrial Control Systems are computer-controlled systems that monitor and control industrial processes that exist in the physical world

Theory

cRIO

LabVIEW

I/O Module

Programmable Automation Controller (**PAC**)

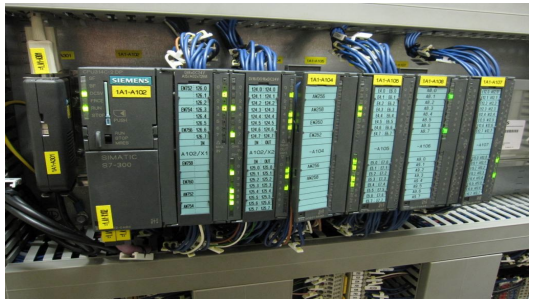
Industrial **PID** Controller

PC based Control System/**SCADA** System (Supervisory Control And Data Acquisition)

Distributed Control Systems (**DCS**)


DeltaV

**PLC** (Programmable Logic Controller)



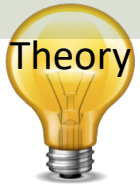
Siemens PLC

# PC-based Control System

## Industrial PID Controller


Embedded system with built-in PID algorithm, etc.

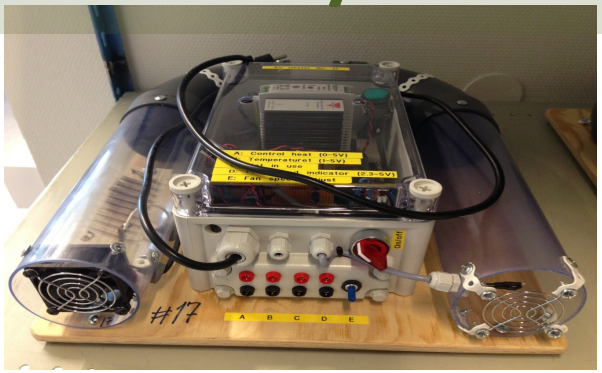
## PID Control using PC and I/O Module






I/O Module

# PC-based Control System Example


Process

Theory

0-5V/1-5V

0-5V

Analog In
Measurement(s)

Analog Out
Control Signal

I/O Module

USB-6008

AD Converter

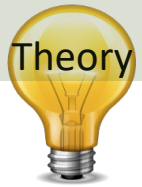
DA Converter

USB

Controller (PID) and
Lowpass Filter
Implementation

C#

# PC-based Control System




PC with Control Application


Control Signal

$u$

USB-6008 DAQ


$0-5V$

$u$

Air Heater Process


$1-5V$

$T_{out}$

$y$

Process Value

AI


USB-6008 DAQ

Digital Signal

Analog Measurement

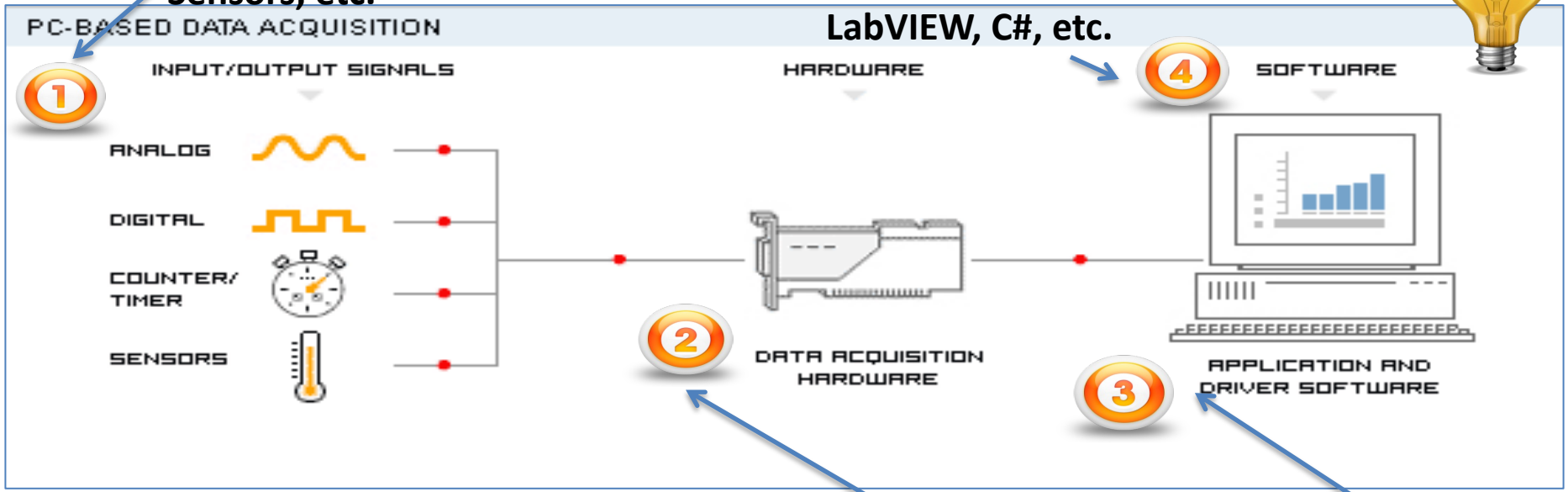
Temperature

# DAQ – Data Acquisition

Theory

Sensors, etc.

Your App created with LabVIEW, C#, etc.




A DAQ System consists of 4 parts:

1. Physical input/output signals
2. DAQ device/hardware
3. Driver software
4. Your software application (Application software)

**NI USB 6008 DAQ Device**

**NI DAQmx Driver or similiar**

# SCADA System

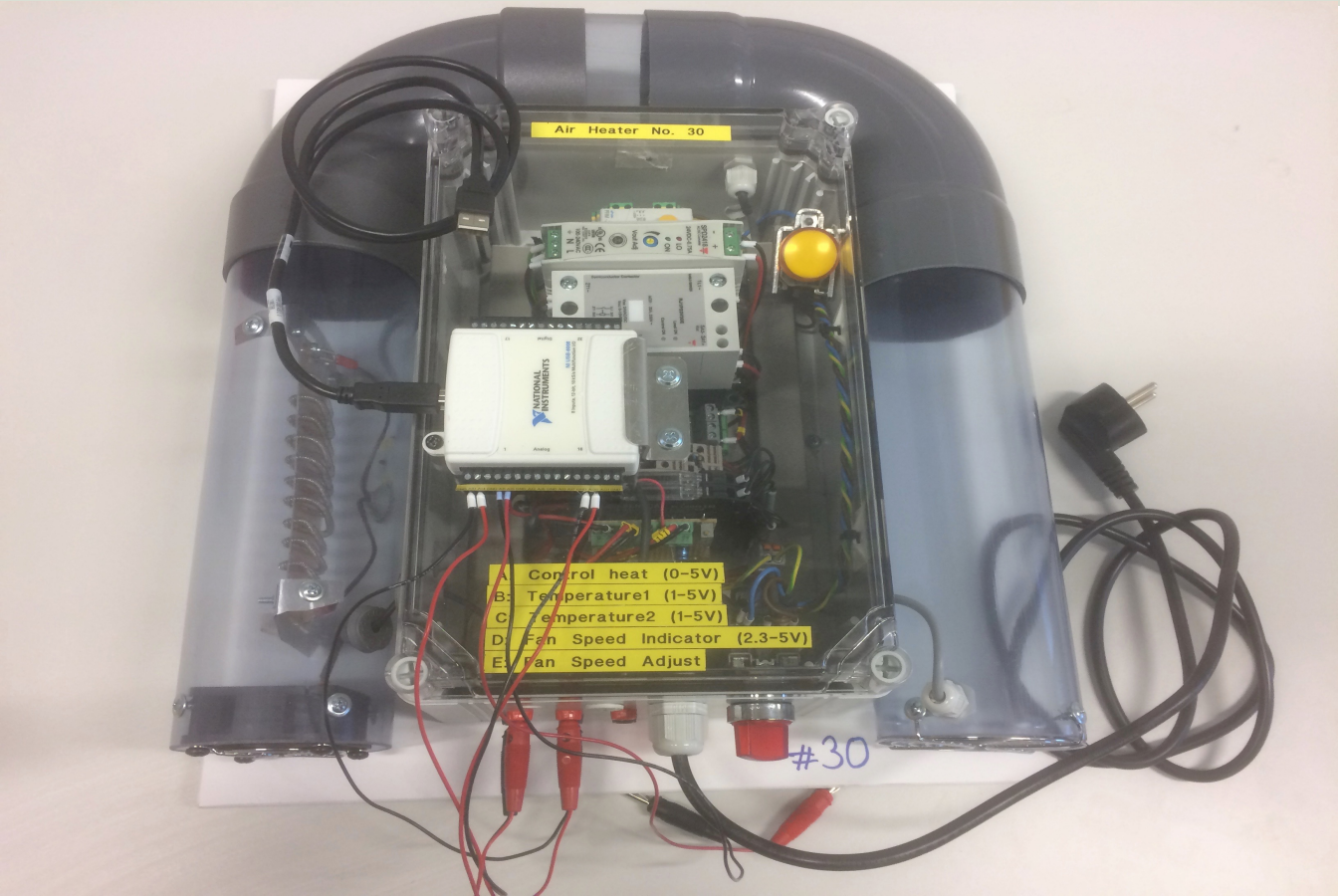
- The SCADA system typically contains different modules, such as:
  1. OPC Server
  2. A Database that stores all the necessary data
  3. **Control System**
  4. **Datalogging System**
  5. **Alarm System**
- Note! They are typically implemented as separate applications because they should be able to run on different computers in a network (distributed system).

# Air Heater

Hans-Petter Halvorsen

Table of Contents

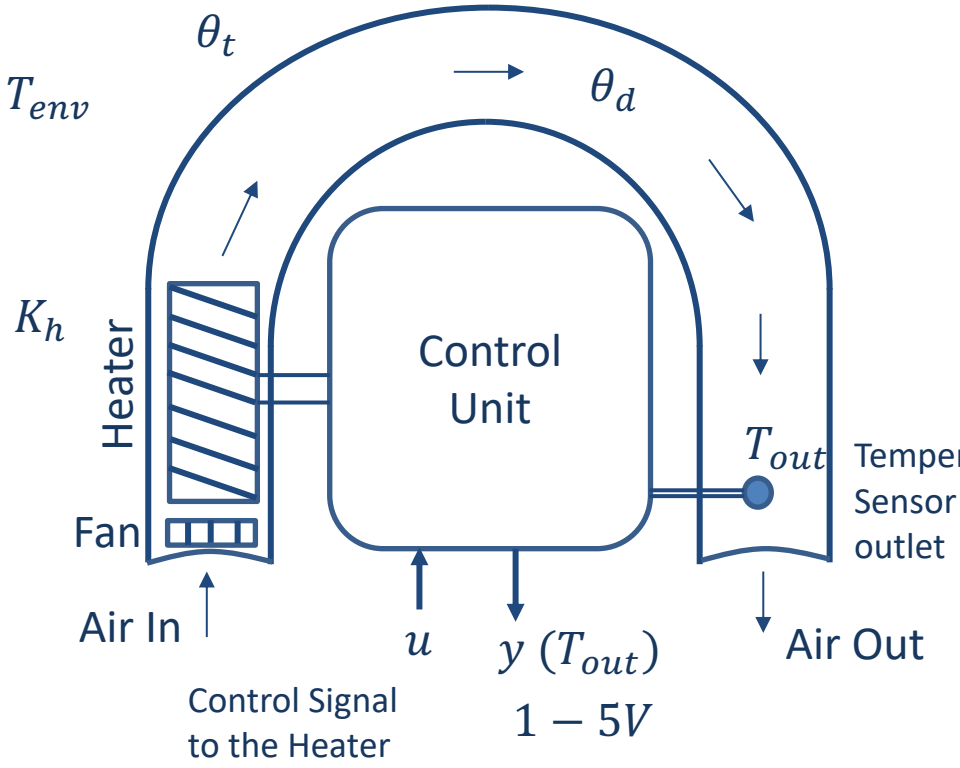
# Air Heater




The Air Heater is a small-scale laboratory process suitable for learning about control systems

The purpose is to control the Temperature on the outflow of the Air Heater tube.

# Air Heater




Aim:

Control the Temperature on the outlet ($T_{out}$)

# Air Heater Mathematical Model

$$\dot{T}_{out} = \frac{1}{\theta_t}\{-T_{out} + [K_h u(t-\theta_d) + T_{env}]\}$$

Where:

- $T_{out}$ is the air temperature at the tube outlet
- $u\ [V]$ is the control signal to the heater
- $\theta_t\ [s]$ is the time-constant
- $K_h\ [deg\ C\ /\ V]$ is the heater gain
- $\theta_d\ [s]$ is the time-delay representing air transportation and sluggishness in the heater
- $T_{env}$ is the environmental (room) temperature. It is the temperature in the outlet air of the air tube when the control signal to the heater has been set to zero for relatively long time (some minutes)

Example of Model Parameters:

Use, e.g., these values:

$$\theta_t = 22\ sec$$

$$\theta_d = 2\ sec$$

$$K_h = 3.5\ \frac{°C}{V}$$

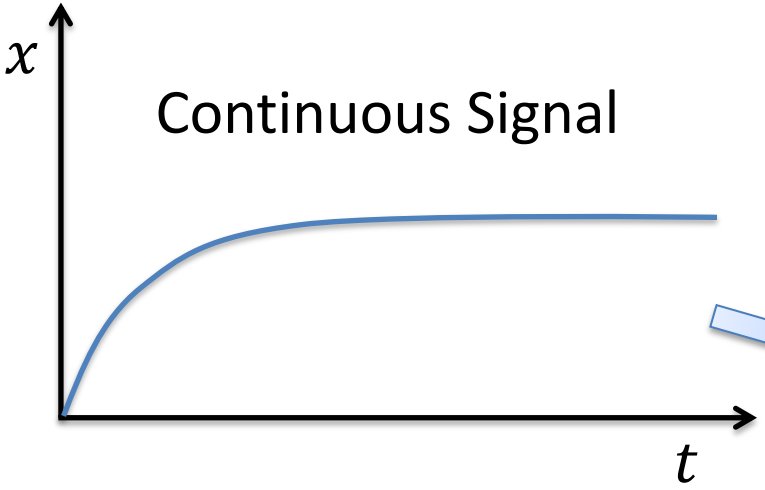
$$T_{env} = 21.5\ °C$$

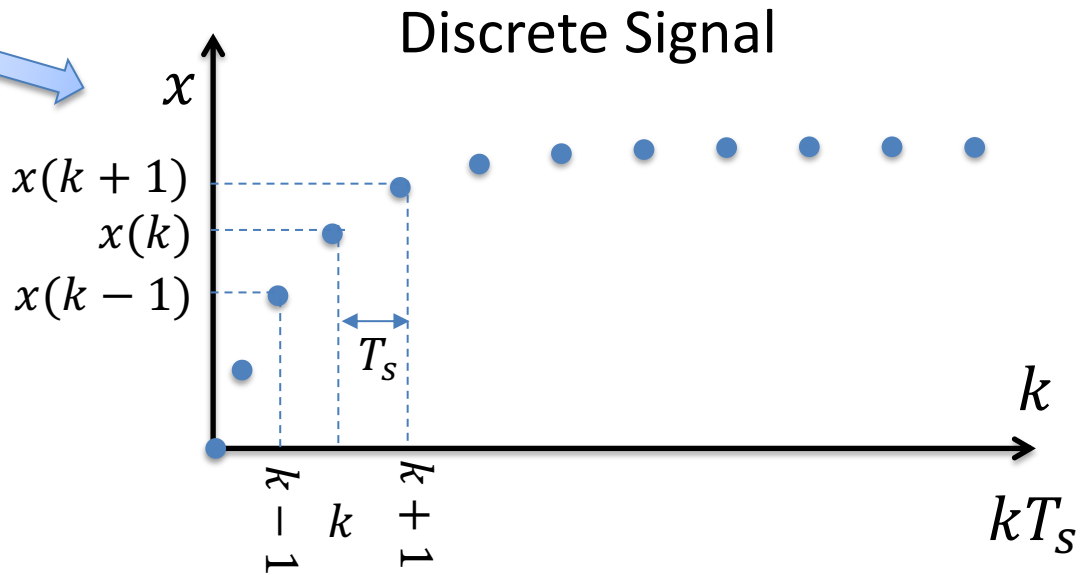
# Air Heater Implementation

- The Air Heater is a standard 1.order process with time delay.
- In LabVIEW or C#, you can make a discrete version of the model and implement it. In LabVIEW you can, e.g., use a Formula Node.
- Discrete version: $T_{out}(k+1) = \ \ldots$
- Tip! The Time delay part of the system can be a bit “tricky“ to implement. Start by discarding/remove the Time delay part and implement ant test it.
- Then later, try to implement and include the time delay part of the system. The time delay part can typically be implemented by creating an array/FIFO queue.

# Continuous vs. Discrete Systems

A computer can only deal with discrete signals




$T_s$ - Sampling Interval

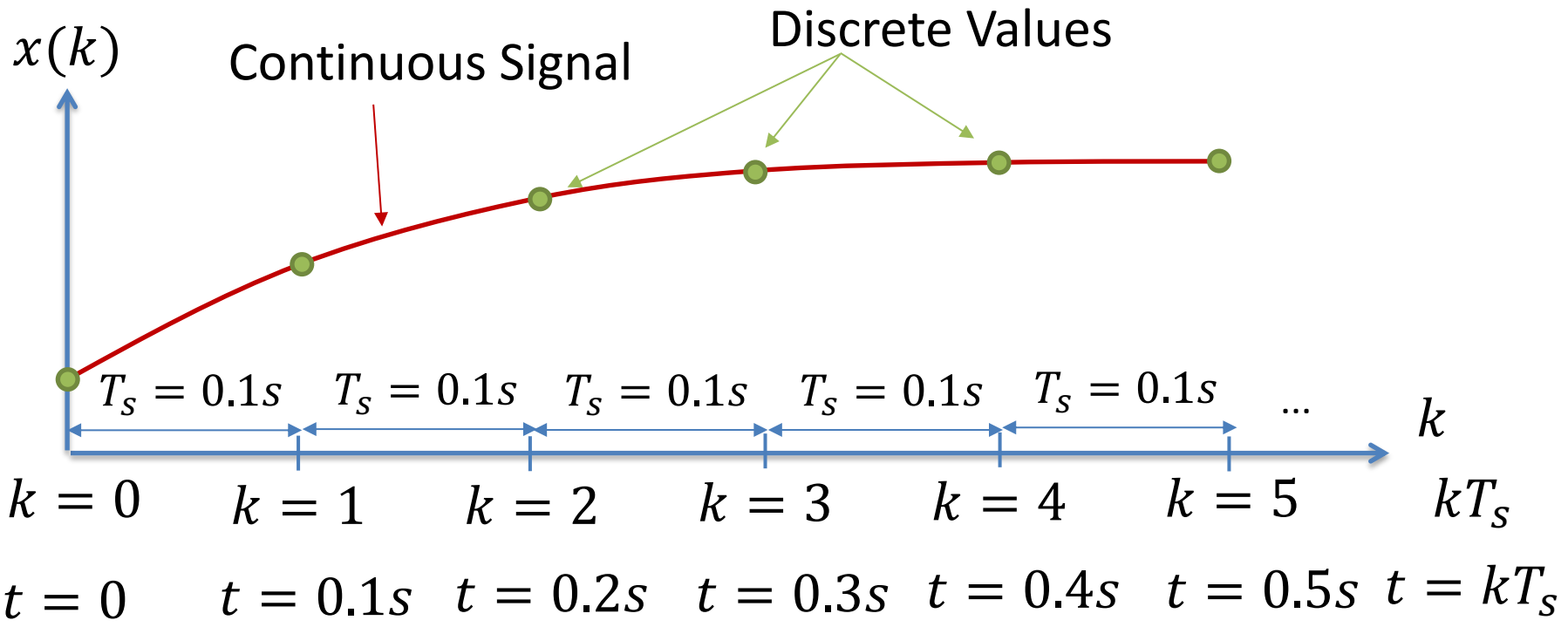
$x(k-1)$ - Previous Value

$x(k)$ - Current Value

$x(k-1)$ - Next Value

# Continuous vs. Discrete Systems

In this Example we have used Sampling Interval $T_s = 0.1s$



# Discretization

Continuous Model:

$$\dot{T}_{out} = \frac{1}{\theta_t}\{-T_{out} + [K_h u(t - \theta_d) + T_{env}]\}$$

We can use e.g., the Euler Approximation in order to find the discrete Model:

$$\dot{x} \approx \frac{x(k+1) - x(k)}{T_s}$$

$T_s$ - Sampling Time

$x(k)$ - Present value

$x(k+1)$ - Next (future) value

The discrete Model will then be on the form:

$$x(k+1) = x(k) + \ \ldots$$

We can then implement the discrete model in C#

# Finding Model Parameters using “Trial and Error”

You may use, e.g., the following Parameters as a starting point, but since every Air Heater is unique, you may want to adjust these parameters. The “Trial and Error Method” may be an easy way to find the Parameters for your Process.

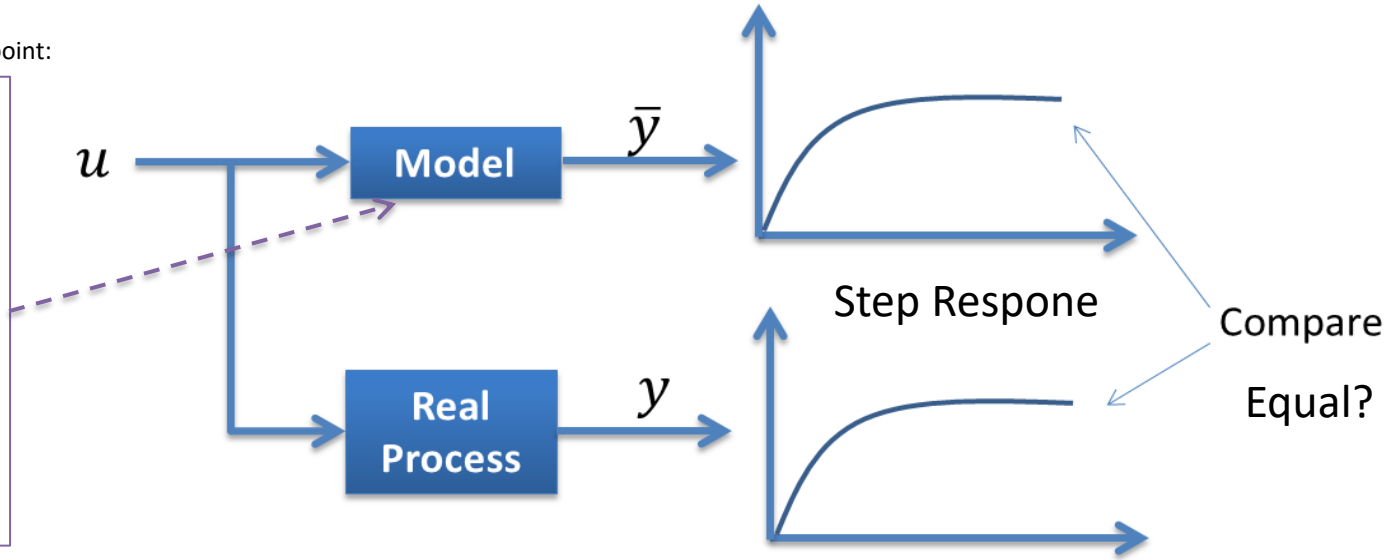
These values can be a good starting point:

$$\theta_t = 22\ sec$$

$$\theta_d = 2\ sec$$

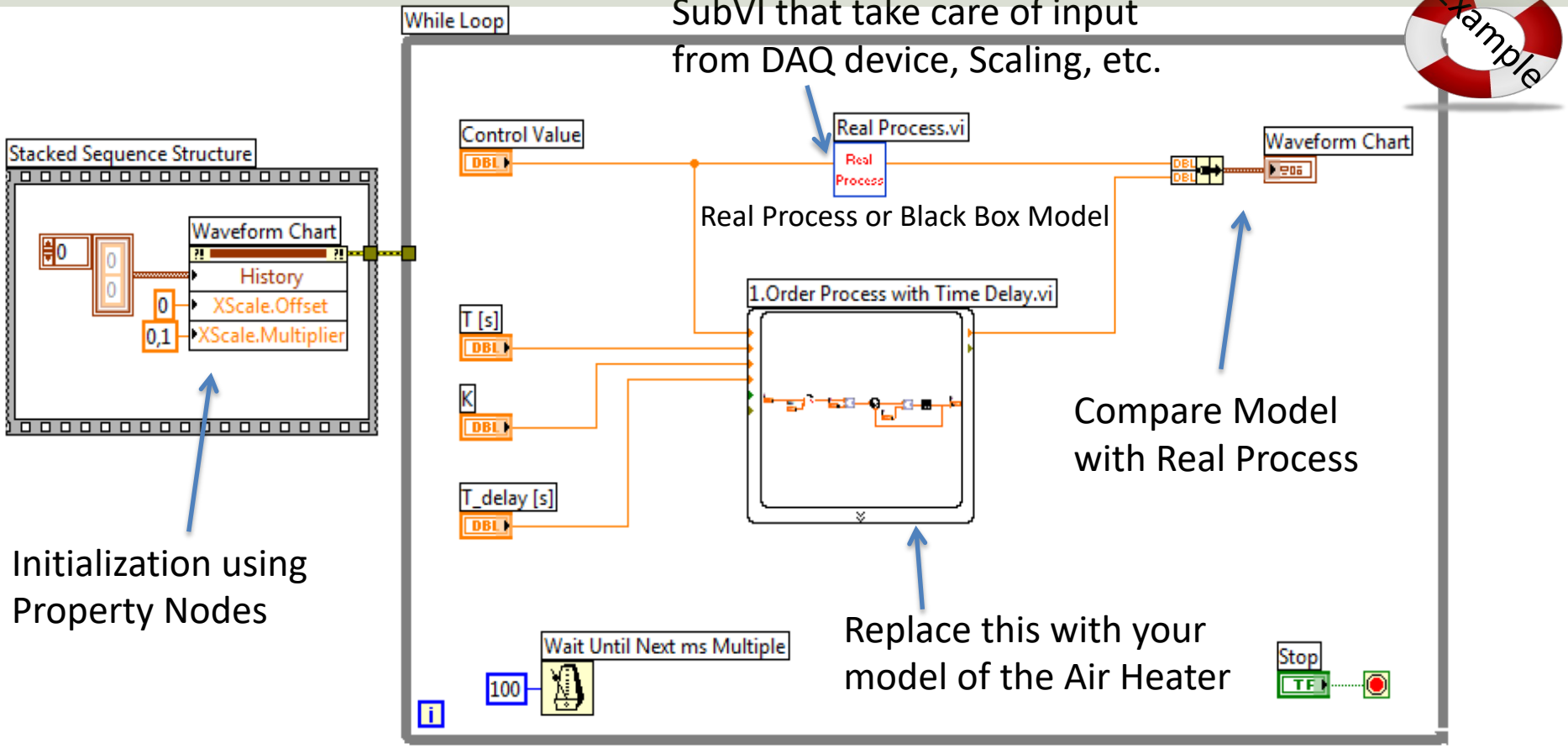
$$K_h = 3.5\ \frac{°C}{V}$$

$$T_{env} = 21.5\ °C$$




Procedure: You run the Model and the Real Process in Parallel. Adjust the Model Parameters until the output of the Model and the Real Process is “equal”.

# “Trial and Error” Example in LabVIEW



# Control System

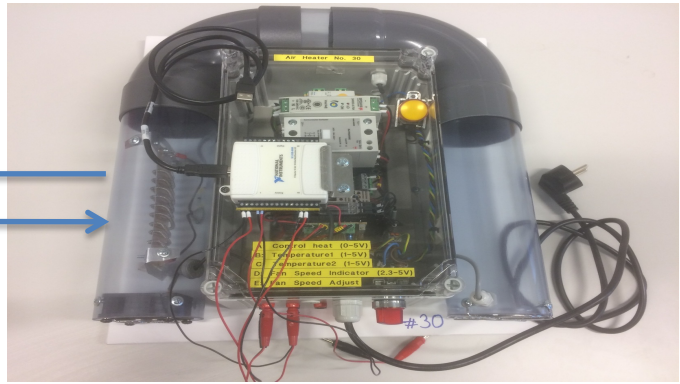
Hans-Petter Halvorsen

Table of Contents

# Control System

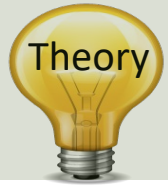
Computer


I/O Module

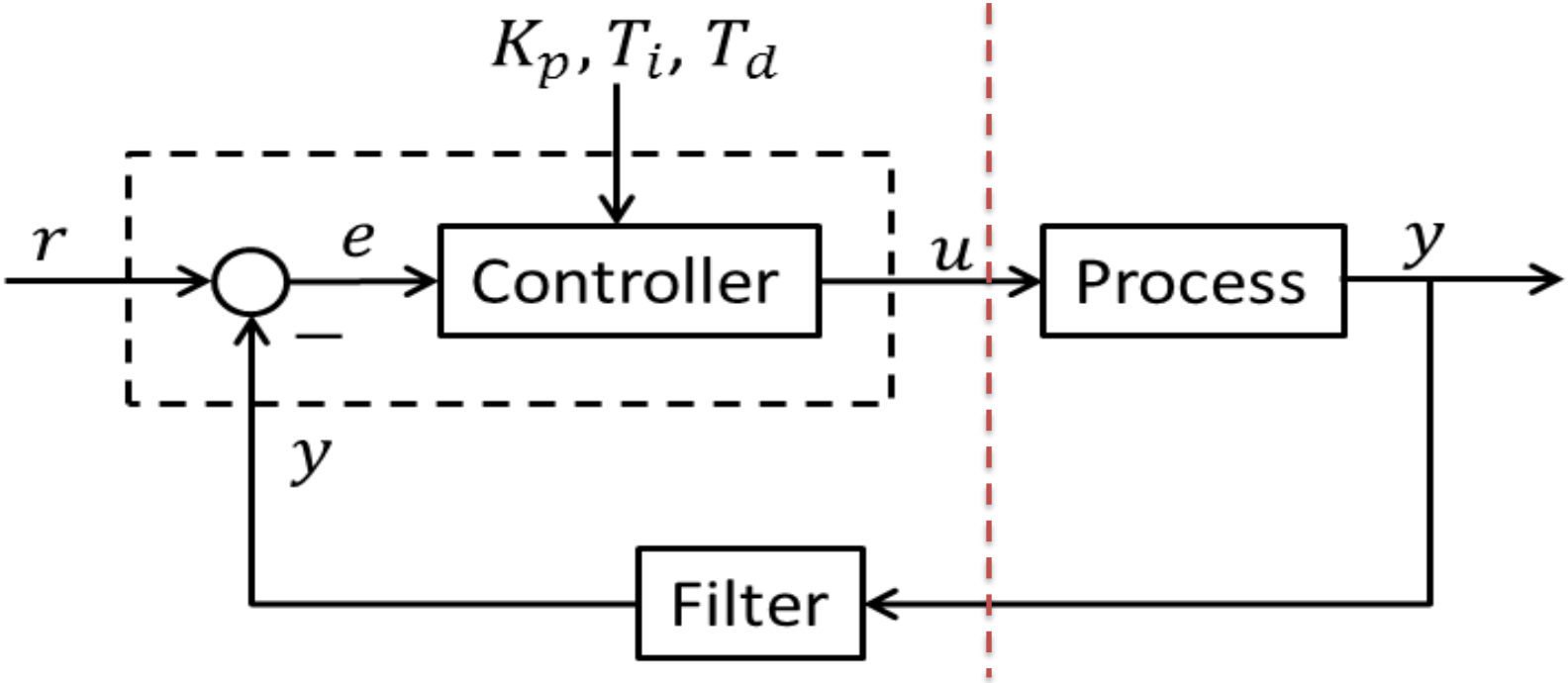



Air Heater

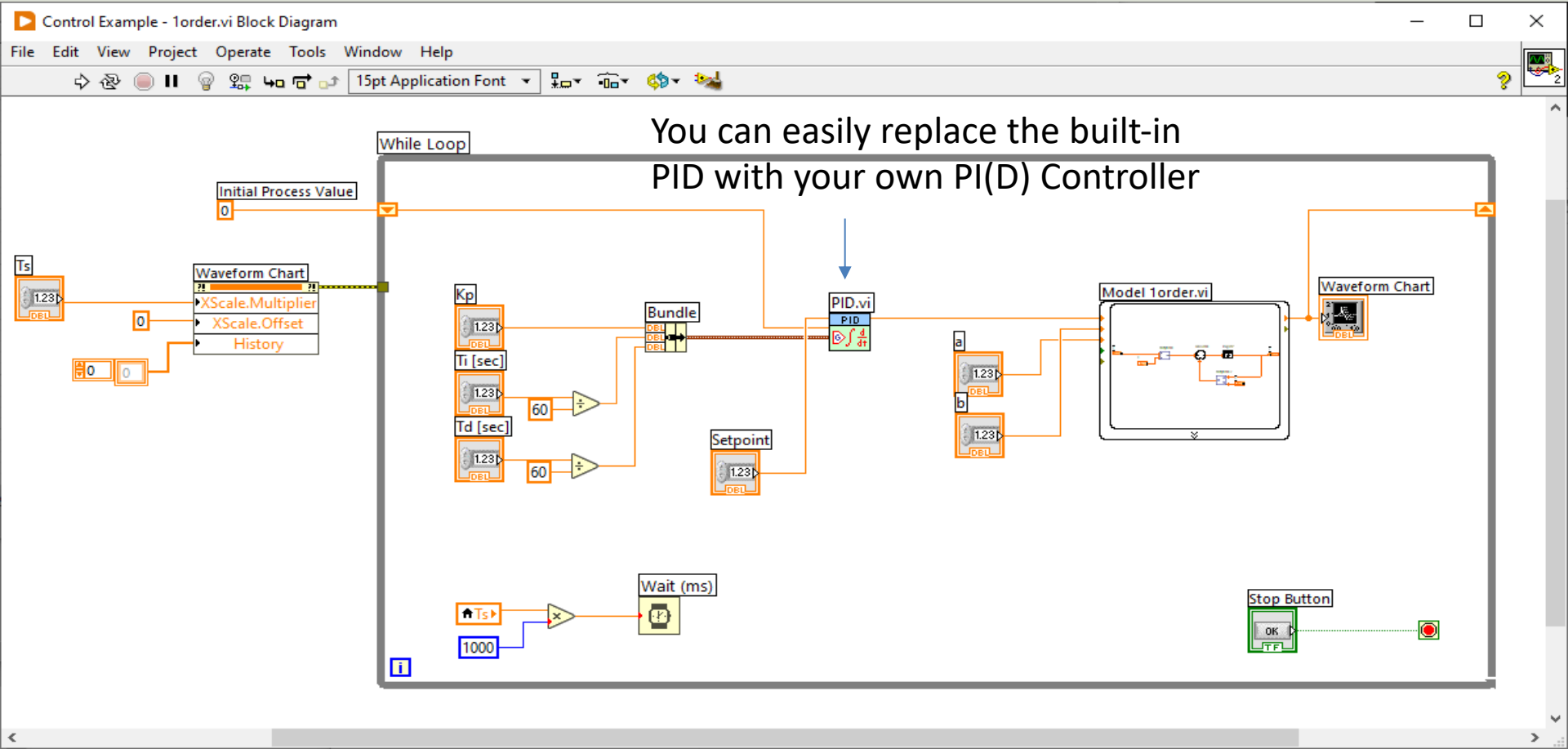
# Control System Example




While the real process is continuous, normally the Controller and the Filter is implemented in a computer.



# LabVIEW Control System



# Control System in C#

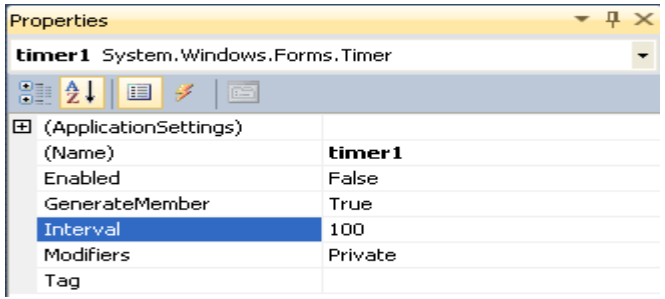
1  Timer

Select the “Timer” component in the Toolbox

2 **Initialization:**

```
public Form1()
        {
            InitializeComponent();
            timer1.Start();
        }
```

3 **Properties:**



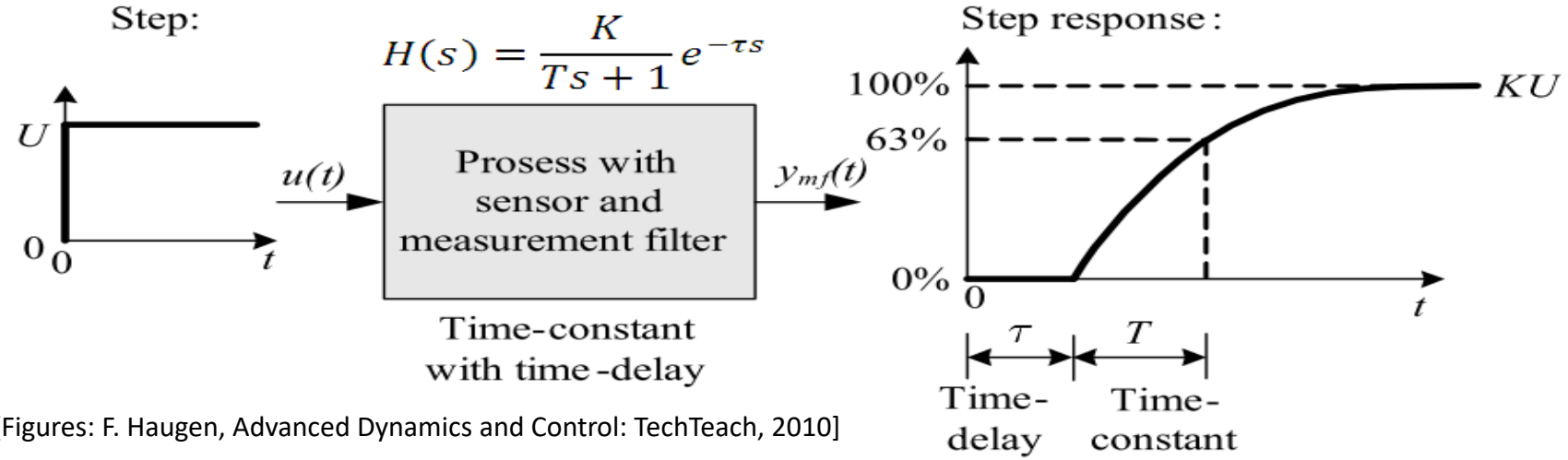

You may specify the Timer Interval in the Properties Window

4 **Timer Event:**

```
private void timer1_Tick(object sender, EventArgs e)
        {
          … //DAQ
          … //Scaling
          … //Control
          … //Plot Data
          … //Write to OPC
        }
```

In Visual Studio you may want to use a Timer instead of a While Loop in order to read values at specific intervals.

# PID Tuning with Skogestad




[Figures: F. Haugen, Advanced Dynamics and Control: TechTeach, 2010]

We can set, e.g., Tc=10 sec. and c=1.5.
You may use other values if these values give a poor result.

| Process type | $H_{psf}(s)$ (process) | $K_p$ | $T_i$ | $T_d$ |
|---|---|---|---|---|
| Integrator + delay | $\frac{K}{s}e^{-\tau s}$ | $\frac{1}{K(T_C+\tau)}$ | $c\,(T_C+\tau)$ | 0 |
| Time-constant + delay | $\frac{K}{Ts+1}e^{-\tau s}$ | $\frac{T}{K(T_C+\tau)}$ | $\min\,[T,\, c\,(T_C+\tau)]$ | 0 |
| Integr + time-const + del. | $\frac{K}{(Ts+1)s}e^{-\tau s}$ | $\frac{1}{K(T_C+\tau)}$ | $c\,(T_C+\tau)$ | $T$ |
| Two time-const + delay | $\frac{K}{(T_1 s+1)(T_2 s+1)}e^{-\tau s}$ | $\frac{T_1}{K(T_C+\tau)}$ | $\min\,[T_1,\, c\,(T_C+\tau)]$ | $T_2$ |
| Double integrator + delay | $\frac{K}{s^2}e^{-\tau s}$ | $\frac{1}{4K(T_C+\tau)^2}$ | $4\,(T_C+\tau)$ | $4\,(T_C+\tau)$ |

Table 1: Skogestad's formulas for PI(D) tuning.

# Discrete PI(D) Controller

Hans-Petter Halvorsen

Table of Contents

# Discrete PI Controller Example

Continuous PI Controller:

$$u(t) = u_0 + K_p e(t) + \frac{K_p}{T_i}\int_0^t e d\tau$$

$$\dot{u} = \dot{u}_0 + K_p \dot{e} + \frac{K_p}{T_i} e$$

We use the Euler Backward method:

$$\dot{x} = \frac{x_k - x_{k-1}}{T_s}$$

$$\frac{u_k - u_{k-1}}{T_s} = \frac{u_{0,k} - u_{0,k-1}}{T_s} + K_p \frac{e_k - e_{k-1}}{T_s} + \frac{K_p}{T_i} e_k$$

$$u_k = u_{k-1} + u_{0,k} - u_{0,k-1} + K_p(e_k - e_{k-1}) + \frac{K_p}{T_i} T_s e_k$$

We may set:

$$\Delta u_k = u_k - u_{k-1}$$

This gives the following discrete PI algorithm:

$$e_k = r_k - y_k$$

$$\Delta u_k = u_{0,k} - u_{0,k-1} + K_p(e_k - e_{k-1}) + \frac{K_p}{T_i} T_s e_k$$

$$u_k = u_{k-1} + \Delta u_k$$

This algorithm can be easly implemented in a Programming language

# Discrete PI Controller Example


```
class PidController
{
        public double r;
        public double Kp;
        public double Ti;
        public double Ts;

        public double PiController(double y)
        {
            double e;
            double u;

            e = r - y;
            u = ...;
            return u;
        }
}
```

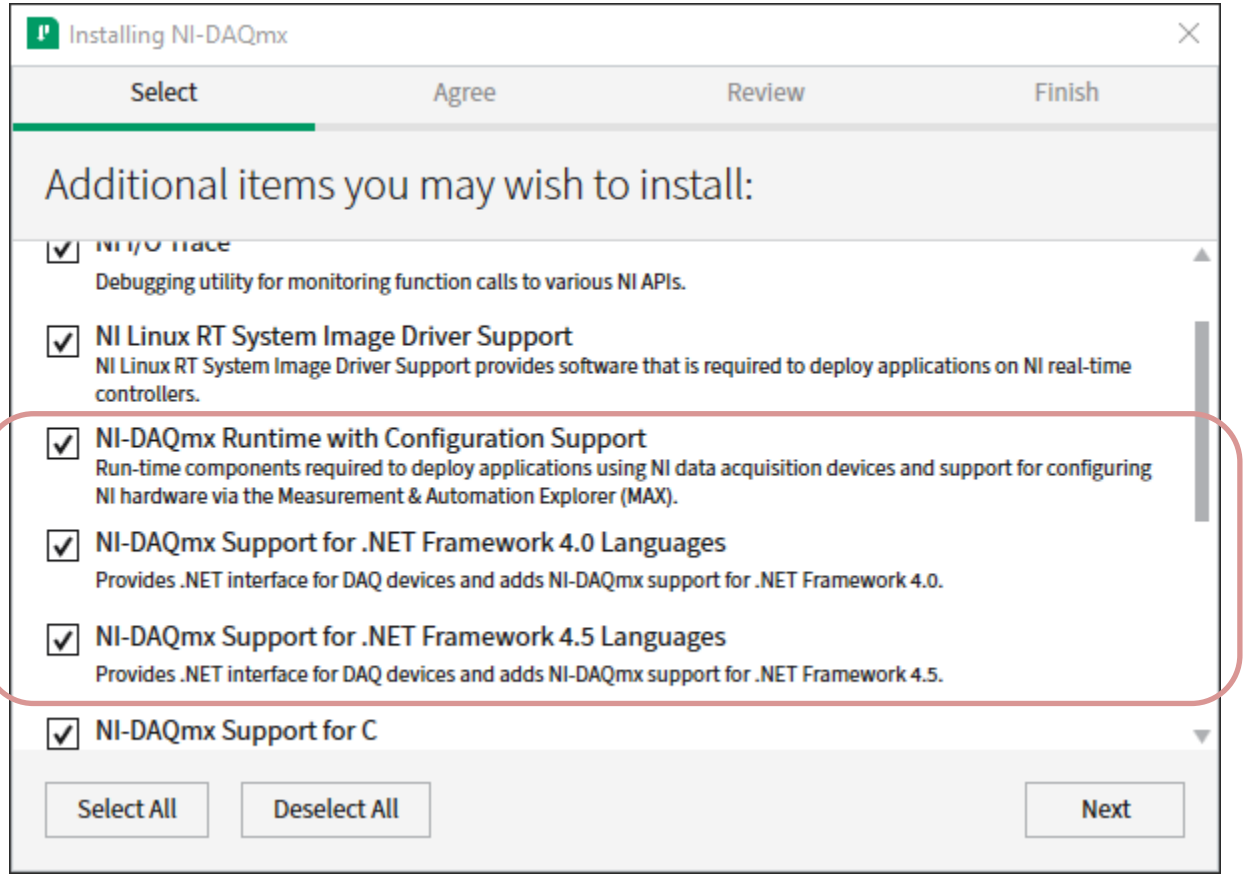
Note! This is just a simple Example

# DAQ in C#

Hans-Petter Halvorsen

Table of Contents

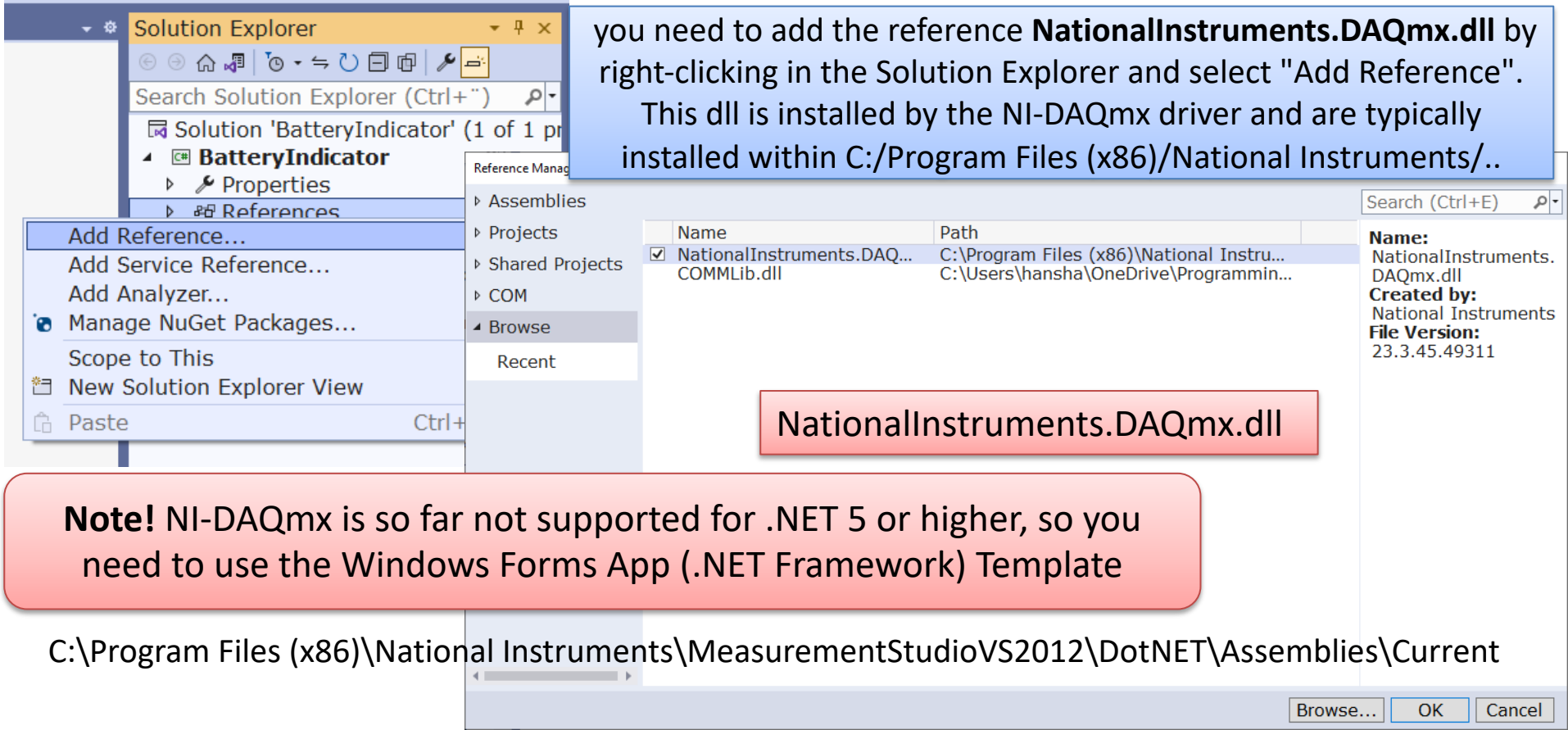
# NI-DAQmx Installation




Make sure to add support for Visual Studio/.NET during installation of the NI-DAQmx software

# Add Reference

you need to add the reference **NationalInstruments.DAQmx.dll** by right-clicking in the Solution Explorer and select "Add Reference". This dll is installed by the NI-DAQmx driver and are typically installed within C:/Program Files (x86)/National Instruments/..




**Note!** NI-DAQmx is so far not supported for .NET 5 or higher, so you need to use the Windows Forms App (.NET Framework) Template

C:\Program Files (x86)\National Instruments\MeasurementStudioVS2012\DotNET\Assemblies\Current

# Simple DAQ in C# with DAQmx

```
double ReadDaqData()
{
    Task analogInTask = new Task();

    AIChannel myAIChannel;

    myAIChannel = analogInTask.AIChannels.CreateVoltageChannel(
                "dev1/ai0",
                "myAIChannel",
                AITerminalConfiguration.Differential,
                0,
                5,
                AIVoltageUnits.Volts
                );

     AnalogSingleChannelReader reader = new
             AnalogSingleChannelReader(analogInTask.Stream);

     double analogDataIn = reader.ReadSingleSample();
     return analogDataIn;
}
```

**Analog In** Example

Note! The physical wiring on the DAQ device can either be “Differential” or “RSE”. Make sure your code and the wiring uses the same configuration.

# Simple DAQ in C# with DAQmx

**Analog Out** Example

```
WriteDaqData(double analogDataOut)
{
      Task analogOutTask = new Task();

      AOChannel myAOChannel;

      myAOChannel = analogOutTask.AOChannels.CreateVoltageChannel(
                  "dev1/ao0",
                  "myAOChannel",
                  0,
                  5,
                  AOVoltageUnits.Volts
                  );

      AnalogSingleChannelWriter writer = new
              AnalogSingleChannelWriter(analogOutTask.Stream);

      writer.WriteSingleSample(true, analogDataOut);
}
```

# Discrete Lowpass Filter

Hans-Petter Halvorsen

Table of Contents

# Discrete Lowpass Filter Example

Lowpass Filter Transfer function:

$$H(s) = \frac{y(s)}{u(s)} = \frac{1}{T_f s + 1}$$

Inverse Laplace the differential Equation:

$$T_f \dot{y} + y = u$$

We use the Euler Backward method:

$$\dot{x} = \frac{x_k - x_{k-1}}{T_s}$$

This gives:

$$T_f \frac{y_k - y_{k-1}}{T_s} + y_k = u_k$$

$$y_k = \frac{T_f}{T_f + T_s} y_{k-1} + \frac{T_s}{T_f + T_s} u_k$$

We define:

$$\frac{T_s}{T_f + T_s} \equiv a$$

This gives:

$$y_k = (1 - a) y_{k-1} + a u_k$$

Filter output

Noisy input signal

$$T_s \leq \frac{T_f}{5}$$

This algorithm can be easly implemented in a Programming language

# Discrete Lowpass Filter Example

```
class Filter
{
        public double yk;
        public double Ts;
        public double Tf;

        public double LowPassFilter(double yFromDaq)
        {
            double a;
            double yFiltered;

            a = Ts / (Ts + Tf);
            yFiltered = (1 - a) * yk + a * yFromDaq;
            yk = yFiltered;
            return yFiltered;
        }
}
```

Note! This is just a simple Example

# Database Design

Hans-Petter Halvorsen

Table of Contents

# Database Examples

**Tag Configuration:**

The TAG table(s) could e.g., have the following columns:

- TagId (int, Primary Key, Identity)
- TagName (varchar)
- ItemId (varchar) (OPC)
- ItemUrl (varchar) (OPC)
- Description (varchar)
- etc.

**Alarm Configuration & Alarm Data:**

Important fields in an alarm handling system could be:

- AlarmId
- Activation Time
- Acknowledge Time and Person
- Type of Alarm
- Which Tag
- Alarm Limits
- Textual Description
- etc.

Here are some examples of functionality of the SCADA system and information that typically could be stored in the Database.

**Tag Data:**

Create one or more tables used for logging the Tag Values into the Database. Example of information:

- Value
- Timestamp
- Status (e.g., “Active”, “Not Active”)
- Quality (e.g., “Good”, “Bad”)
- etc.

# Database Examples

The alarm system will be responsible for the warnings and the alarms in a monitoring and control system. An alarm system contains different **Alarms** and **Warnings** like:

- Timeout; no input from a sensor or another computer system within a specific amount of time,
- High High (HH) or Low Low (LL) alarm; a critical alarm condition,
- High (H) or Low alarm (L)
- I/O device errors
- System device errors
- etc.

Make sure your Alarm tables and system can handle some of these kinds of alarms and warnings.

An Alarm System use different Alarm Devices, such as, e.g.,

- Screen; display the alarms
- Keyboard; alarm operations
- Horn; indicate an active alarm, or security alarm
- Lamp; indicate an active alarm, or an active alarm by blinking and an acknowledge alarm by a steady light
- Printer; logging of the alarm states
- SMS
- E-mail
- Etc.

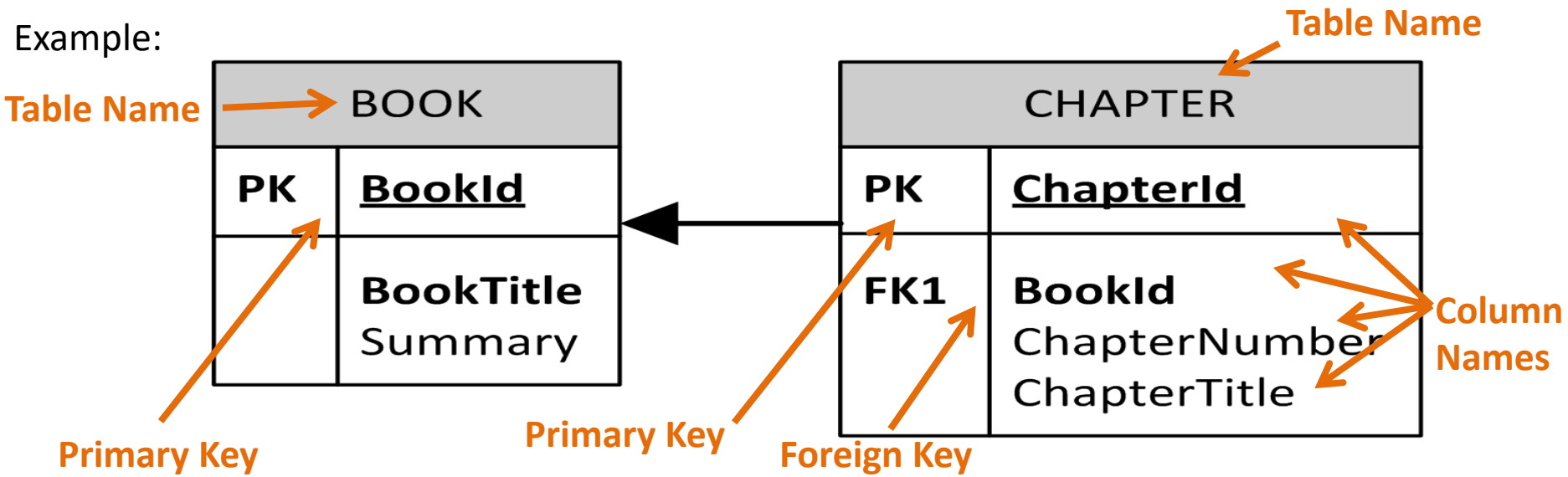
Make use of one or more of these alarm devices in your Alarm Handling and Management System.

# Database Design – ER Diagram

ER Diagram (Entity-Relationship Diagram)

- Used for Design and Modeling of Databases.
- Specify Tables and **relationship** between them (**Primary Keys** and **Foreign Keys**)

Example:



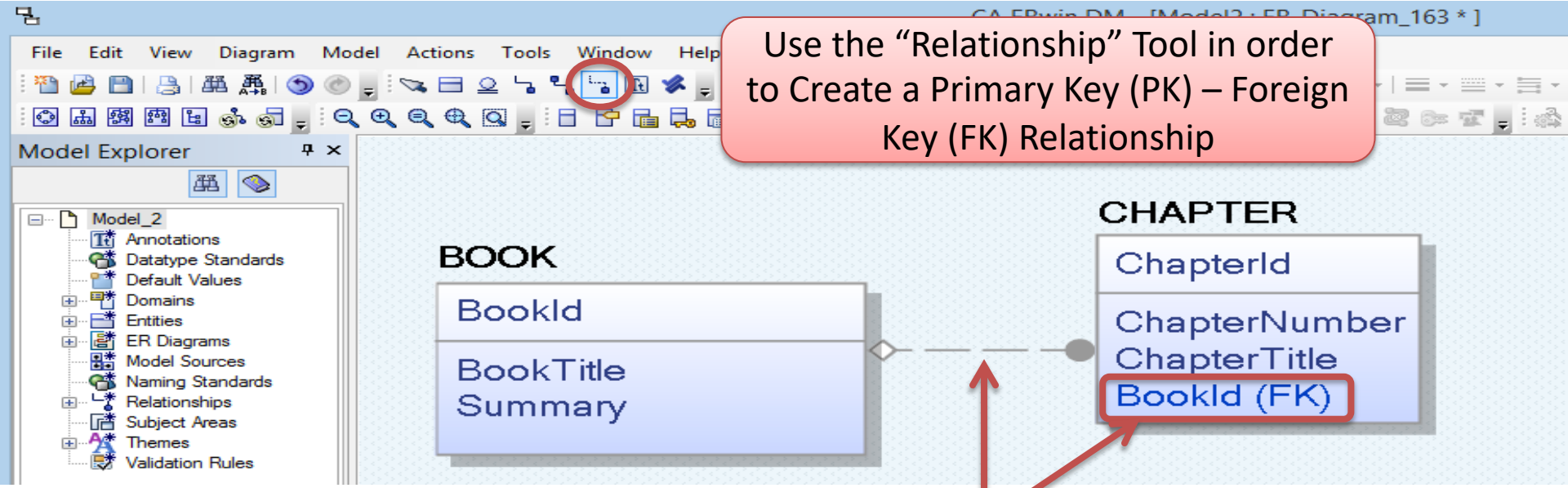

Relational Database. In a relational database all the tables have one or more relation with each other using Primary Keys (PK) and Foreign Keys (FK). Note! You can only have one PK in a table, but you may have several FK's.

# Database - “Best Practice”

- **Tables**: Use upper case and singular form in table names – not plural, e.g., “STUDENT” (not “students”)
- **Columns**: Use Pascal notation, e.g., “StudentId”
- **Primary Key**:
  - If the table name is “COURSE”, name the Primary Key column “CourseId”, etc.
  - “Always” use Integer and Identity(1,1) for Primary Keys. Use UNIQUE constraint for other columns that needs to be unique, e.g. “RoomNumber”
- Specify **Required** Columns (NOT NULL) – i.e., which columns that need to have data or not
- Standardize on few/these **Data Types**: *int*, *float*, *varchar(x), datetime*, *bit*
- Use English for table and column names
- Avoid abbreviations! (Use “RoomNumber” – not “RoomNo”, “RoomNr”, ...)

# Introduction to ERwin

## How-To: Create Primary Key (PK) – Foreign Key (FK) Relationships:



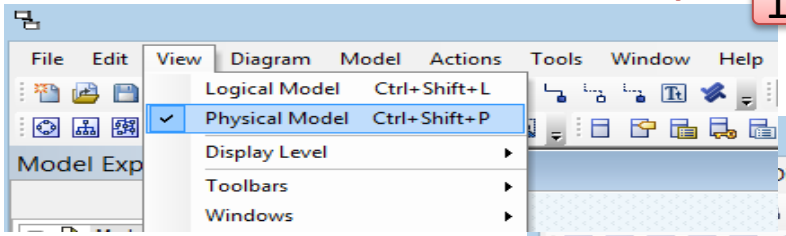

Click first on the PK table and then on the FK table using the "Relationship" Tool. The Relationship Connection and Foreign Key column are then Created Automatically

# Introduction to ERwin

## How-To: Create a SQL Script

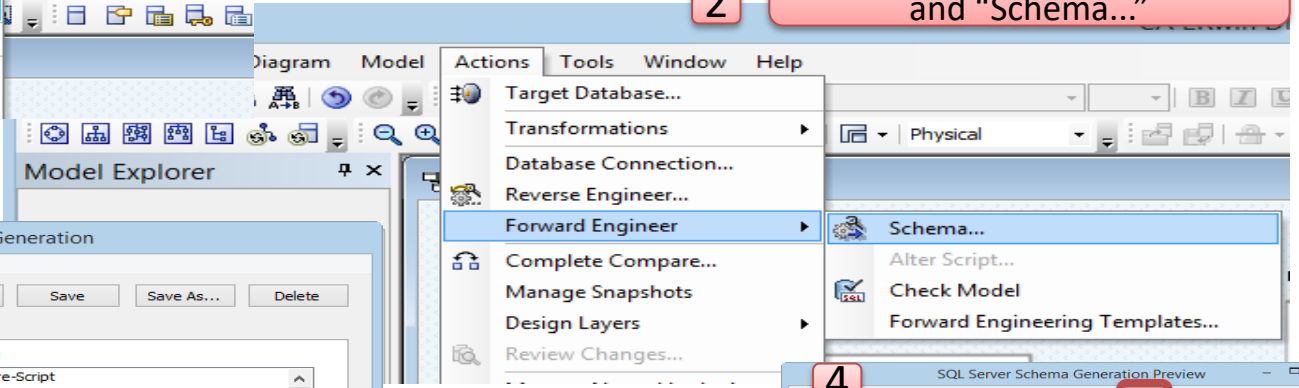
1

Make sure you are using the Physical Model

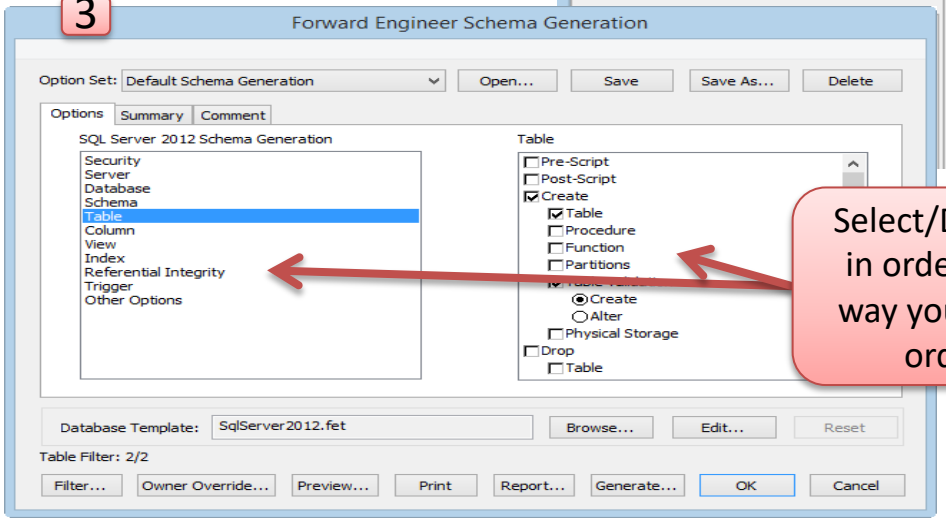



2

Select “Forward Engineering” and “Schema...”

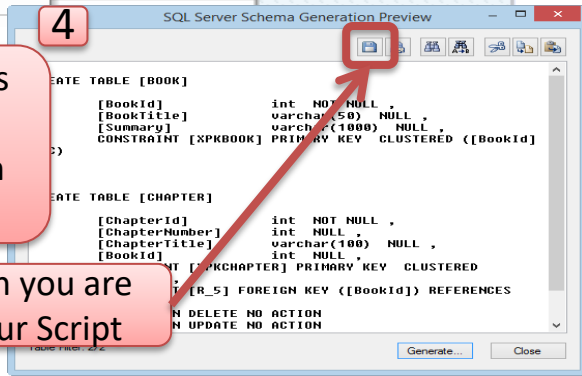



3




Select/Deselect different Options in order to make your script the way you want. Click “Preview” in order to see the results.

4




Click “Save” when you are satisfied with your Script

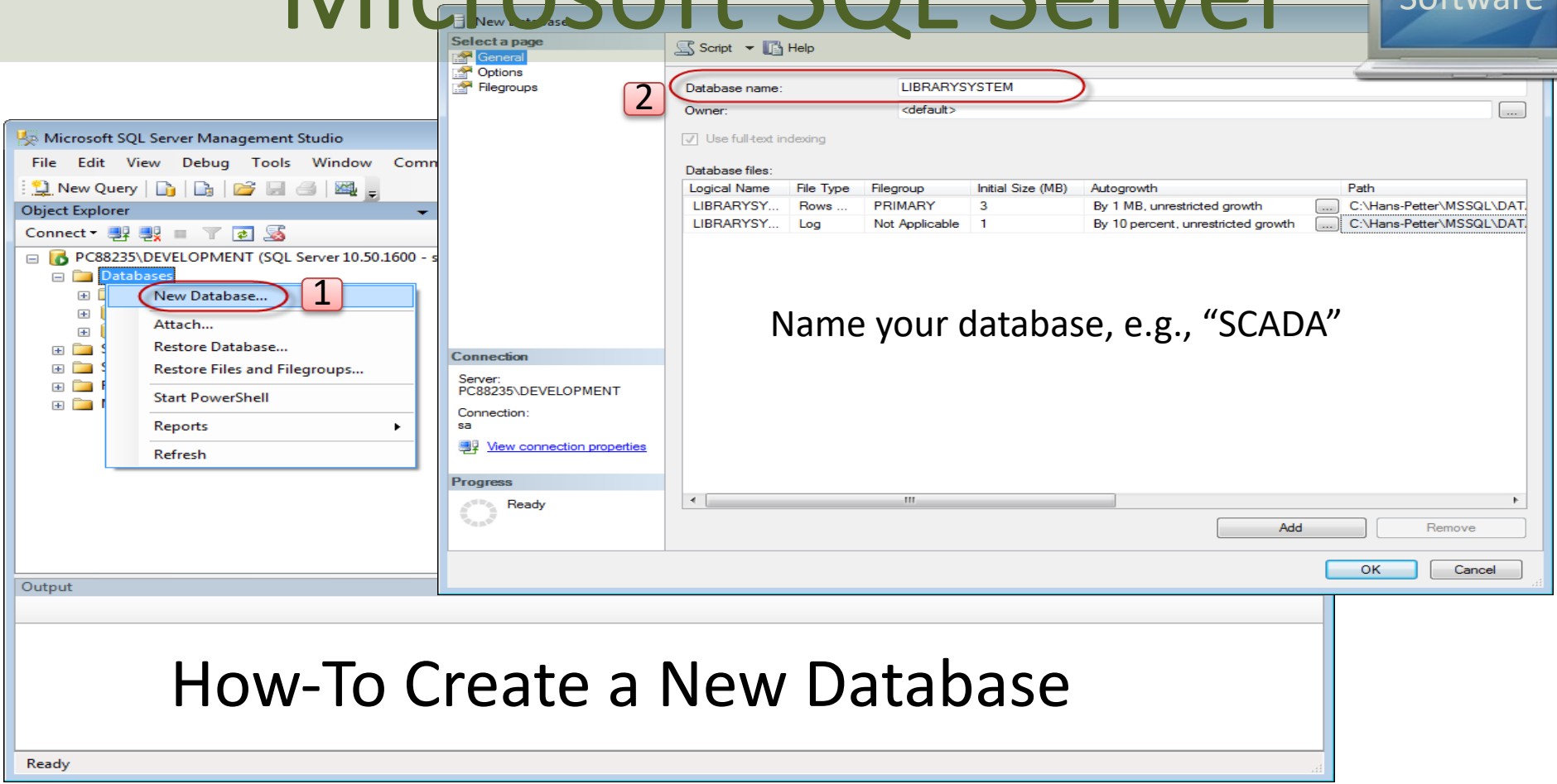
https://www.halvorsen.blog

# Database Implementation with SQL Server

Hans-Petter Halvorsen

Table of Contents

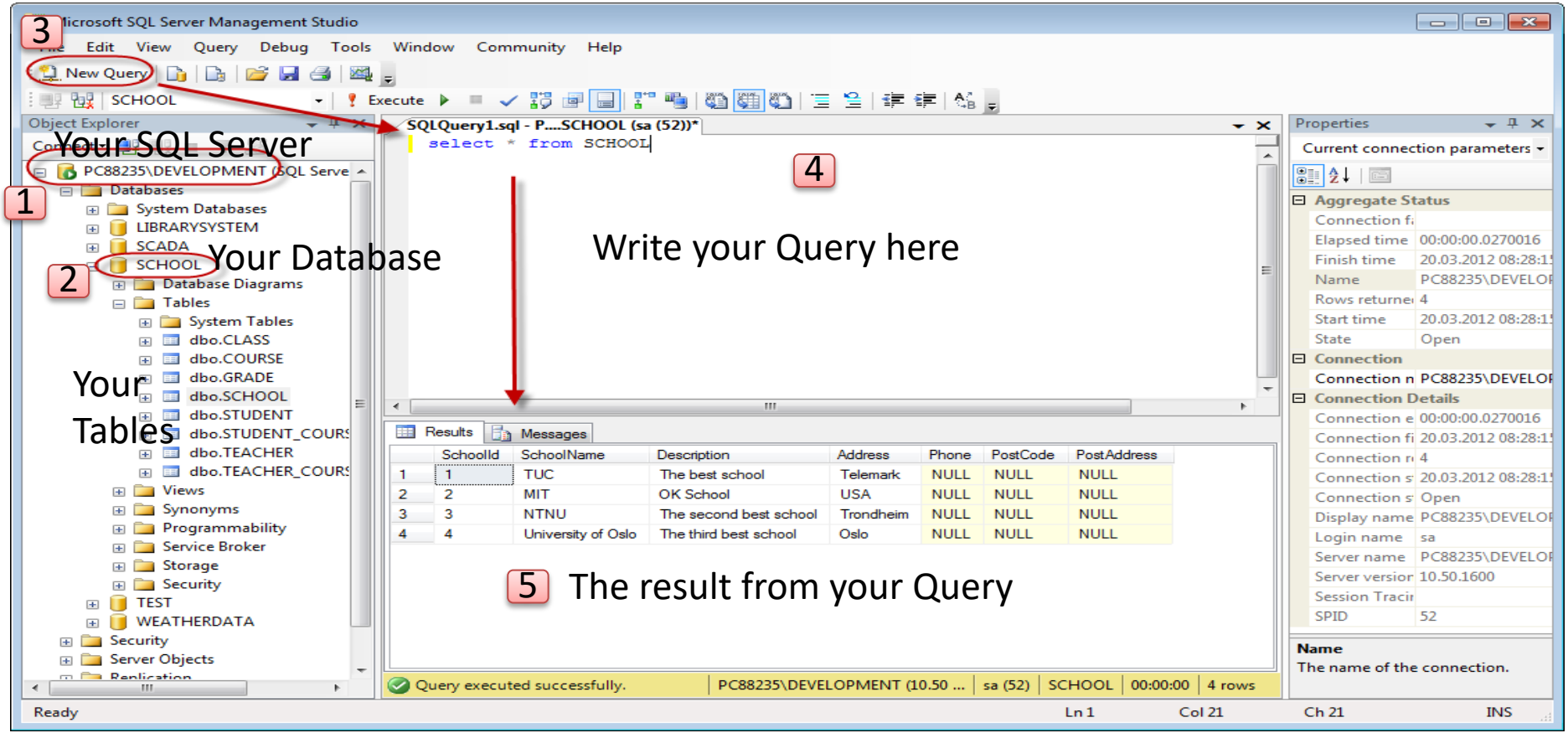
# Microsoft SQL Server

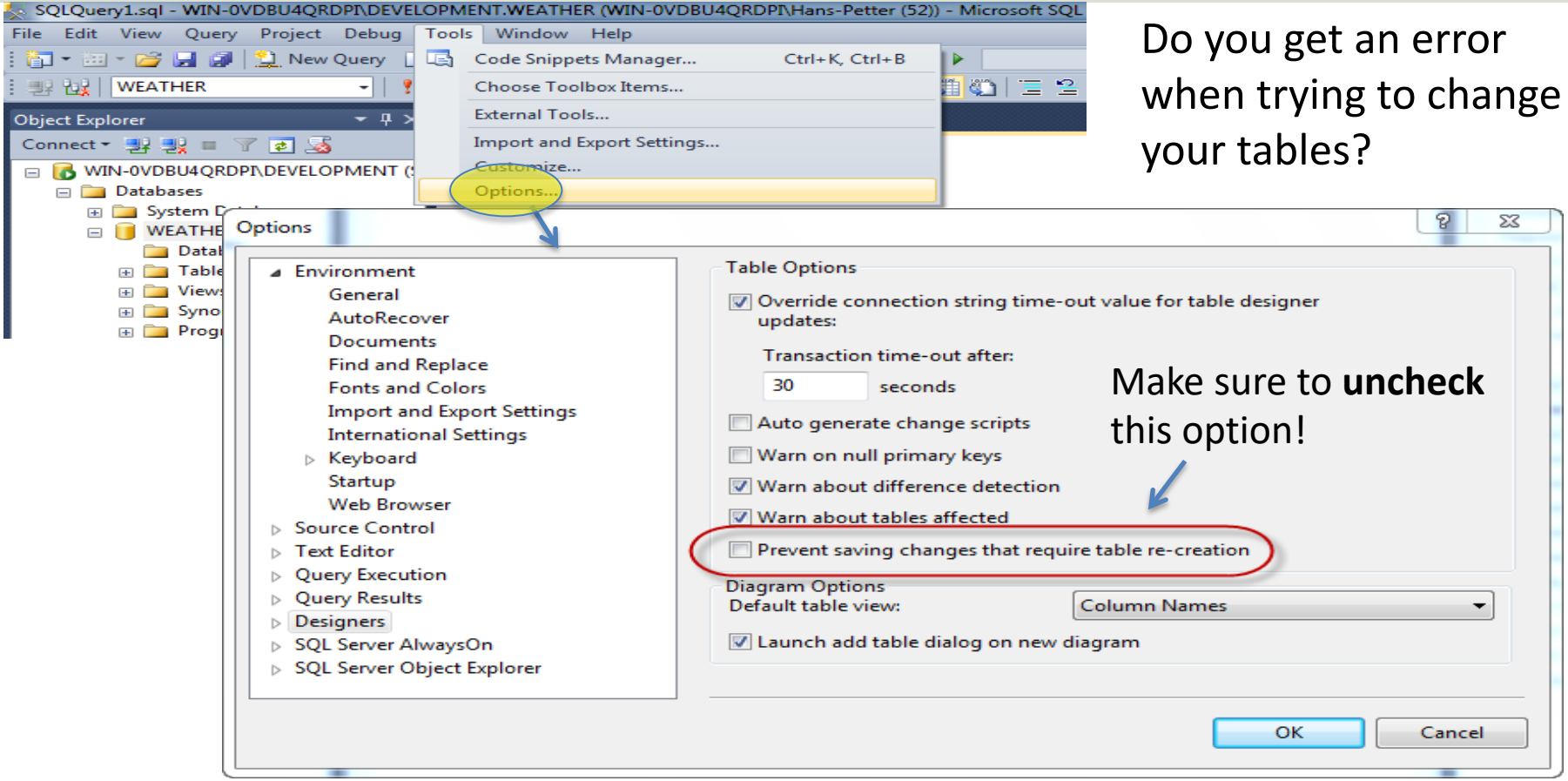

Name your database, e.g., “SCADA”

## How-To Create a New Database

# Microsoft SQL Server



# Microsoft SQL Server




Do you get an error when trying to change your tables?

Make sure to **uncheck** this option!

# OPC in Visual Studio

Hans-Petter Halvorsen

Table of Contents

# OPC in Visual Studio

- There is no built-in support for OPC in Visual Studio, so you need to install and use an external Add-on, Package or Library
- Examples of such Add-on packages:
  - OPC DA using "**Measurement Studio**" Add-on with "**MatrikonOPC Simulation Server**" as OPC DA Server
  - OPC UA using "**OPC UA .NET SDK**" with "**OPC UA Server Simulator**" as OPC UA Server

https://www.halvorsen.blog

# Measurement Studio

## + MatrikonOPC Simulation Server

Hans-Petter Halvorsen

Table of Contents

# Measurement Studio 2019

- Measurement Studio is an add-on to Visual Studio.
- Measurement Studio is used for development of measurement, control and monitoring applications using .NET and Visual Studio.
- Measurement Studio has a library (NetworkVariable) that makes it possible to communicate with **OPC DA servers**
- In order to communicate with an OPC Server using Measurement Studio you also need install the "**LabVIEW DSC Module**"
- Then you can use the "**Distributed System Manager**" to configure the system before you start to develop OPC DA Clients with Visual Studio
- Download Software here: https://www.ni.com/download

Measurement Studio

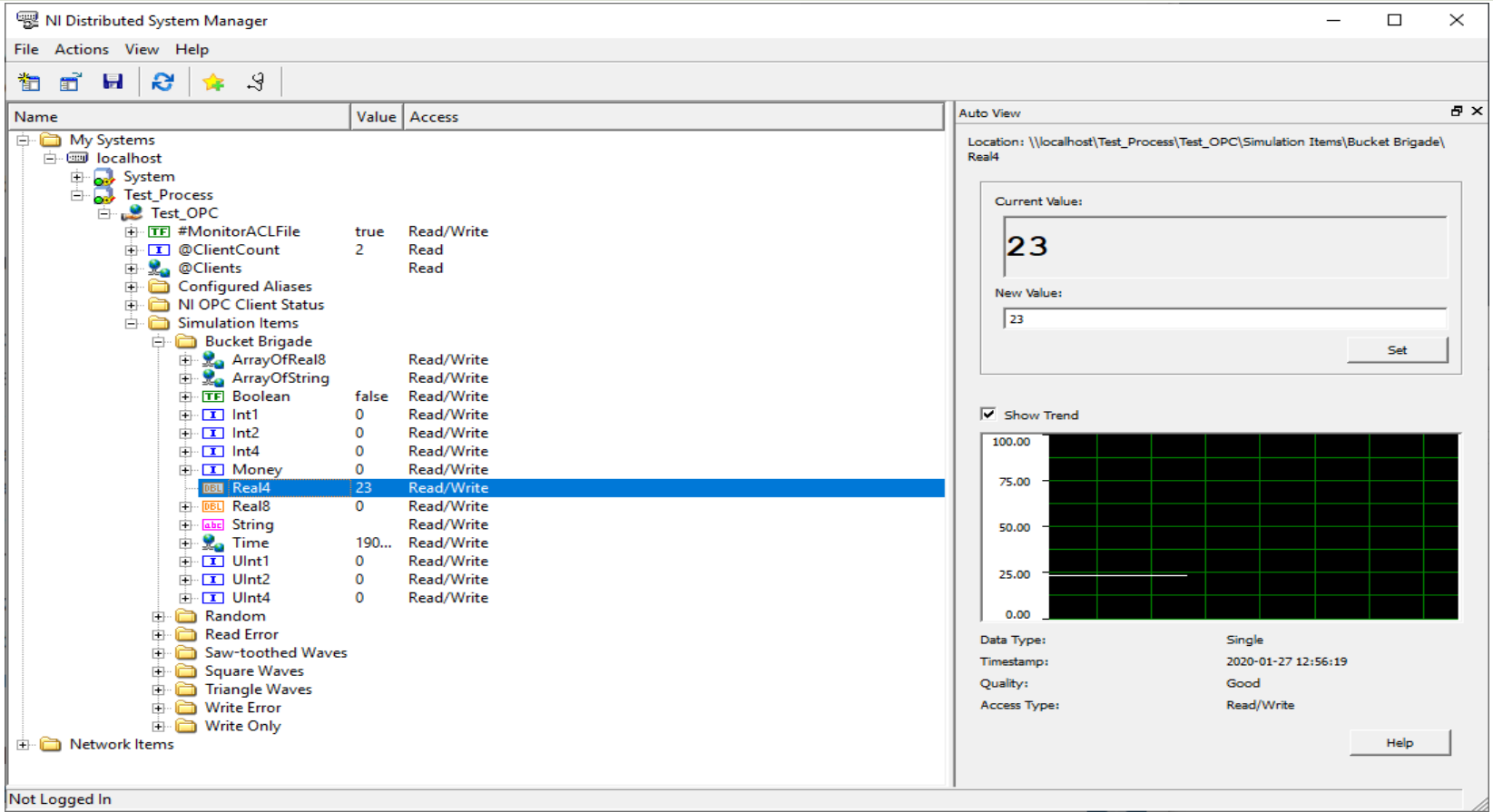
# Distributed System Manager

The following paragraphs explain how to use NetworkVariable with an OPC server using the LabVIEW DSC Run-Time System.

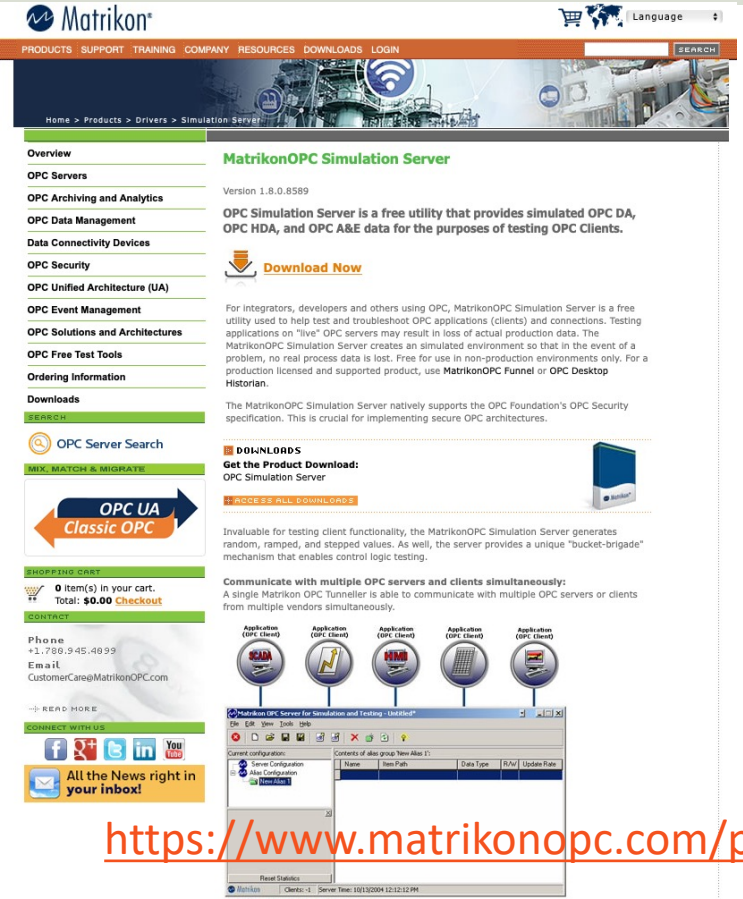
1. **Install LabVIEW Datalogging and Supervisory Control (DSC)** Run-Time System.
2. **Install your OPC server**. Only OPC2 and higher are supported by LabVIEW DSC Run-Time System.
3. Select Start»All Programs»National Instruments»**Distributed System Manager** to launch the application.
4. Right-click localhost and select **Add Process** to create a new process. Type Test_Process in the Add Process dialog box and click OK. Grouping variables by process allows you to organize your variables. You can start and stop processes independently, which allows you to easily manage your variables.
5. Right-click on Test_Process and select **Add I/O Server**.
6. For the I/O Server Type, **select OPC Client** and click Continue.
7. Type Test_OPC in the **Enter IO Server Name** dialog box and click OK.
8. **Select the OPC server** that you want to access through the Network Variable API from the list of Registered OPC Servers you installed in step 3 and click OK.
9. Right-click on Test_Process and select **Add Variable** to launch the **Shared Variable Properties** dialog box.
10. In the Shared Variable Properties dialog box, select the **Enable Aliasing** checkbox and click the Browse button.
11. In the Browse for Variable dialog box, select one of the OPC items from the OPC I/O server you configured in step 6.
12. Click OK to **bind the new variable to the OPC source**.
13. Click OK to return to NI Distributed System Manager. Use the new variable as you would any other shared variable. You can access the variable you have configured through the .NET **NetworkVariable class library**, as you would any other network variable.

http://zone.ni.com/reference/en-XX/help/375857B-01/mstudionetvar/netvar_opc/

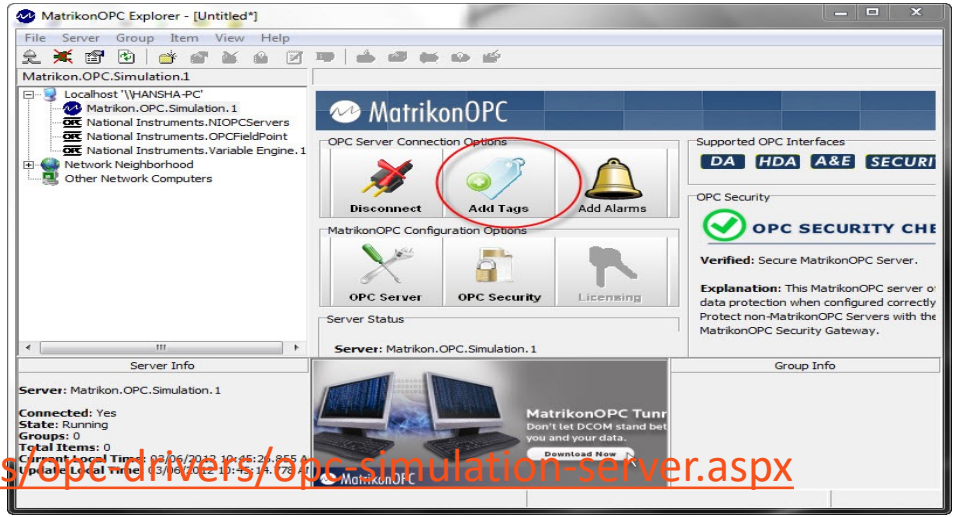
# Distributed System Manager



# MatrikonOPC Simulation Server




**MatrikonOPC Simulation Server** is a free utility that provides Simulated OPC DA, OPC HDA, and OPC A&E Data for the Purposes of Testing OPC Clients




https://www.matrikonopc.com/products/opc-drivers/opc-simulation-server.aspx

https://www.halvorsen.blog

# OPC UA .NET SDK

## + OPC UA Server Simulator

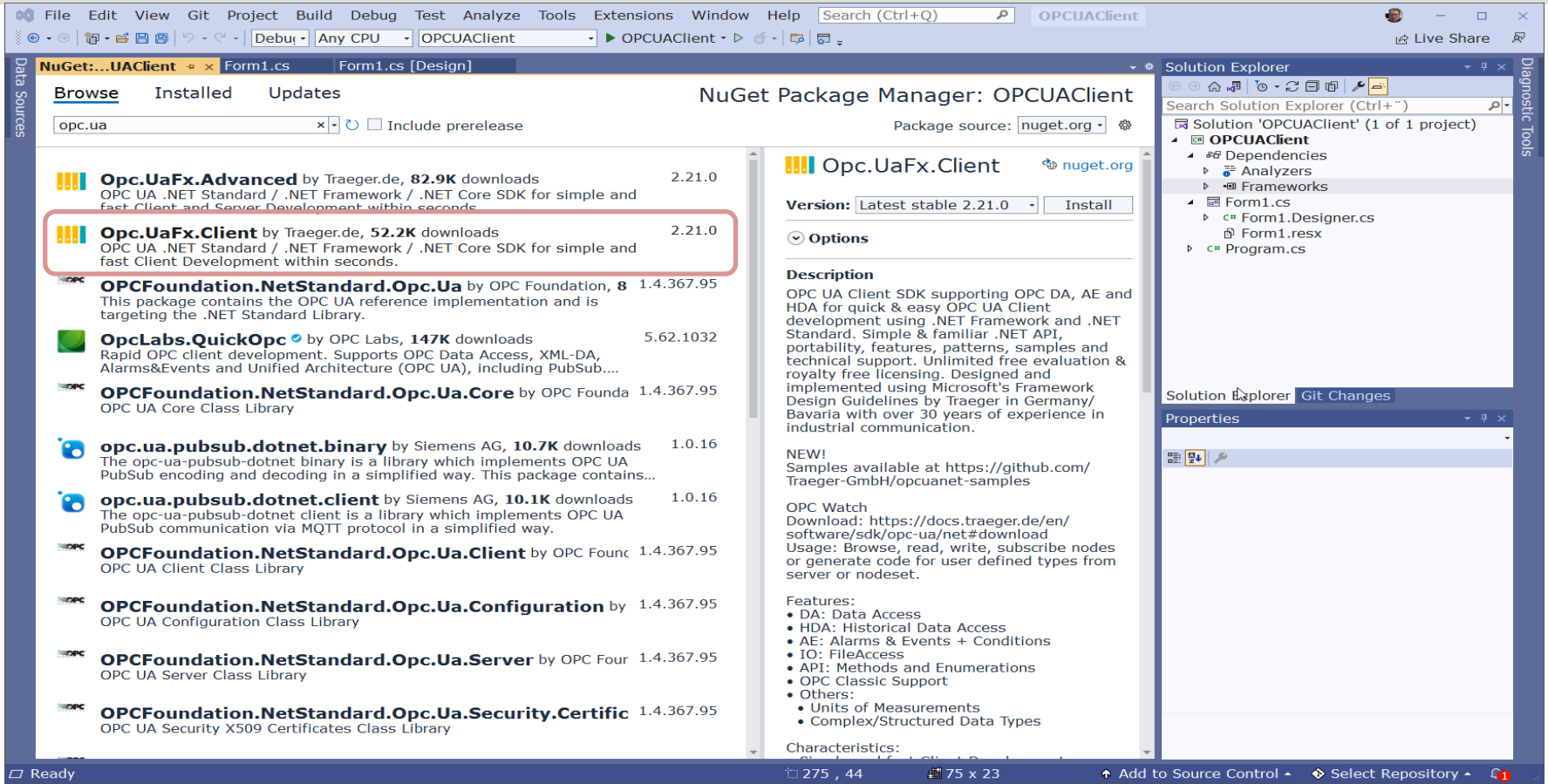
Hans-Petter Halvorsen

Table of Contents

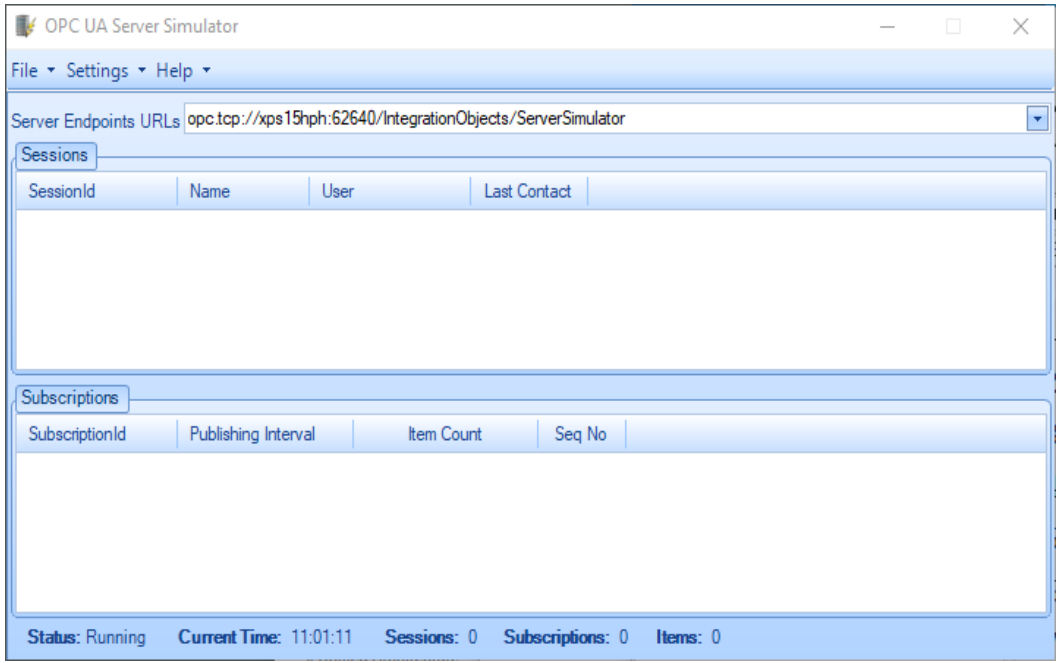
# OPC UA .NET SDK

- The “OPC UA .NET SDK” comes with an evaluation license which can be used unlimited for each application run for 30 minutes
- It comes in a NuGet Package you can install and use in your Visual Studio Project
- https://opcfoundation.org/products/view/opc-ua-net-sdk-for-client-and-server

# OPC UA .NET SDK Installation



# OPC UA Server Simulator




- This free OPC UA Server tool supports data access and historical access information models of OPC UA.
- It provides simulated real-time and historical data.
- It is possible to configure your own tags and the data simulation via CSV files.
- OPC UA clients can monitor real-time data and explore history data from this simulator.

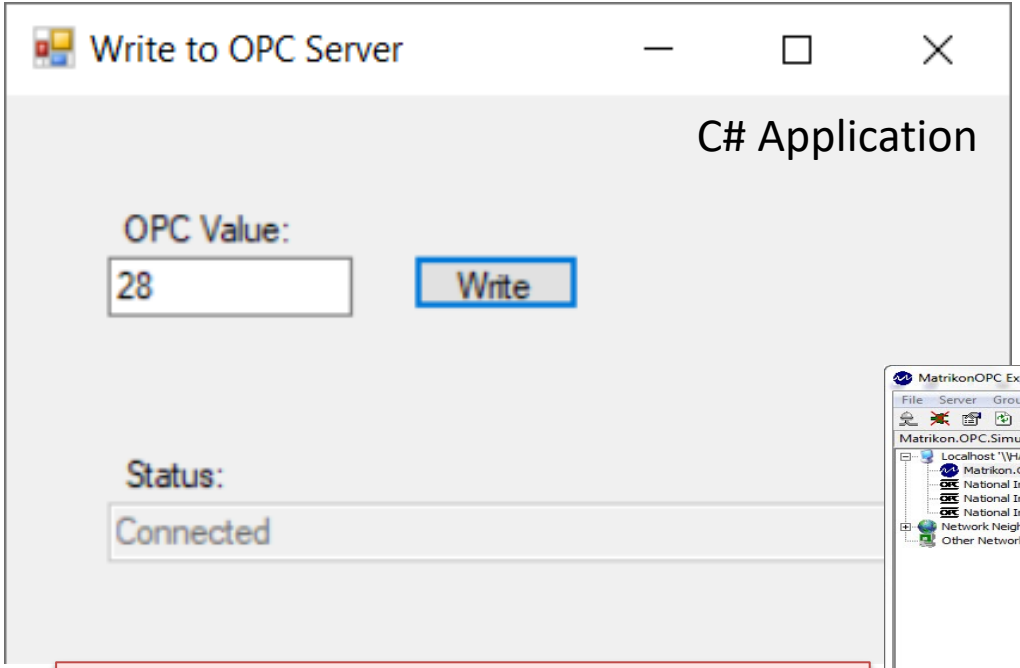
https://opcfoundation.org/products/view/opc-ua-server-simulator

# Write Data to OPC Server in C#

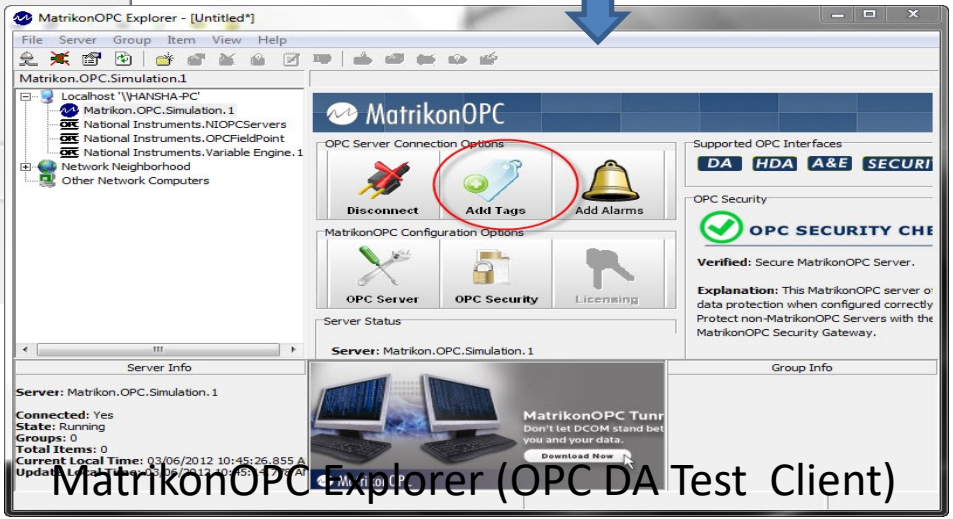
Hans-Petter Halvorsen

Table of Contents

# Measurement Studio Example




MatrikonOPC Simulation Server




MatrikonOPC Explorer (OPC DA Test Client)

This is just a very simple Example. You need to implement “OPC Write” within your “Control Application”

# Measurement Studio Example

```
using NationalInstruments.NetworkVariable;
namespace OPCExample
{
   public partial class Form1 : Form
   {
      private NetworkVariableWriter<double> _writer;
      private const string NetworkVariableLocation = @"\\localhost\OPCProcess\Temperature";

      public Form1()
      {
         InitializeComponent();
         ConnectOPCServer();
      }

     ...

   }
}
```

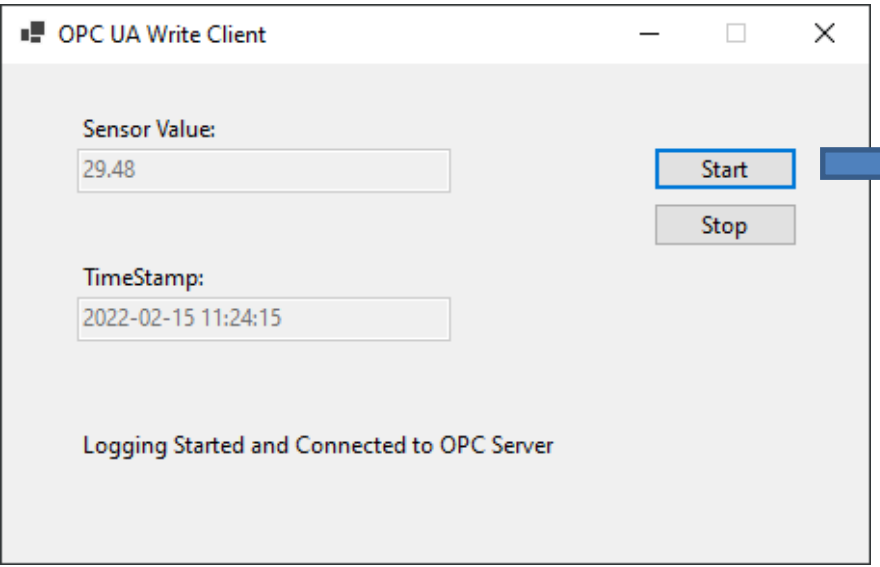
```
private void btnWriteData_Click(object sender, EventArgs e)
{
        double temperature;
        try
        {
           temperature = Convert.ToDouble(txtOpcData.Text);

           _writer.WriteValue(temperature);
        }
        catch (TimeoutException)
        {
           MessageBox.Show("The read has timed out.", "Timeout");
           return;
        }
}
```

```
private void ConnectOPCServer()
{
        _writer = new NetworkVariableWriter<double>(NetworkVariableLocation);

        _writer.Connect();

        txtStatus.Text = _writer.ConnectionStatus.ToString();
}
```

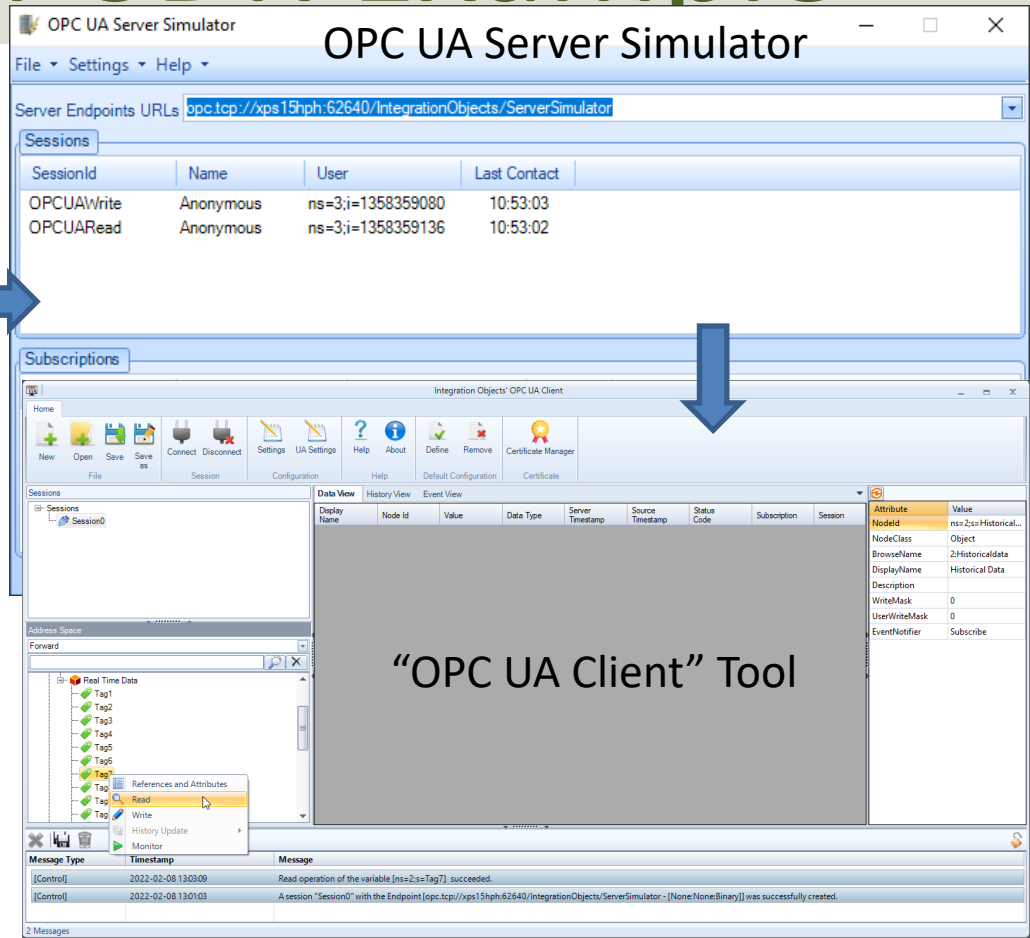
```
private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
        _writer.Disconnect();
}
```

# OPC UA .NET SDK Example

## OPC UA Write C# App




This is just a very simple Example. You need to implement “OPC Write” within your “Control Application”



# OPC UA .NET SDK Example

```
private void btnOpcWrite_Click(object sender, EventArgs e)
{
	string opcUrl = "opc.tcp://localhost:62640/";
	var tagName = "ns=2;s=Tag7";

	var client = new OpcClient(opcUrl);
	client.Connect();

	double temperature;
	temperature = Convert.ToDouble(txtOpcDataWrite.Text);

	client.WriteNode(tagName, temperature);

	client.Disconnect();
}
```

# Datalogging System

Hans-Petter Halvorsen

Table of Contents

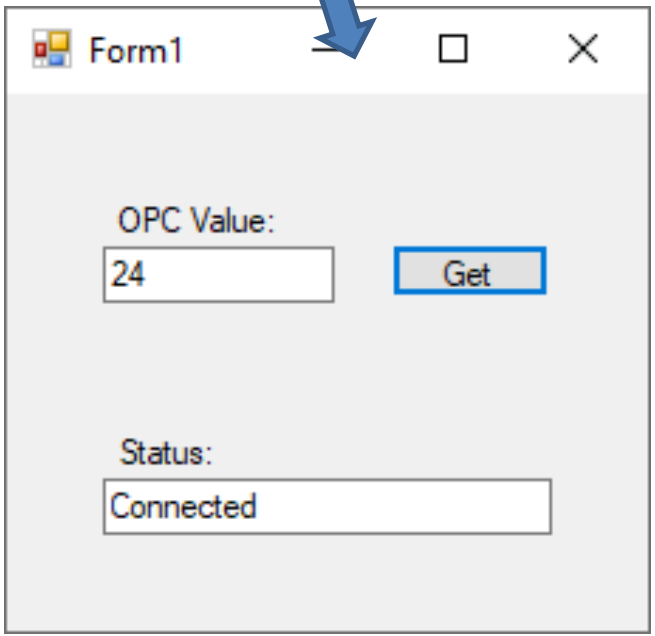
# Read Data from OPC Server

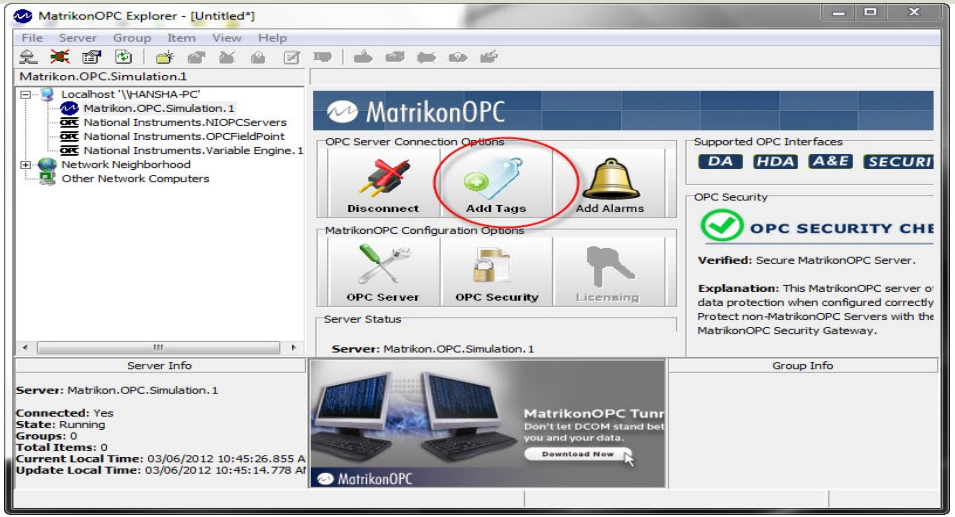
Hans-Petter Halvorsen

Table of Contents

# Measurement Studio Example

MatrikonOPC Simulation Server




This is just a very simple Example. You need to implement “OPC Read” within your “Datalogging Application”

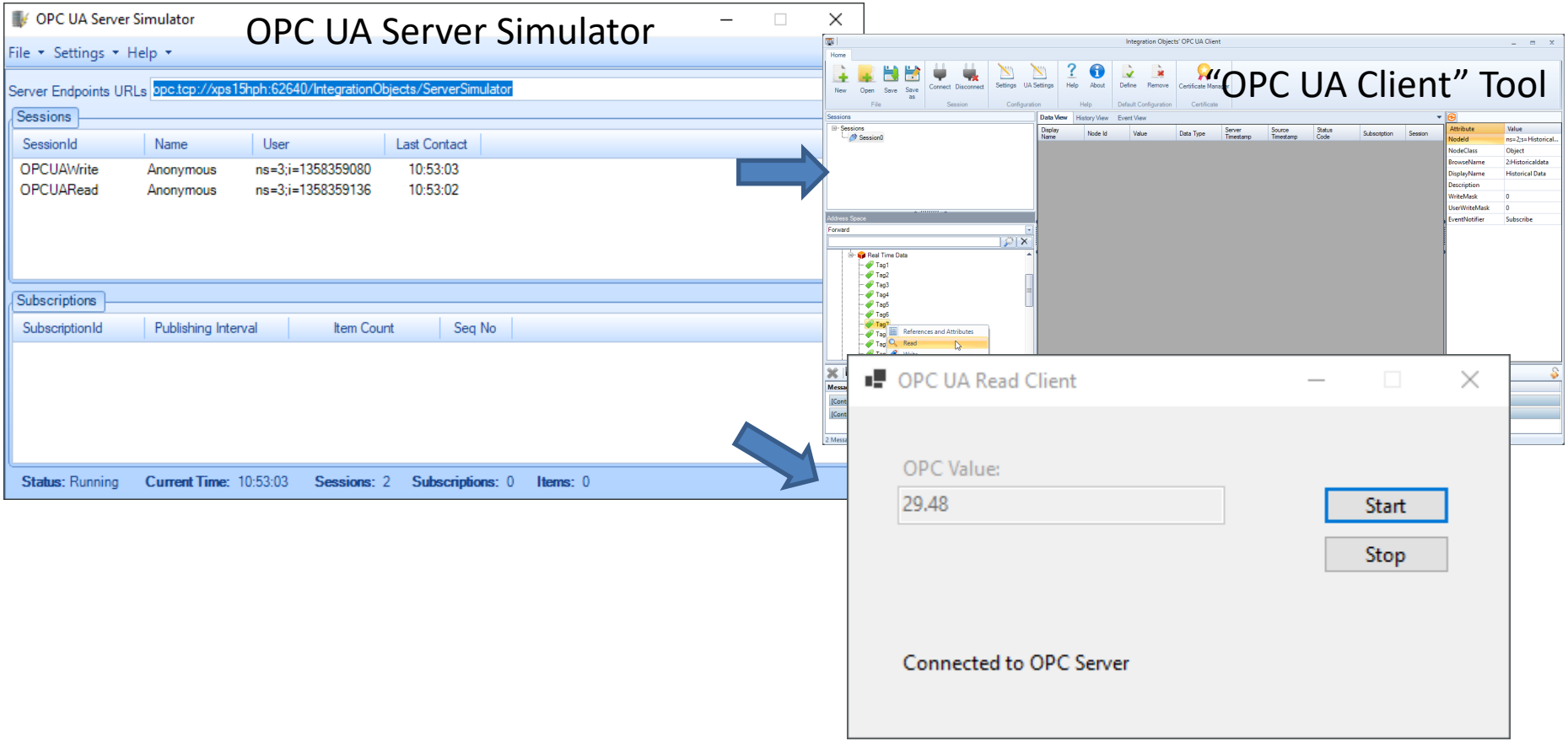
# Measurement Studio Example

```
using NationalInstruments;
using NationalInstruments.NetworkVariable;

namespace OPCExample
{
   public partial class Form1 : Form
   {
      private NetworkVariableReader<float> _reader;
      private const string NetworkVariableLocation = @"\\localhost\Test_Process\opctempdata";

      public Form1()
      {
         InitializeComponent();

         ConnectOPCServer();
      }

      private void btnGetData_Click(object sender, EventArgs e)
      {
         NetworkVariableData<float> opcdata = null;
         try
         {
            opcdata = _reader.ReadData();

            txtOpcData.Text = opcdata.GetValue().ToString();
         }
         catch (TimeoutException)
         {
            MessageBox.Show("The read has timed out.", "Timeout");
            return;
         }
      }
...
```

Note! This Code Snippet reads only one value once when clicking the button. You can use e.g. a **Timer** in order to read values at specific intervals.

```
....

      private void ConnectOPCServer()
      {
         _reader = new NetworkVariableReader<float>(NetworkVariableLocation);

         _reader.Connect();

         txtStatus.Text = _reader.ConnectionStatus.ToString();
      }

      private void Form1_FormClosing(object sender, FormClosingEventArgs e)
      {
         _reader.Disconnect();
      }

   }
}
```

# OPC UA .NET SDK Example



# OPC UA .NET SDK Example

```
private void btnOpcRead_Click(object sender, EventArgs e)
{
        string opcUrl = "opc.tcp://localhost:62640/";
        var tagName = "ns=2;s=Tag7";

        var client = new OpcClient(opcUrl);
        client.Connect();

        var temperature = client.ReadNode(tagName);
        txtOpcDataRead.Text = temperature.ToString();

        client.Disconnect();
}
```

# Save Data to SQL Server

Hans-Petter Halvorsen

Table of Contents

# Stored Procedure Example

```
IF EXISTS (SELECT name
           FROM   sysobjects
           WHERE  name = 'StudentGrade'
           AND            type = 'P')
          DROP PROCEDURE StudentGrade
OG

CREATE PROCEDURE StudentGrade
@Student varchar(50),
@Course varchar(10),
@Grade varchar(1)

AS

DECLARE
@StudentId int,
@CourseId int

select @StudentId=StudentId from STUDENT where StudentName = @Student

select @CourseId=CourseId from COURSE where CourseName = @Course

insert into GRADE (StudentId, CourseId, Grade)
values (@StudentId, @CourseId, @Grade)
GO
```

A Stored Procedure is like a Method in C# - it is a piece of code with SQL commands that do a specific task – and you reuse it

This part is not necessary – but if you make any changes, you need to delete the old version before you can update it

Procedure Name

Input Arguments

Internal/Local Variables
Note! Each variable starts with @

SQL Code (the "body" of the Stored Procedure)

Using the Stored Procedure:

```
execute StudentGrade 'John Wayne', 'SCE2006', 'B'
```

# Saving Data to SQL from C#

```
public void CreateBook(string connectionString, Book book)
{
            try
           {
                using (SqlConnection con = new SqlConnection(connectionString))
                {
                    SqlCommand cmd = new SqlCommand("CreateBook", con);
                       cmd.CommandType = CommandType.StoredProcedure;

                       cmd.Parameters.Add(new SqlParameter("@Title", book.Title));
                       cmd.Parameters.Add(new SqlParameter("@Isbn", book.Isbn));
                       cmd.Parameters.Add(new SqlParameter("@PublisherName", book.PublisherName));
                       cmd.Parameters.Add(new SqlParameter("@AuthorName", book.AuthorName));
                       cmd.Parameters.Add(new SqlParameter("@CategoryName", book.CategoryName));

                       con.Open();
                       cmd.ExecuteNonQuery();
                       con.Close();
                }
            }
            catch (Exception ex)
            {
                throw ex;
            }
}
```

It is recommended to create and use a **Stored Procedure**.
It is also recommended that the Connection String is stored in **App.config**

https://www.halvorsen.blog

# Alarm System

## Alarm Generation and Alarm Monitoring

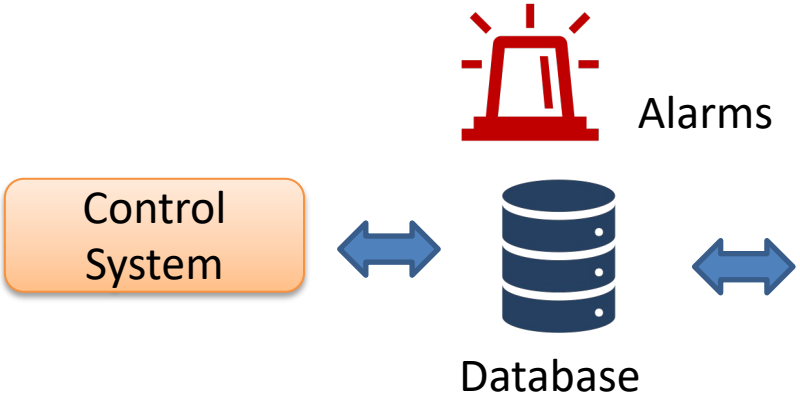
Hans-Petter Halvorsen

Table of Contents

# Alarm System

- The Alarm System typically checks for Alarms and saves the Alarm information in your Database.
- The operator should be able to see the Alarms and make some actions to these alarms, i.e., the operator should have the possibility to Acknowledge Alarms, Show Alarms with different Priorities, etc.
- A **Database Trigger** can be used in order to generate Alarms in the Database
- A Database **View** can be created and used to retrieve Alarm Data from the Database

# Alarm System Example




Trigger that checks new Process values against Alarm Levels and generates Alarms

Alarms that need to be Acknowledged by the Operator

Simple Example:

**Alarm Application**

Alarm List: Operator: Nils-Olav

| AlarmId | TagName | AlarmType | Priority | ActivitionTime | AckTime |
|---|---|---|---|---|---|
| 5 | Level | High | High | 12:45 | Ack |
| 6 | Temp | Low | Low | 12:10 | Ack |
| 9 | Pressure | High | Low | 12:20 | 12:22 |
| 12 | Level | Low | High | 12:30 | 12:31 |
| 14 | Pressure | High | Low | 12:35 | 12:36 |
| 4 | Level | HighHigh | High | 12:40 | 12:42 |
| | | | | | |

Alarms that have been Acknowledged by the Operator

# Trigger

A Trigger is executed when you insert, update or delete data in a Table specified in the Trigger.

```
IF EXISTS (SELECT name
               FROM   sysobjects
               WHERE  name = 'CalcAvgGrade'
               AND         type = 'TR')
            DROP TRIGGER CalgAvgGrade
GO

CREATE TRIGGER CalcAvgGrade ON GRADE
FOR UPDATE, INSERT, DELETE
AS

DECLARE
@StudentId int,
@AvgGrade float


select @StudentId = StudentId from INSERTED

select @AvgGrade = AVG(Grade) from GRADE where StudentId = @StudentId

update STUDENT set TotalGrade = @AvgGrade where StudentId = @StudentId

GO
```

This part is not necessary – but if you make any changes, you need to delete the old version before you can update it

Name of the Trigger

Specify which Table the Trigger shall work on

Specify what kind of operations the Trigger shall act on

Internal/Local Variables

Inside the Trigger you can use ordinary SQL statements, create variables, etc.

SQL Code (The "body" of the Trigger)

Note! "INSERTED" is a temporarily table containing the latest inserted data, and it is very handy to use inside a trigger

https://www.halvorsen.blog

# ASP.NET Core

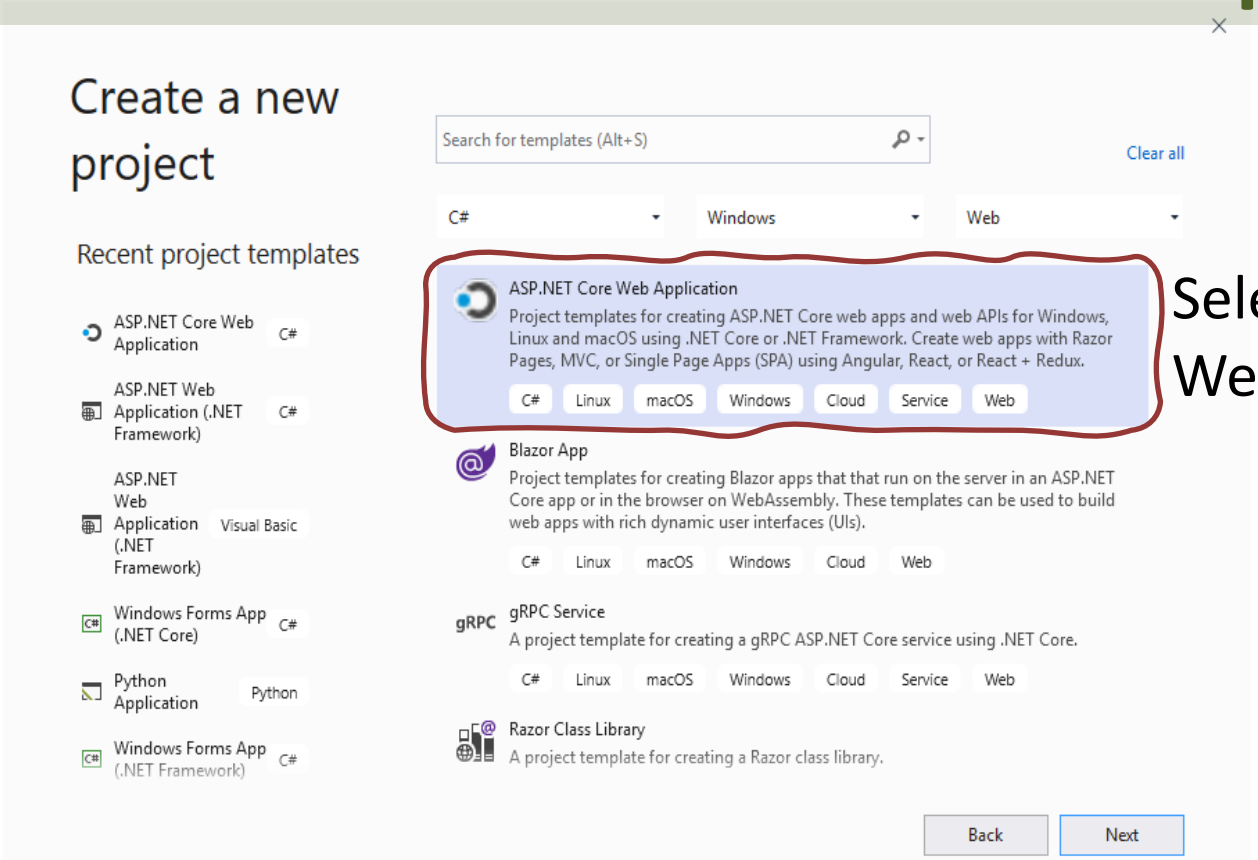
Hans-Petter Halvorsen

Table of Contents

# ASP.NET Core Web Application

- ASP.NET is a Web Framework for creating Web Applications
- ASP.NET is integrated with Visual Studio and you will use the C# Programming Language
- .NET Core is cross-platform, meaning it will work on Windows, Linux and macOS.
- ASP.NET Core is Microsoft’s newest baby, and it is the future of Web Programming

# ASP.NET Core Web Application




Select the ASP.NET Core Web Application Project

# ASP.NET Core Examples

Recommended Videos:

- ASP.NET Core – Introduction: https://youtu.be/zkOtiBcwo8s
- **ASP.NET Core - Database Communication: https://youtu.be/0Ta3dQ3rxzs**
- ASP.NET Core - Database CRUD Application: https://youtu.be/k5TCZDwTYcE

Download Examples here: https://www.halvorsen.blog/documents/programming/web/aspnet

# ASP.NET Core Resources


- Textbook
- Videos
- Tutorials
- Example Code

https://www.halvorsen.blog/documents/programming/web/aspnet

https://www.halvorsen.blog

# Cyber Security

Hans-Petter Halvorsen

Table of Contents

# Cyber Security in IACS Systems

- CSMS – Cyber Security Management System
- IACS – Industrial Automation and Control Systems
- Security is critical in IACS systems because a potential hacker can do great damage
- In the Norwegian energy and oil and gas sector alone, more than 50 cyber security incidents are detected the last year.*

Source: Norwegian National Security Authority

# Cyber Attacks

- In computers and computer networks an **attack** is any attempt to expose, alter, disable, destroy, steal or gain unauthorized access to or make unauthorized use of the system
- A **cyber attack** is any type of action that targets computer information systems, infrastructures, computer networks, or personal computer devices.
- An **attacker** is a person or process that attempts to access data, functions or other restricted areas of the system without authorization, potentially with malicious intent

https://en.wikipedia.org/wiki/Cyberattack

# Cyber Security Standards

- To protect the cyber environment of a user or organization.
- This environment includes users themselves, networks, devices, all software, processes, information in storage or transit, applications, services, and systems that can be connected directly or indirectly to networks
- Reduce the risks and prevent Cyber Attacks
- IEC62443 – Cyber Security standard for IACS systems

https://en.wikipedia.org/wiki/Cyber_security_standards

# Cyber Security in IACS Systems

Things to consider:

- Data & Cyber Security Issues regarding your SCADA Software
- How can you secure your Software against threats and vulnerabilities?
- What kind of precautions have you done when implementing your system?
- What can/should you/have you done do to protect your Software?

# Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog