2016.10.27

SCADA Lab

Supervisory Control and Data Acquisition

http://home.hit.no/~hansha/?lab=scada

Hans-Petter Halvorsen, M.Sc.

Lab Overview

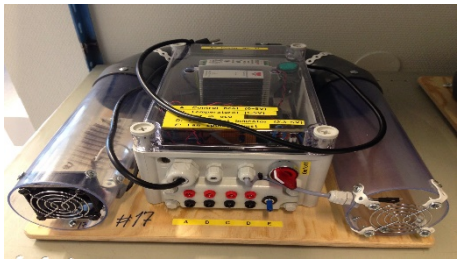
USB-6008

Fuji PXG5 PID Controller

HIL Simulation

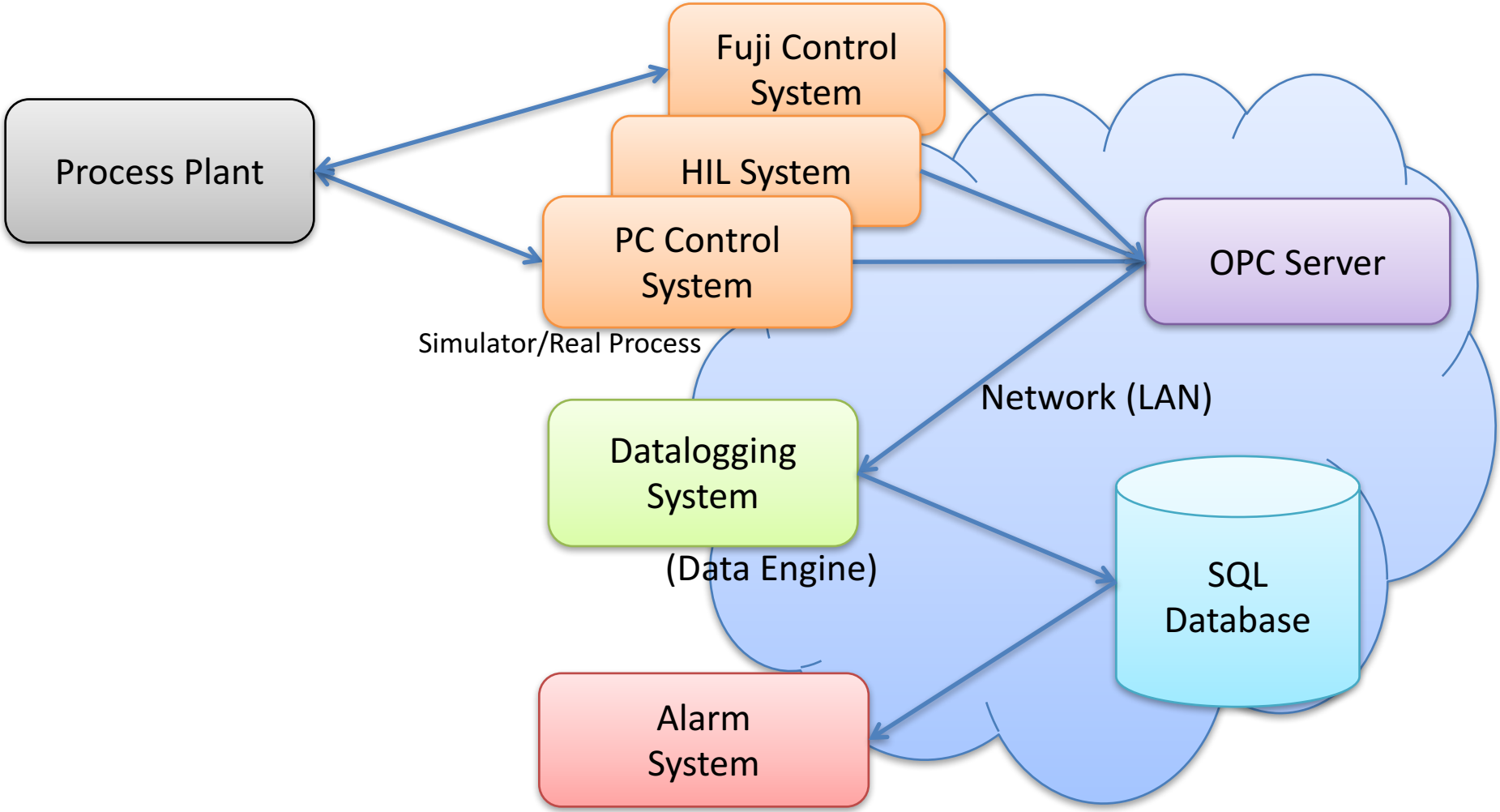
Matrikon OPC Simulation Server

Air Heater



Simulator/Real Process

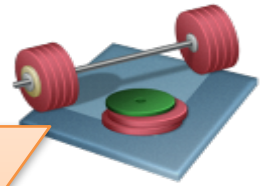
The SCADA system should be distributed, meaning that the different components could be located on different computers in a network


Process Plant
Fuji Control System
HIL System
PC Control System
Simulator/Real Process
OPC Server
Network (LAN)
Datalogging System
(Data Engine)
SQL Database
Alarm System

Lab Assignment

- In this Assignment you will need “everything” you have learned in Lectures and previous Lab Assignments
- Make sure to read the whole assignment before you start to solve any of the problems.
- If you miss assumptions for solving some of the problems, you may define proper assumptions yourself.
- It is assumed you use proper “System Design” principles as learned in the Lectures
- The suggested Tasks are somewhat loosely defined and more like guidelines, so feel free to interpret the Tasks in your own way.
- Feel free to Explore! – and add more value to the suggested solutions

Lab Assignment Overview



All Programming shall be done with **Visual Studio/C#**

Suggested Steps:

A. Design the Database using **ERwin**

B. Implement the Database using **SQL Server**

C. Create a **Control System** and send data to **OPC** Server.
   - Start with a model of the system. When the Simulations works, try with the real system
   - You should create and use your own PI(D) controller and Lowpass Filter from Scratch (there is no built-in PID in C#)

D. Control the System with a **Fuji PXG5 PID** Controller. The Fuji Controller should first be tested using **HIL Simulation**. Send Data to OPC

E. **Datalogging System**: Read Data from OPC Server and Log the Data to a SQL Server Database

F. **Alarm System**: Create an Alarm Generation and Monitoring System

G. **Distributed System/Network** (OPC Server, Database Server, OPC Clients should be distributed on different computers in a network)

See next slides for details...

Necessary Software

CA ERwin Data Modeler Community Edition, free download from Internet

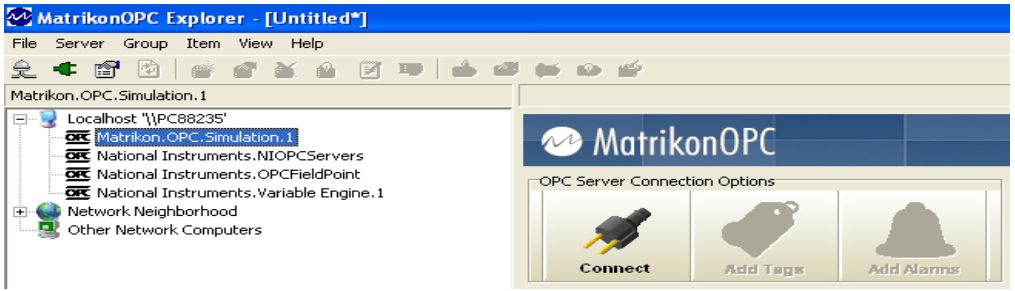
Visual Studio

You can download Visual Studio from DreamSparks

Microsoft® SQL Server®

You can download SQL Server Express for Free either from Internet or DreamSparks

MatrikonOPC Server for Simulation

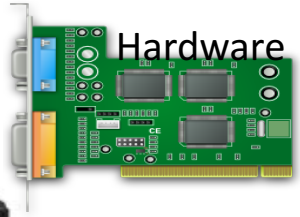


You can download MatrikonOPC Server for Simulation for Free from Internet

Make sure to install the necessary Software before you go to the laboratory!

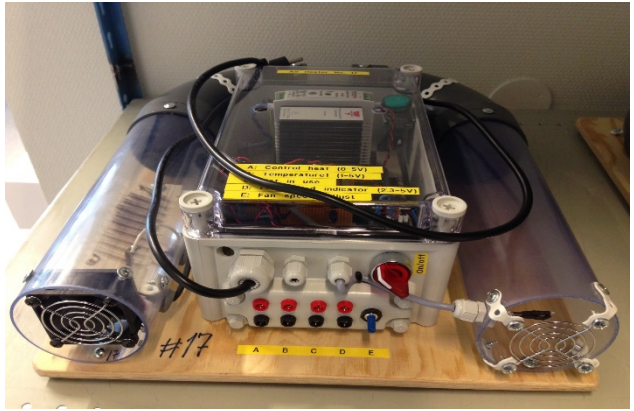
Hardware

Fuji PXG5 PID Controller



Your Personal Computer

USB-6008



Air Heater

Switch/Router

Ethernet Cables

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

Visual Studio

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

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

NATIONAL INSTRUMENTS LabVIEW

My System is not Working??

Multimeter, etc.

"Google It"!

You probably will find the answer on the Internet

Google

Another person in the world probably had a similar problem

Your hardware device most likely works, so you don't need a new device! Still not working after Troubleshooting & Debugging? Fill out an "Equipment Error Form"

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.

Recommended Litterature

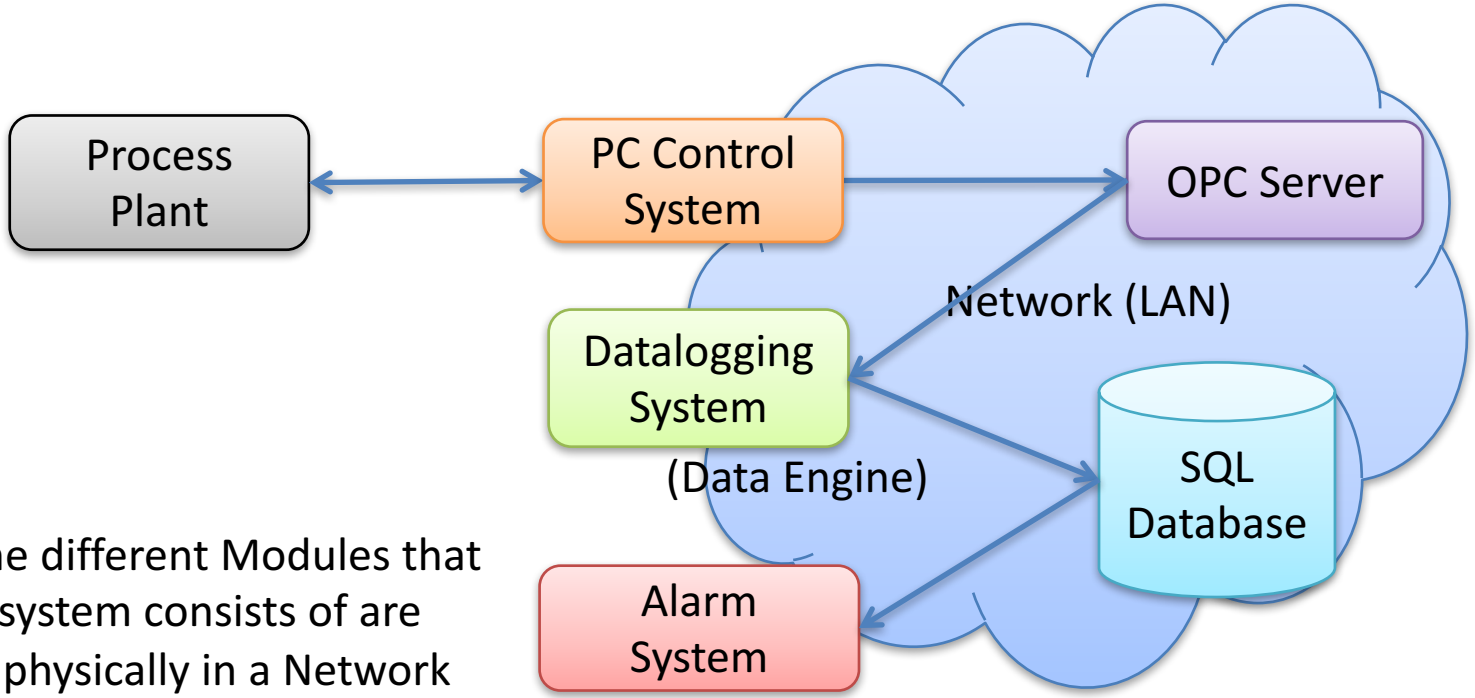
- Tutorial: Introduction to Visual Studio and C#
  http://home.hit.no/~hansha/?tutorial=csharp
  Tutorial: Data Acquisition in C#
  http://home.hit.no/~hansha/?tutorial=csharp_daq
- Structured Query Language
  http://home.hit.no/~hansha/?tutorial=sql
- Database Communication in Visual Studio/C#
  http://home.hit.no/~hansha/documents/database/database_visual_studio.htm
- Writing a Technical Lab Report
  http://home.hit.no/~hansha/documents/lab/lab_documents.htm
- Developing an OPC Client Application Using Visual Basic
  http://www.ni.com/white-paper/3259/en/

Introduction to SCADA

Supervisory Control and Data Acquisition

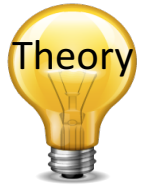
Hans-Petter Halvorsen, M.Sc.

SCADA System Overview



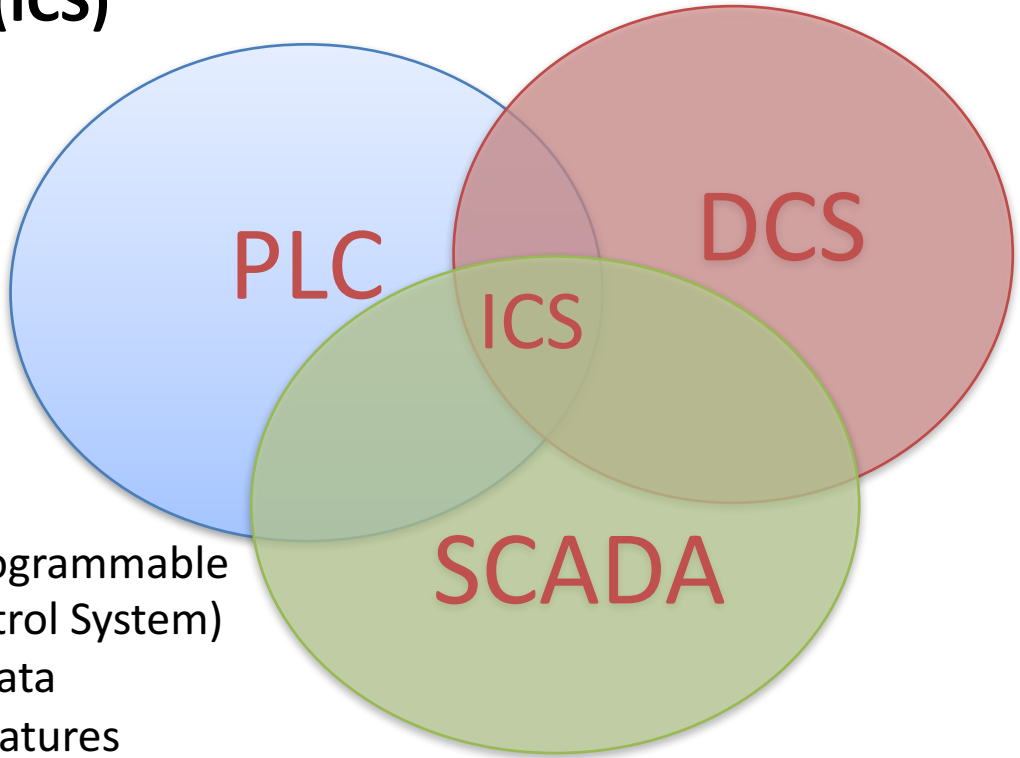
Typically, the different Modules that the SCADA system consists of are distributed physically in a Network (local network or over Internet)

SCADA System



SCADA (Supervisory Control And Data Acquisition) is a type of **Industrial Control System (ICS)**

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



Industrial 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)

Theory

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

cRIO

Programmable Automation Controller (**PAC**) 4

LabVIEW 1

I/O Module

5

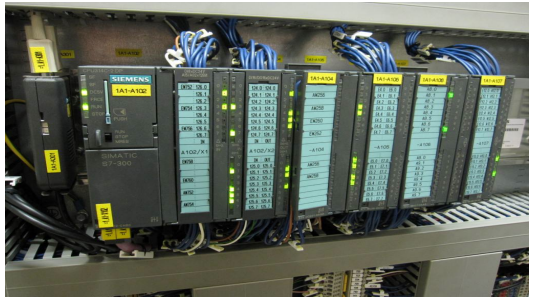
PC based Control System/**SCADA** System (Supervisory Control And Data Acquisition)

Industrial **PID** Controller

2 Distributed Control Systems (**DCS**)

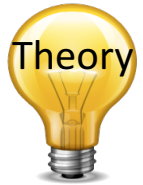
DeltaV

PLC (Programmable Logic Controller) 3



Siemens PLC

PC-based Control System



Industrial PID Controller

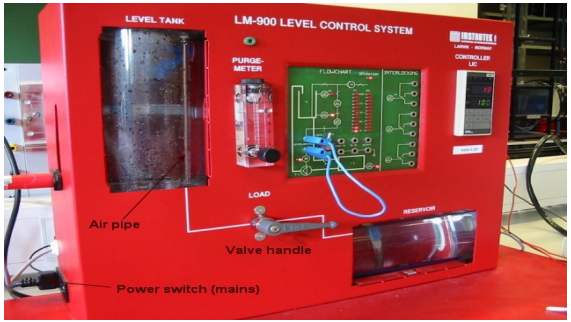
Embedded system with built-in PID algorithm, etc.

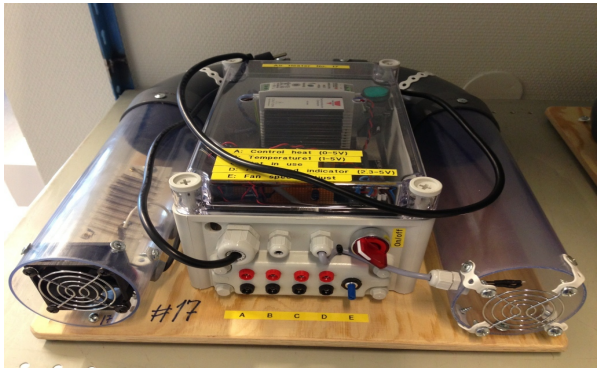
PID Control using PC and I/O Module

I/O Module

PC-based Control System Example

Theory





Process

0-5V/1-5V

0-5V

Analog In
Measurement(s)

Analog Out
Control Signal

I/O Module

USB-6008

AD Converter

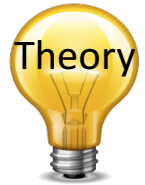
DA Converter

USB

Controller (PID) and
Lowpass Filter
Implementation

C#

PC-based Control System



PC with Control Application

Control Signal

u

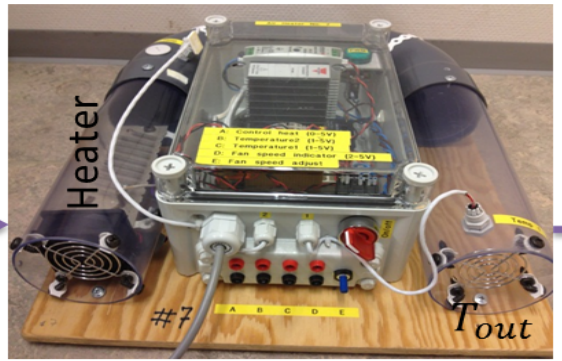
USB-6008 DAQ

AO

$0-5V$

u

Air Heater Process



$1-5V$

T_{out}

Temperature

Analog Measurement

AI

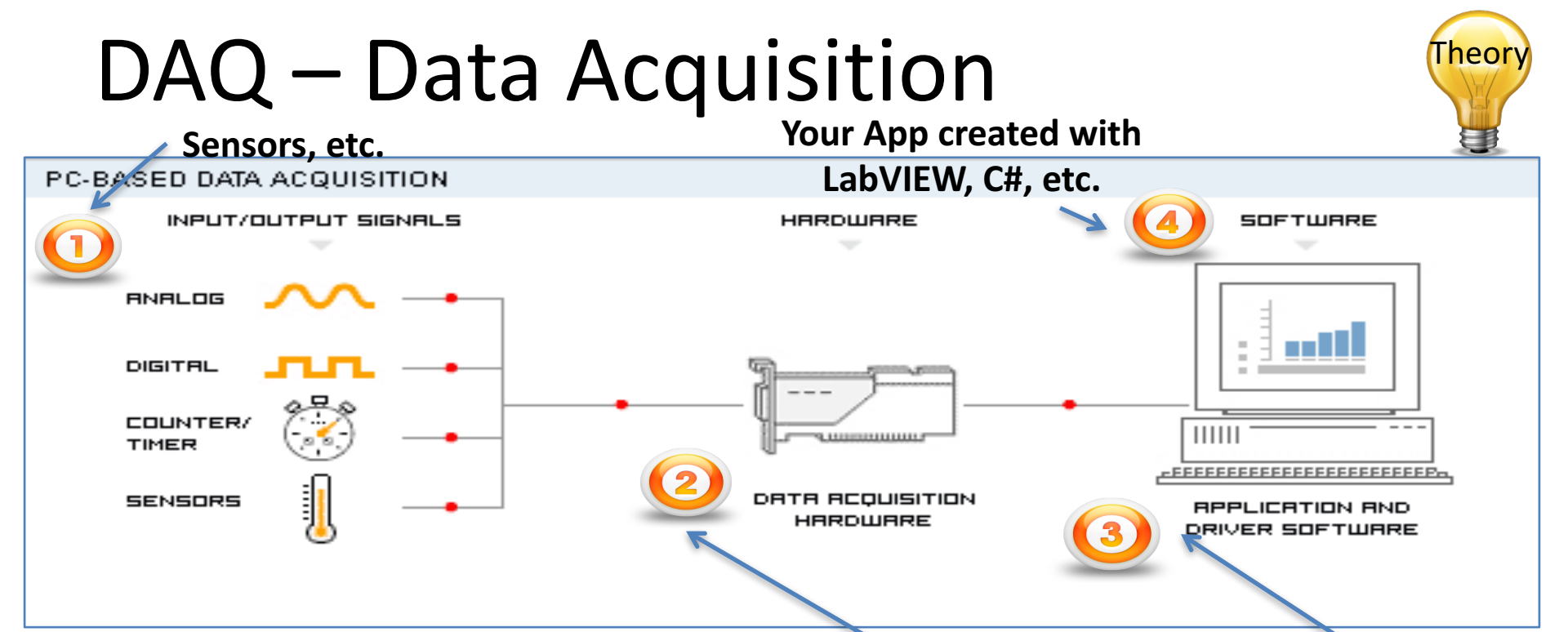
USB-6008 DAQ

Digital Signal

Process Value

y

DAQ – Data Acquisition



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)

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 should be implemented as separate applications because they should be able to run on different computers in a network (distributed).

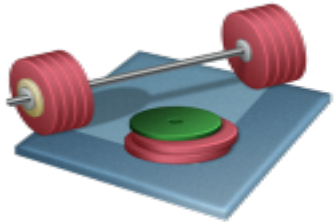
Task A

ERwin

Database Design/Modelling

Hans-Petter Halvorsen, M.Sc.

Database Design & Modelling



- Design the Database using ERwin (ER diagram)
- Generate a SQL Script that can be used to create the tables in SQL Server

See next slides for details about Database Requirements

Database Requirements

Tag Configuration:

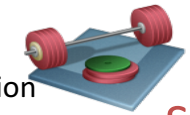
The TAG table(s) could e.g. have the following columns:

- 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.

Students: Design the the Tables according to the Requirements

Database Requirements

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:

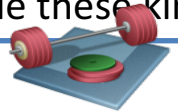
- Timeout; no input from a sensor or another computer system within a specific amount of time,
- High High (HH) or Low Low (LL) alarm; a critical alarm condition,
- High (H) or Low alarm (L)
- I/O device errors
- System device errors
- etc.

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

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

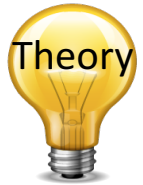
- Screen; display the alarms
- Keyboard; alarm operations
- Horn; indicate an active alarm, or security alarm
- Lamp; indicate an active alarm, or an active alarm by blinking and an acknowledge alarm by a steady light
- Printer; logging of the alarm states
- SMS
- E-mail
- Etc.

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



Students: Design the the Tables according to the Requirements

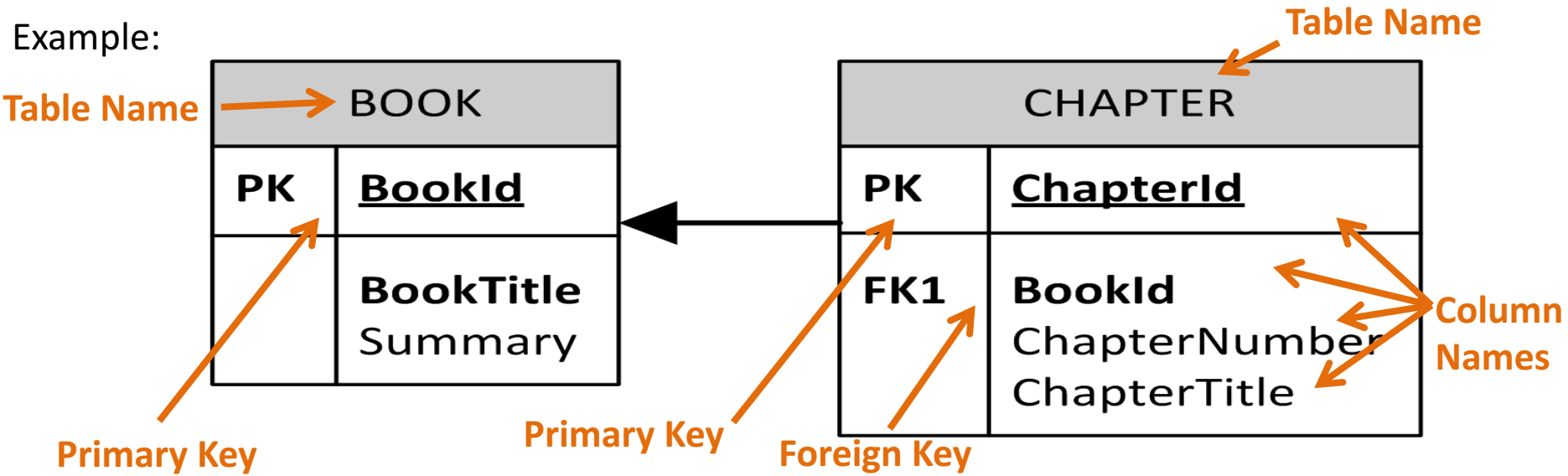
Database Design – ER Diagram



ER Diagram (Entity-Relationship Diagram)

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

Example:



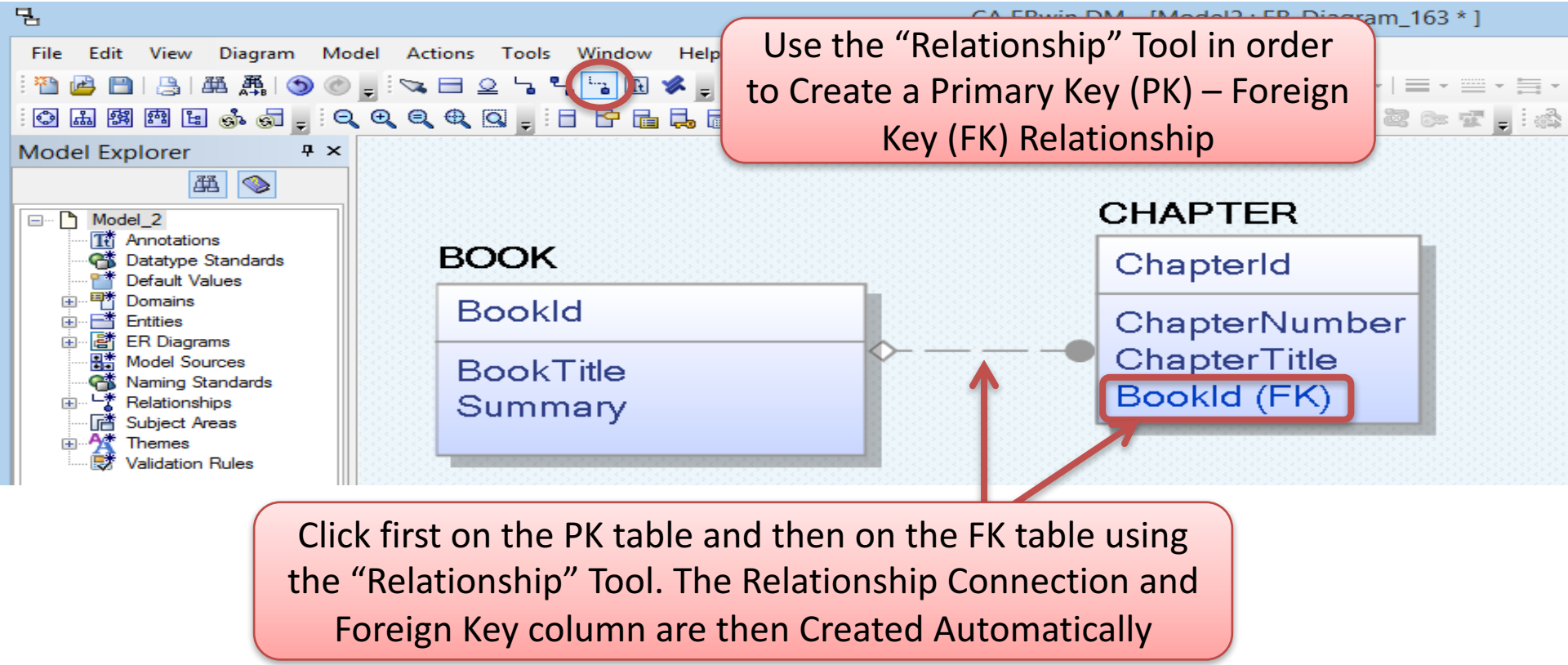
Relational Database. In a relational database all the tables have one or more relation with each other using Primary Keys (PK) and Foreign Keys (FK). Note! You can only have one PK in a table, but you may have several FK's.

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

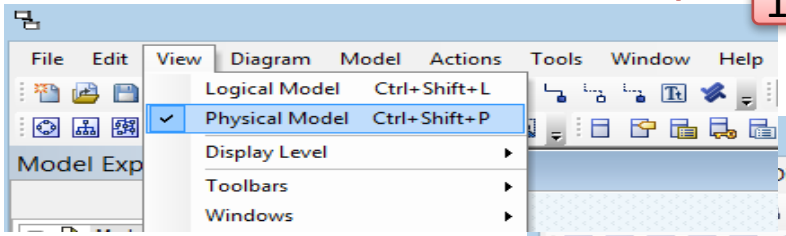
How-To: Create Primary Key (PK) – Foreign Key (FK) Relationships:



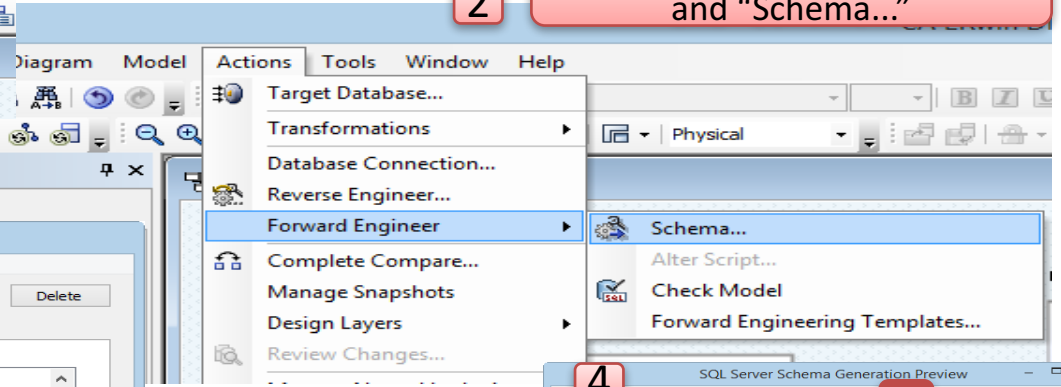
Introduction to ERwin

How-To: Create a SQL Script

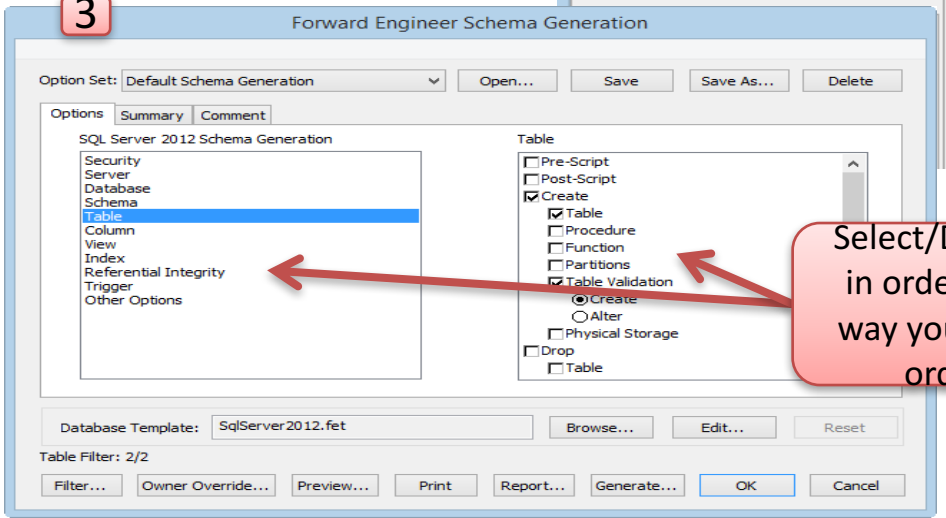
1 Make sure you are using the Physical Model



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

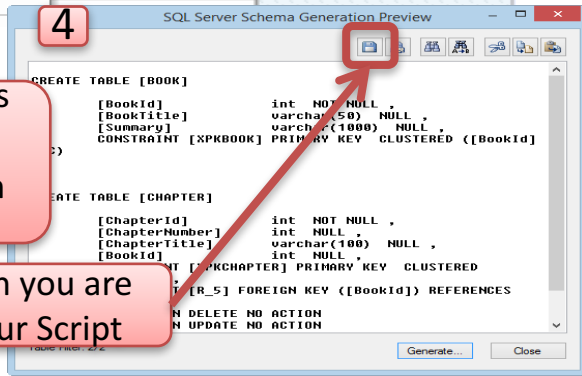


3



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

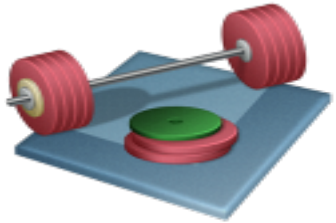
Congratulations! - You are finished with the Task

Task B

Database Implementation

Hans-Petter Halvorsen, M.Sc.

Database Implementation



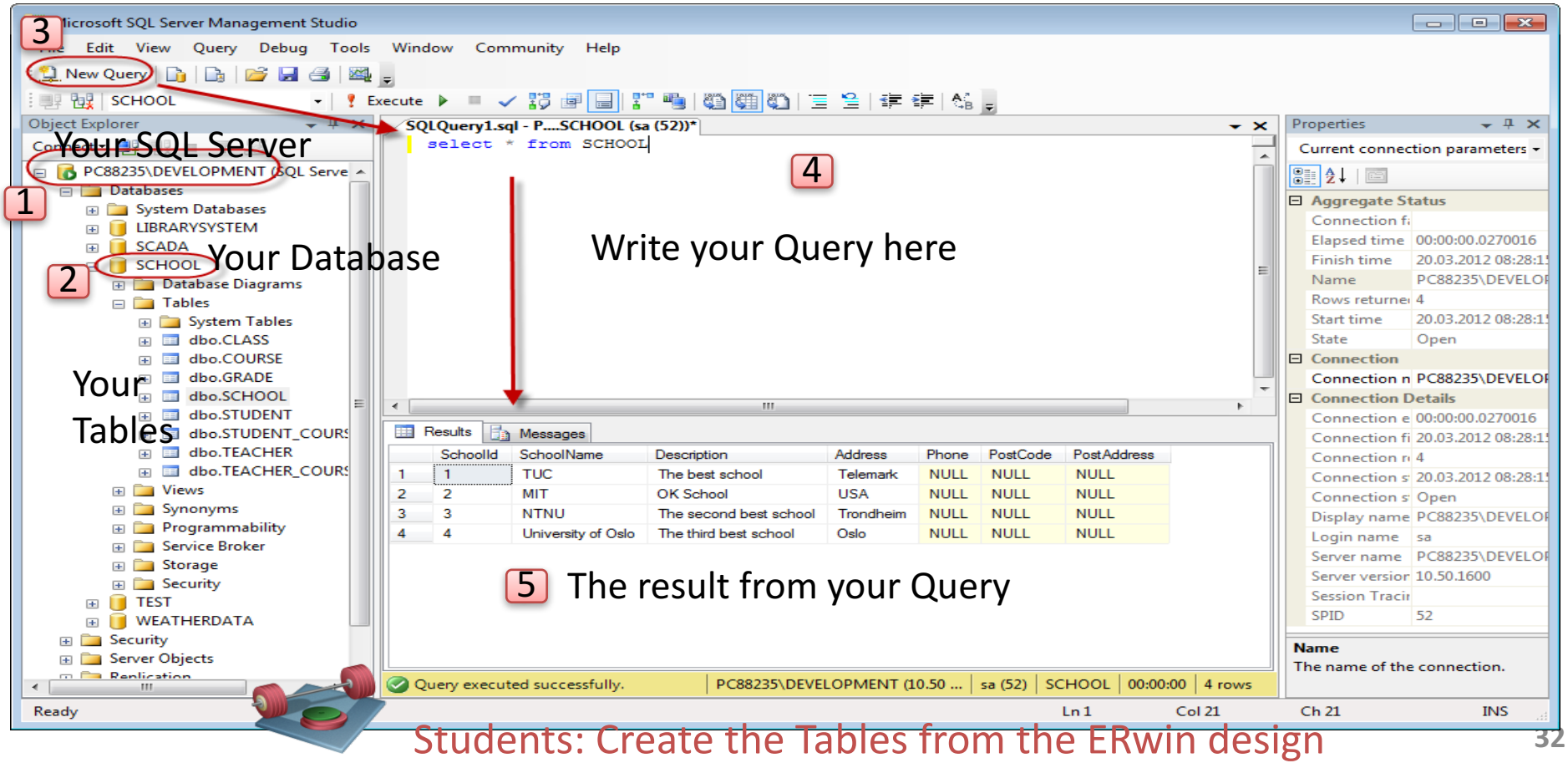
- Implement the Database using SQL Server
- Make necessary Tables, Views, Stored Procedures, etc.
- You should insert the Tables based on the Script generated from ERwin.

Microsoft SQL Server

Name you database, e.g., “SCADA”

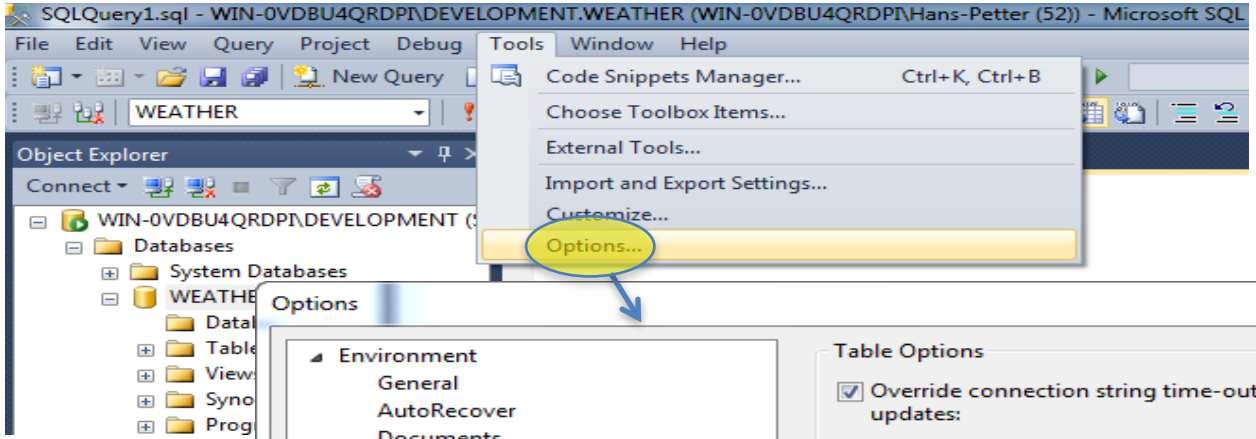
How-To Create a New Database

Microsoft SQL Server

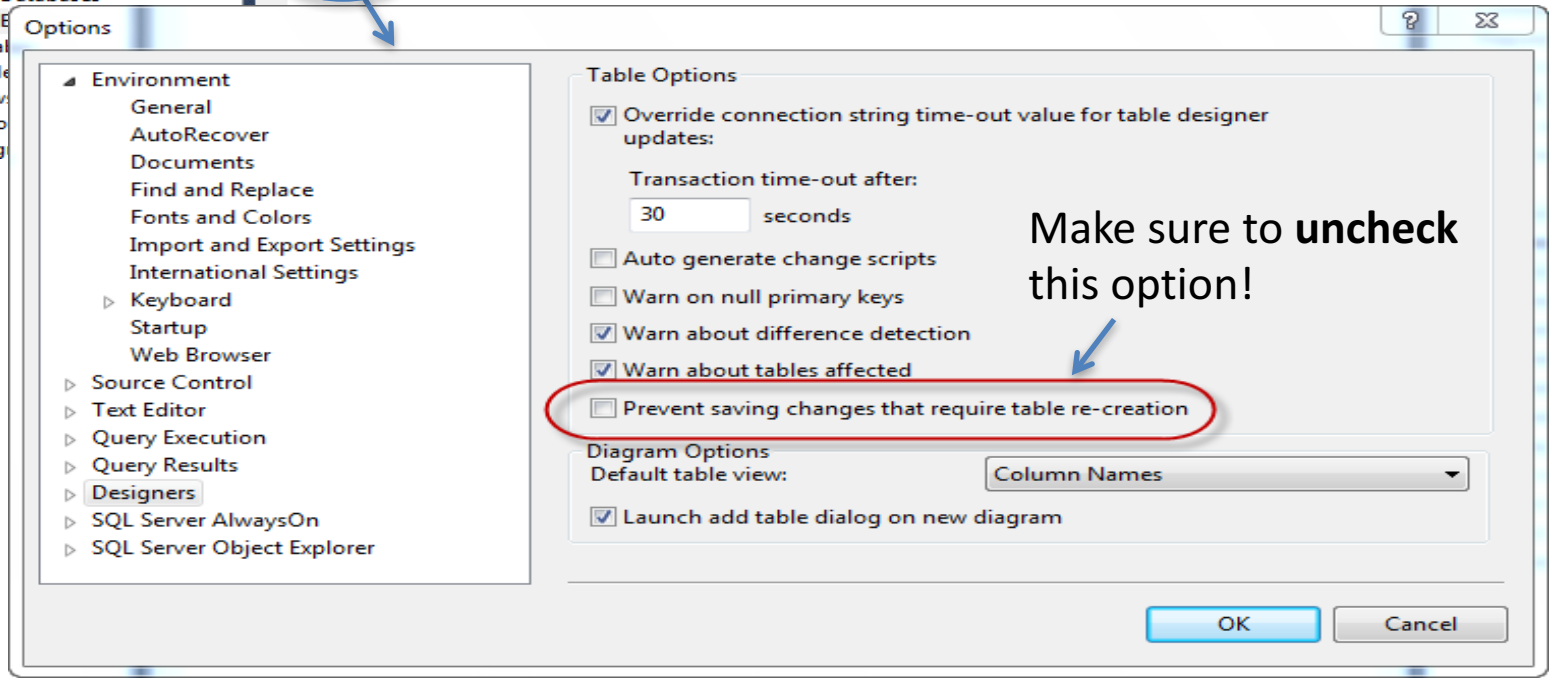


Students: Create the Tables from the ERwin design

Microsoft SQL Server



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



Make sure to **uncheck** this option!

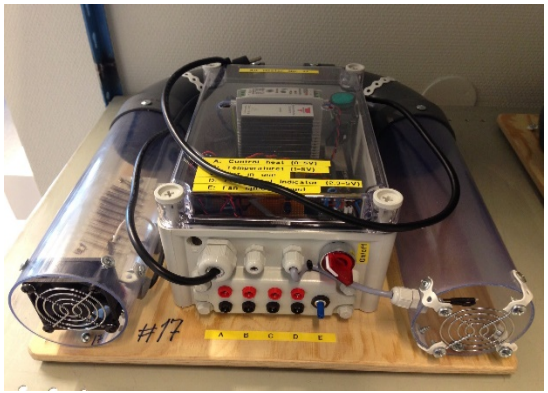
Congratulations! - You are finished with the Task

Task C

Control System

Computer

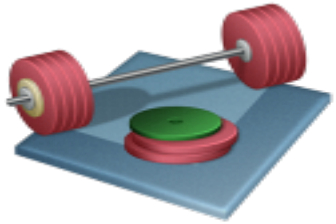
I/O Module



Air Heater

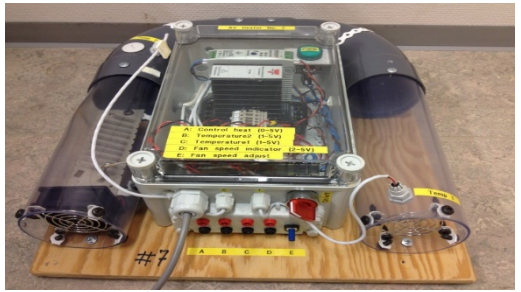
Hans-Petter Halvorsen, M.Sc.

Control System



- Create an Application in **C#** that controls the Air Heater Temperature. Use standard **PI**(D) control.
- Start using a **Simulator** created from scratch in C#
- You need to create a **HMI** (Human Machine Interface) for your application.
- The process data should be sent to an **OPC** Server. You should write the Control Signal (*u*) and the Process Value (*y*) to the OPC Server
- You should create and use your own **PI(D) controller** + **Lowpass Filter**

Air Heater Mathematical Model



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

Note! You need to make a discrete version of this model and implement in C#

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:

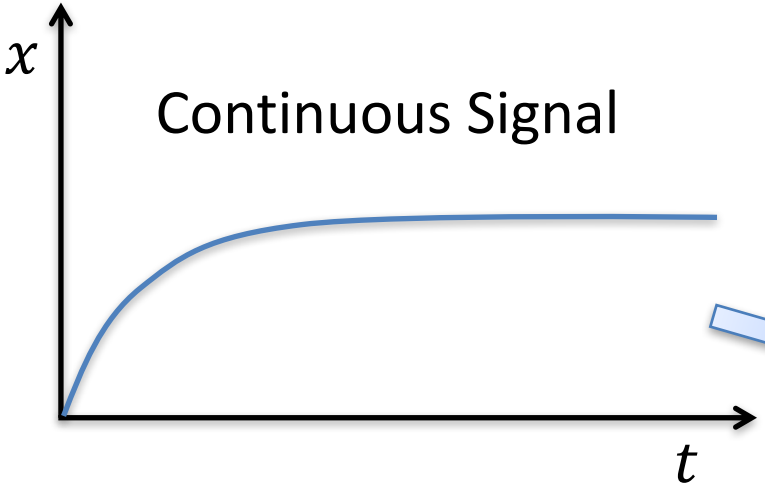
$$\theta_t = 22\ sec$$

$$\theta_d = 2\ sec$$

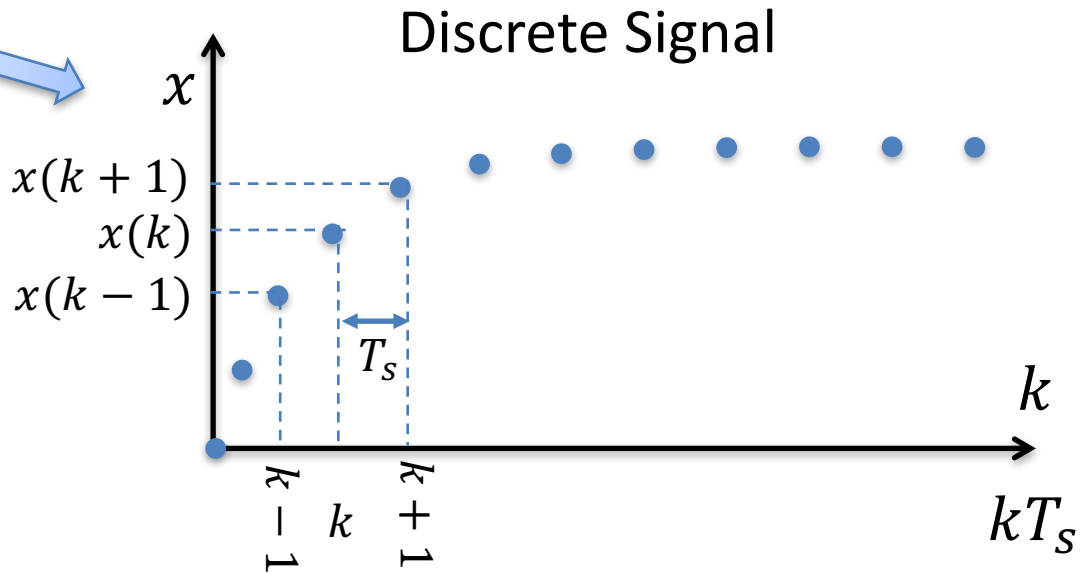
$$K_h = 3.5\ \frac{°C}{V}$$

$$T_{env} = 21.5\ °C$$

Continuous vs. Discrete Systems



A computer can only deal with discrete signals



T_s - Sampling Interval

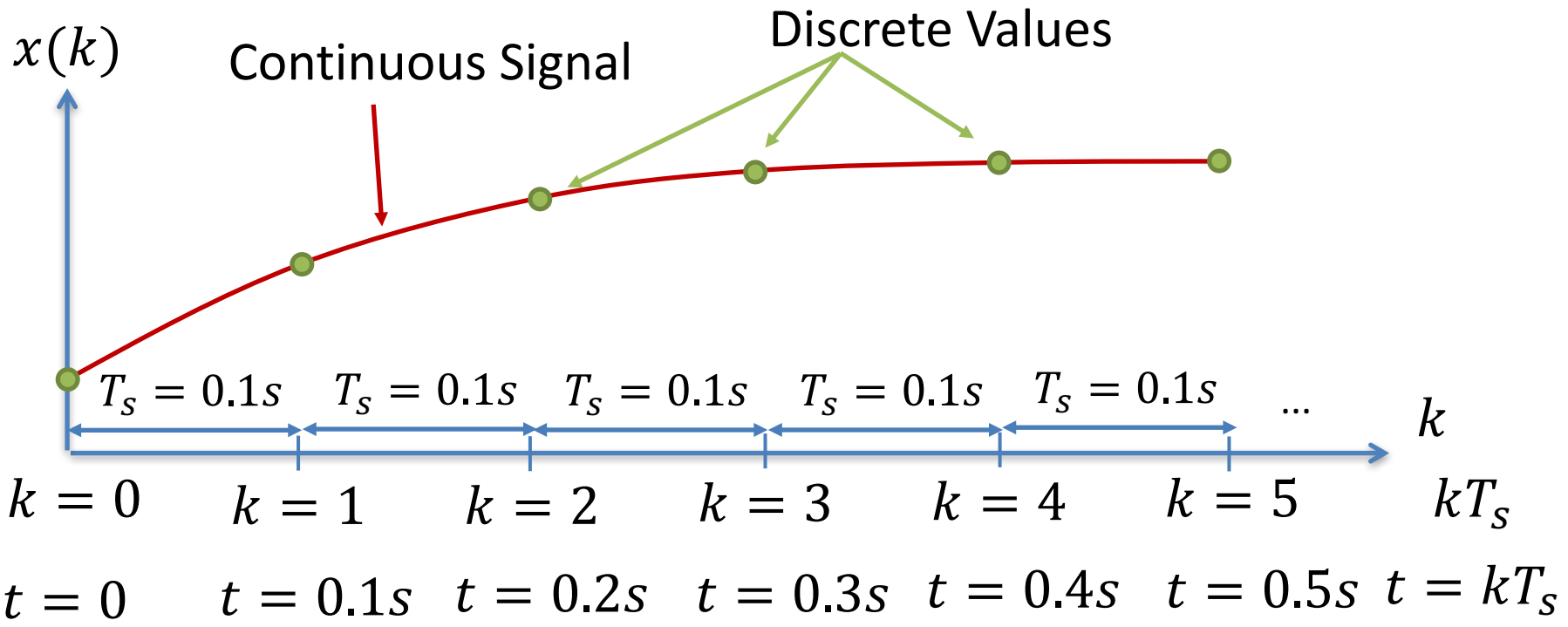
$x(k-1)$ - Previous Value

$x(k)$ - Current Value

$x(k-1)$ - Next Value

Continuous vs. Discrete Systems - Example

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

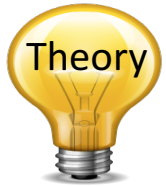
$x(k+1)$ - Next (future) value

The discrete Model will then be on the form:

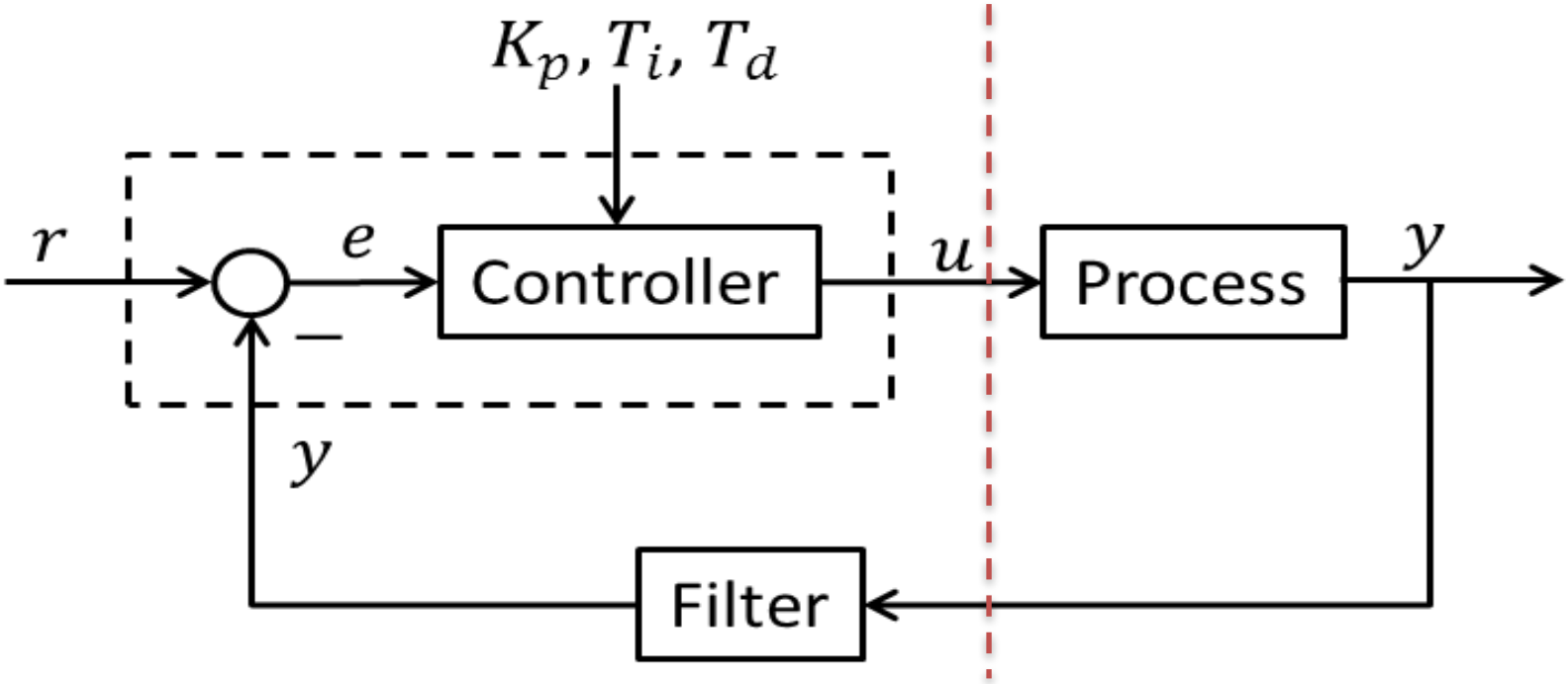
$$x(k+1) = x(k) + \ \ldots$$

We can then implement the discrete model in C#

Control System Example



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

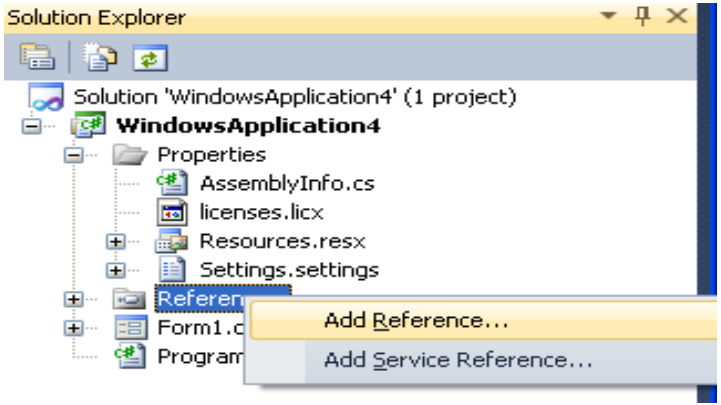


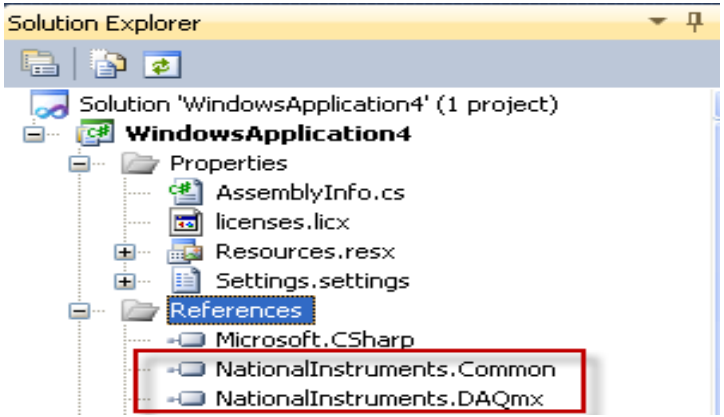
Visual Studio

Control System in C#

Hans-Petter Halvorsen, M.Sc.

Add References to the **DAQmx** Driver in Visual Studio



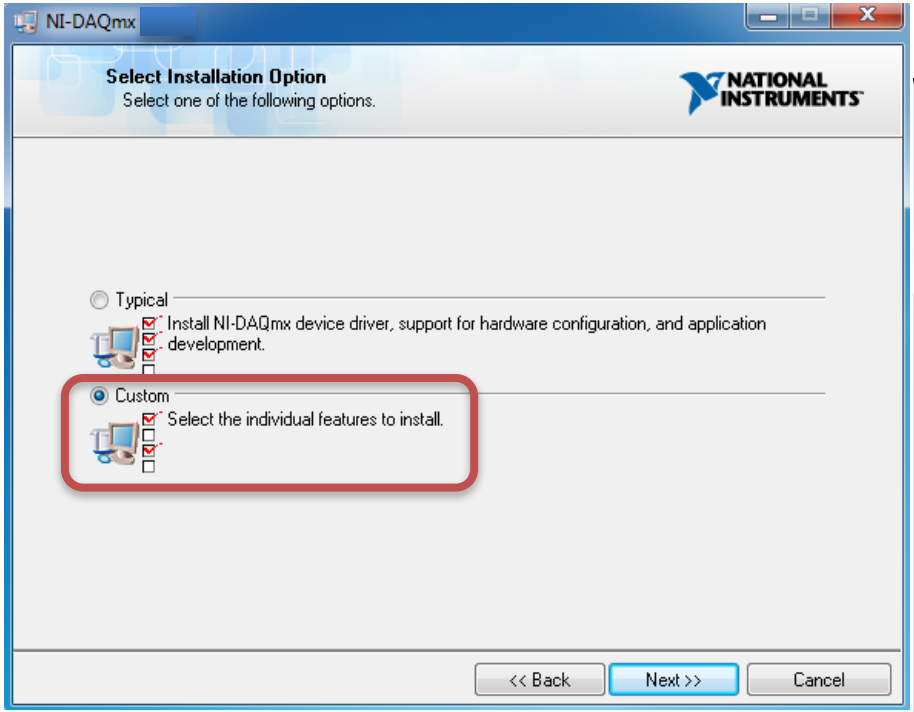



```
NationalInstruments.Common.dll
NationalInstruments.DAQmx.dll
```

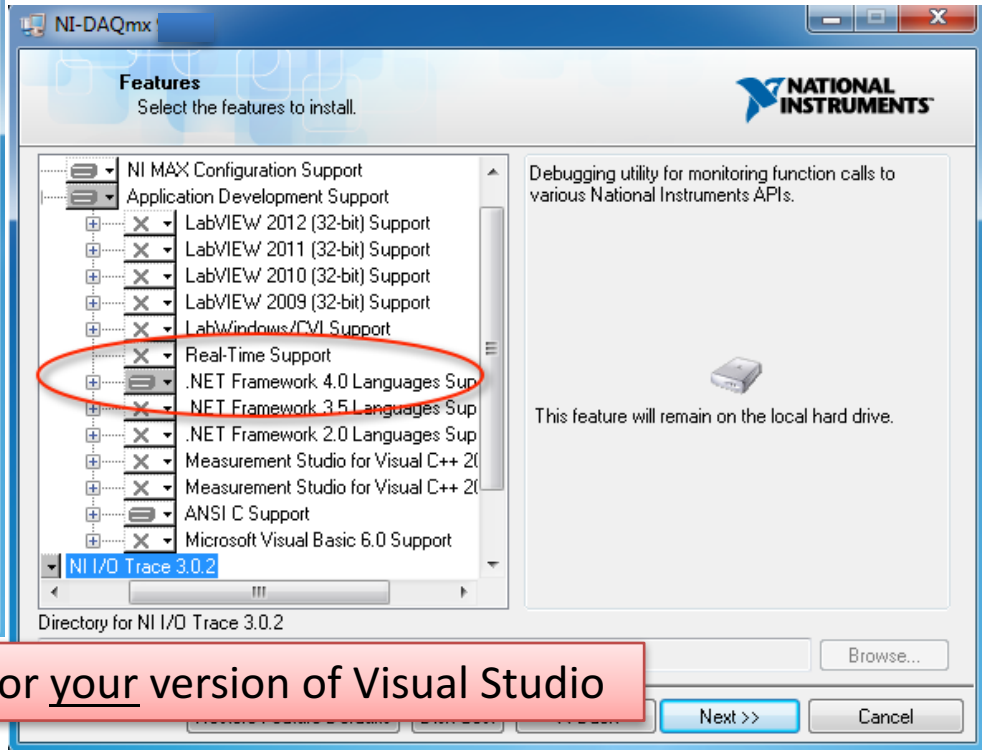
We also need to add the following Namespaces:

```
using NationalInstruments;
using NationalInstruments.DAQmx;
```

Support for .NET Framework with DAQmx Driver



Note! Also included with Measurement Studio



Make sure to add .NET Framework support for <u>your</u> version of Visual Studio

Simple **DAQ** in C# with DAQmx


```
private void btnGetAnalogIn_Click(object sender, EventArgs e)
        {
            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();

            txtAnalogIn.Text = analogDataIn.ToString();
        }
```

Analog In Example

Simple **DAQ** in C# with DAQmx

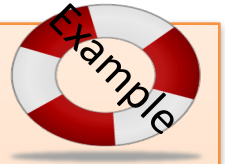

```
private void btnWriteAnalogOut_Click(object sender, EventArgs e)
        {
            Task analogOutTask = new Task();

            AOChannel myAOChannel;

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

            AnalogSingleChannelWriter writer = new
             AnalogSingleChannelWriter(analogOutTask.Stream);

            double analogDataOut;
            analogDataOut = Convert.ToDouble(txtAnalogOut.Text);

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

Analog Out Example

Write to **OPC** Server using Visual Studio

```
string opcUrl;
double opcValue;

opcUrl = "opc://localhost/Matrikon.OPC.Simulation/Bucket Brigade.Real4";

DataSocket dataSocket = new DataSocket();

if (dataSocket.IsConnected)
        dataSocket.Disconnect();

dataSocket.Connect(opcUrl, AccessMode.Write);

double opcValue = 12;
dataSocket.Data.Value = opcValue;
dataSocket.Update();
```

Make sure you have Measurement Studio Installed!!



We also need to add the following **Namespaces**:

```
using NationalInstruments;
using NationalInstruments.Net;
```

We also need to add **References** to the following **DLL** Files:

```
NationalInstruments.Common.dll
NationalInstruments.Net.dll
```

Note! This Code Snippet reads only one value once, you can use e.g. a **Timer** in order to read values at specific intervals

Visual Studio Editions

❑ I have Visual Studio 2013

- You can Install and use Measurements Studio 2015 without problems

❑ I have Visual Studio 2015

- Measurement Studio 2015 is designed to work with Visual Studio 2013 (and older editions). Therefore, Measurement Studio 2015 does not install shipping examples and does not integrate with Visual Studio 2015
- This means, if you install Measurement Studio 2015 with Visual Studio 2015, the Measurement Studio .NET controls are not in the Toolbox, and you do not have a Measurement Studio menu item in the Visual Studio 2015 toolbar.
- Follow these guidelines to do this manually:

http://home.hit.no/~hansha/documents/control/opc/resources/Using%20Measurement%20Studio%202015.pdf

Using Measurement Studio 2015 with Visual Studio 2015

Measurement Studio 2015 does not have integration features for Visual Studio 2015. If you install Measurement Studio 2015 with Visual Studio 2015, the Measurement Studio .NET controls are not in the Toolbox, you do not have a Measurement Studio menu item in the Visual Studio 2015 toolbar, and .licx will not be automatically generated. We have plans to make changes to Measurement Studio that will help us better keep up with new versions. Unfortunately, these changes are a quite a bit more costly than it would seem, particularly the Visual Studio Help integration, so these changes may not come soon.

This situation is the same as previous Measurement Studio software as in the following article 'Using Measurement Studio 2013 with Microsoft Visual Studio 2013': http://digital.ni.com/public.nsf/allkb/C51E3B38578FAD2786257C070069F386

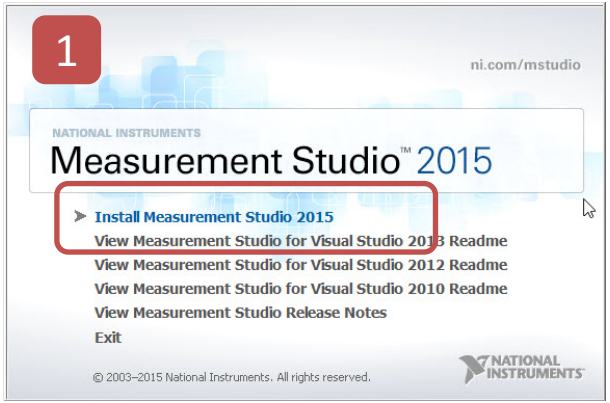
Visual Studio 2015 is not supported officially in Measurement Studio 2015; however, as in the above article, you can add the Measurement Studio .NET controls to the Toolbox manually and can create .licx files manually. I have attached a Help Document on this topic so you can refer to the Adding Measurement Studio 2015 User Interface Controls to the Toolbox section for more information on How to Add controls. This section also describes how these controls are licensed. The following sections discuss additional topics to consider when using Measurement Studio 2015 with Visual Studio 2015.

Rebecca Costin
National Instruments
Applications Engineering
www.ni.com/support

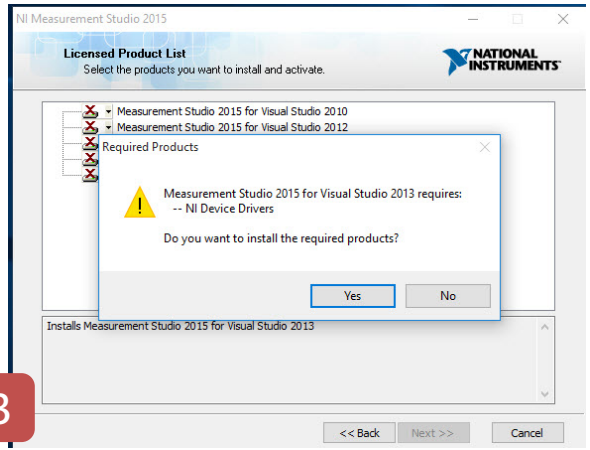
http://home.hit.no/~hansha/documents/control/opc/resources/Using%20Measurement%20Studio%202015.pdf

Measurement Studio 2015 Installation

1

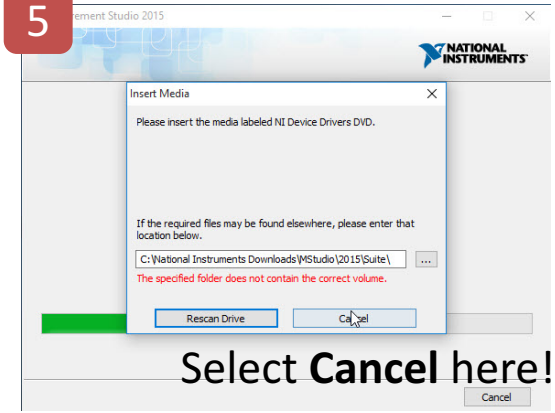


2 Select Measurement Studio 2015 for Visual Studio 2013



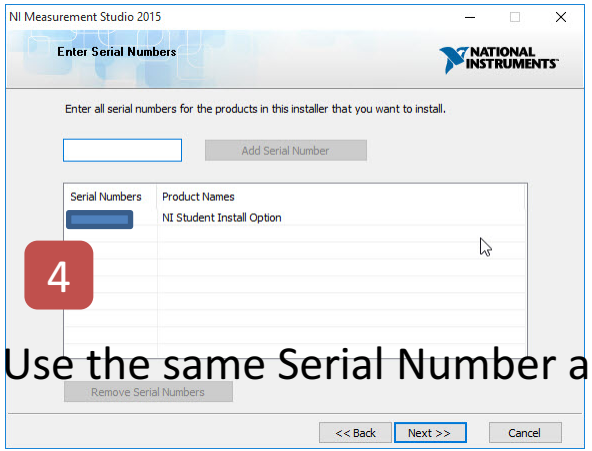
3

5



Select **Cancel** here!

You don need the NI Device Drivers – but still you need to select “Yes”!
If not the installation will not start!

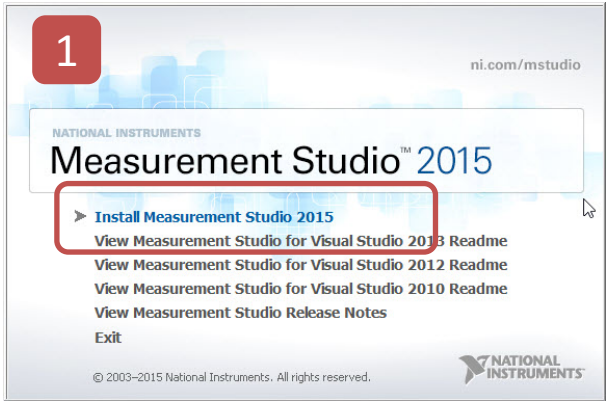


4

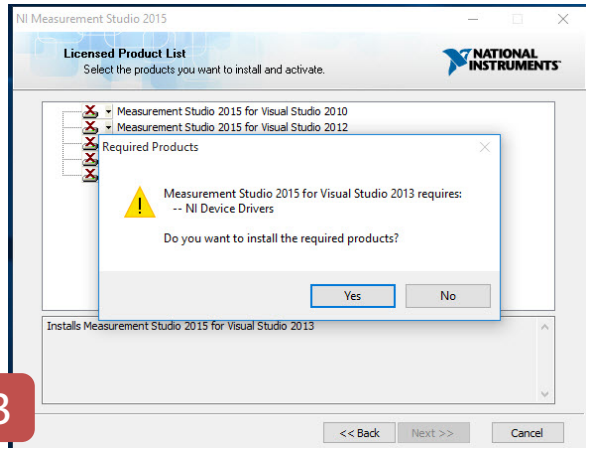
Use the same Serial Number as you use for LabVIEW

Measurement Studio 2015 Installation

1

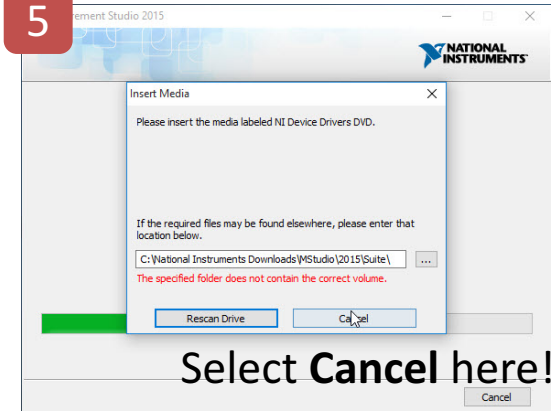


2 Select Measurement Studio 2015 for Visual Studio 2013



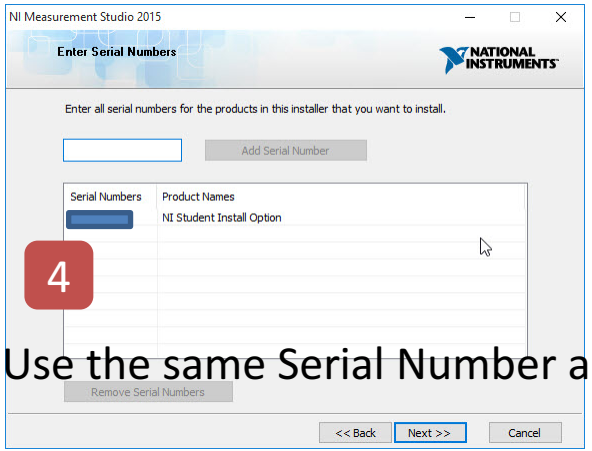
3

5



Select **Cancel** here!

You don need the NI Device Drivers – but still you need to select “Yes”!
If not the installation will not start!



4

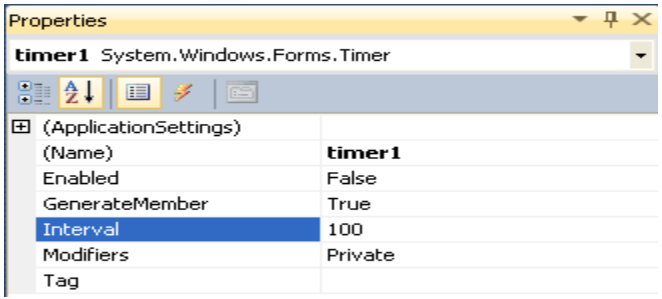
Use the same Serial Number as you use for LabVIEW

Timer

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

1 Select the “Timer” component in the Toolbox

Properties: 3



You may specify the Timer Interval in the Properties Window

2 **Initialization:**

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

Double-click on the Timer object in order to create the Event

4 **Timer Event:**

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

Structure your Code properly!! Define Classes and Methods which you can use here

Visual Studio

Discrete PI (D) Controller

Hans-Petter Halvorsen, M.Sc.

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

Students:
Create a PI(D) Controller in C#

Visual Studio

Simple Discrete **PI Controller** – C#

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

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

            e = r - y;
            u = Kp * e + (Kp / Ti) * z;
            z = z + Ts * e;
            return u;
        }
}
```

Note! This is just a simple Example

Visual Studio

Discrete Lowpass Filter

Hans-Petter Halvorsen, M.Sc.

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:

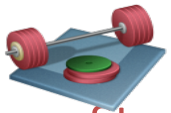
$$y_k = (1 - a) y_{k-1} + a u_k$$

Filter output

Noisy input signal

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

This algorithm can be easly implemented in a Programming language



Students: Create a Lowpass Filter in C#

Discrete **Lowpass** Filter – C#

```
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

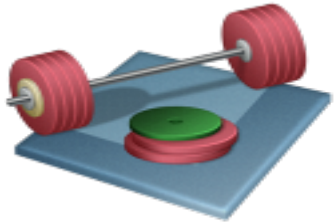
Congratulations! - You are finished with the Task

Task D

Fuji PXG5 PID Controller

Hans-Petter Halvorsen, M.Sc.

Fuji PXG5 PID



- Control the Process using a Fuji PXG5 PID Controller.
- Compare results with your PI(D) Controller created from scratch
- Make sure the Data are stored in the Database (via OPC)
  - This means you should connect the Fuji PID controller both to the Air Heater and a DAQ device
- The Fuji Controller should first be tested using **HIL Simulation & Testing** in C#

HIL Simulation & Testing Setup

PID Control

Analog Out
(Process Value)
1-5V

AO0

GND

Note!!

AI0+

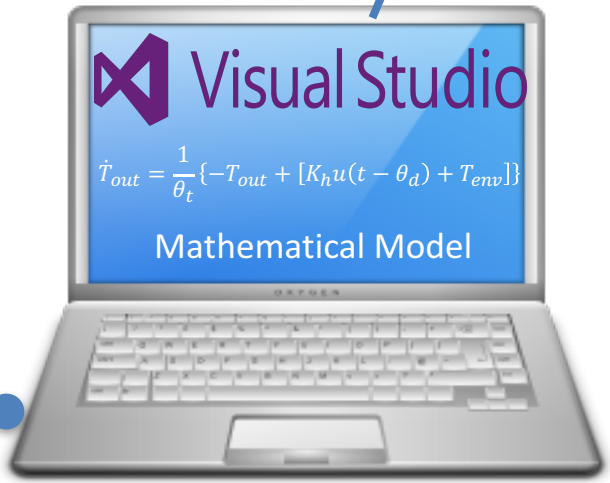
AI0-

Analog In
(Control Signal)
0-5V

I/O Module
(USB-6008)

USB

OPC Server



Model of Process (Air Heater)
Computer (with Visual Studio/C#)

With this setup you can Test the Fuji PID using HIL Simulation and Testing (+ Plot and send Data to OPC)

Fuju PXG5 + Real Air Heater + PC for Monitoring

Industrial PID Controller

OPC Server

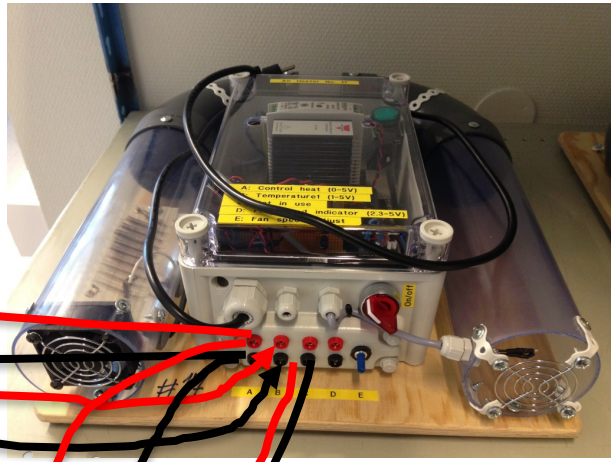
Trending/Monitoring the Process Value and Control Signal on the PC

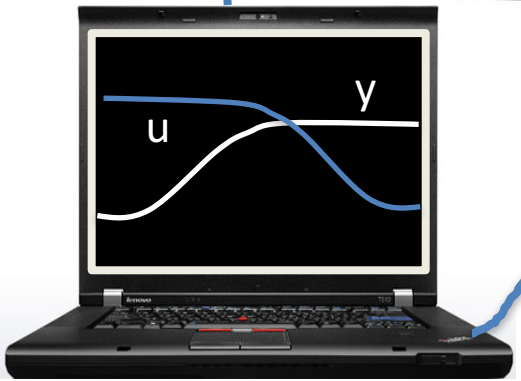
Process Value

1-5V

0-5V

Control Signal





PC + C# App

USB

Process Value

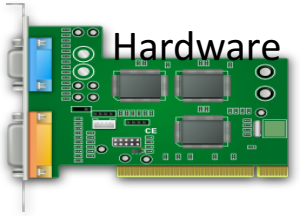
1-5V

Control Signal

0-5V

With this setup you can Monitor (Plot and send Data to OPC) the Process Value and Control Signal on your PC

PXG5 PID Controller



How it works (Main Features):

Display

C1/C2 Lamp
Displays the condition of the control output. Lights ON at 100% output and goes out at 0% output. For values between 0% and 100%, the output is indicated by the length of time the lamp flickers.When acting as a valve control, the C1 lamp will light with OPEN output, and the C2 lamp will flickers with CLOSE output.

DO1/2/3 Lamp
Lights ON when there is digital output is on state (DO1, DO2, DO3). The lamp flickers when delay behavior is on.

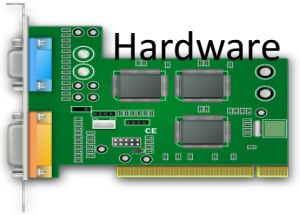
PV Display
Displays the measurement value (PV). Displays the name of the parameter when setting parameters.

SV Display
Displays the setting value (SV). Also can display the output value during manual mode. Displays the parameter setting value when setting parameters. Displays "rEn" during remote SV operation, and "SoFt" and set value alternately during soft start.

SV Lamp
Lights when displaying the setting value (SV). Goes out when displaying the manual output value.
The lamp flickers while performing ramp soak or lamp SV operations.

MAN/AT/SELF Lamp
Normally lights up during manual mode and blinks during auto-tuning or self-tuning.

PXG5 - Configuration



How-To Change Setpoint:

- Changing SV (set values)

1. **Change the display to PV/SV display (shown when you turn on the power and the SV lamp is lit).**
2. **Change the SV with the ▲ ▼ keys.**
3. **Press the SEL key to save the values.**
   (The value will be automatically saved after 3 seconds even if a key is not pressed.)

How-To Change Parameters:

5 – 5 / Setting Parameters

The following explains how to set the parameters.

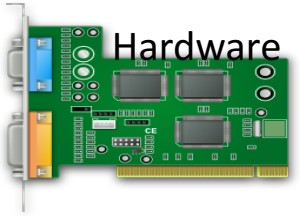
1. **Press and hold the SEL key in operation mode, or manual mode.**
   This switches you to the monitor mode Mv1.
2. **Press and hold the SEL key in monitor mode**
   This switches you to the channel menu of setup mode.
3. **Choose the channel with the ▲ ▼ keys, then press and hold the SEL key.**
   This switches you to the parameter menu.
4. **Choose the parameter with the ▲ ▼ keys, then press the SEL key.**
   The set value flickers.
5. **Choose the parameter with the ▲ ▼ keys, then press the SEL key.**
   The set value is fixed.

- No matter where you are in monitor or setup mode, pressing the AUM key returns you to operation mode. When setting the parameters in manual mode, pressing the key holds manual mode and returns you to operation mode.

Channel 1: Auto-tuning

Channel 2: PID Parameters

PXG5 - Configuration



Recommended Channel Settings:

Channel 1:

MAn = oFF

rEM = LoCl

AT = oFF

Channel 2:

SvL = 20

Svh = 50

Channel 6:

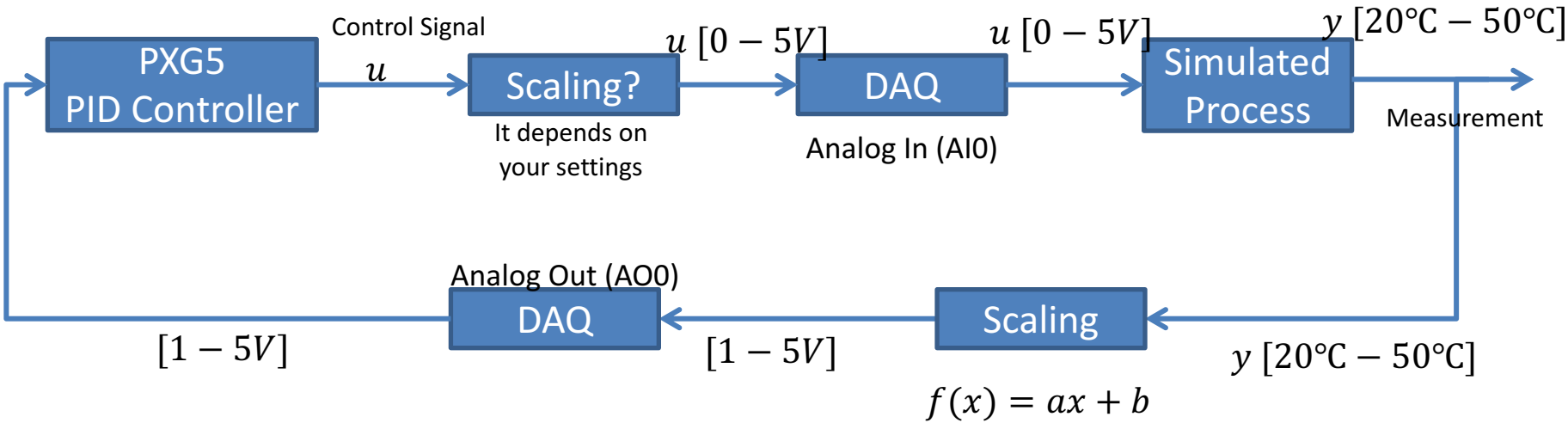
Pvb = 20

PvF = 50

Pvd = 0

C1r = 0-20mA

HIL Simulation using PXG5 PID Controller



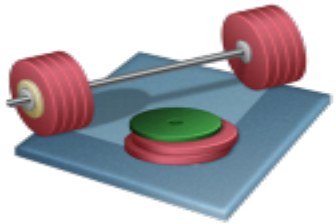
Congratulations! - You are finished with the Task

Task E

Datalogging System

Hans-Petter Halvorsen, M.Sc.

Datalogging System



- Create an application in C# that reads the values from the **OPC** Server and store them in the **SQL Server**.
- You should create a **Stored Procedure** that saves the data into the Database. This Stored procedure should be used from the C# Application

Read from **OPC** Server using Visual Studio

Make sure you have Measurement Studio Installed!!

```
string opcUrl;
double opcValue;

opcUrl = "opc://localhost/Matrikon.OPC.Simulation/Bucket Brigade.Real4";

DataSocket dataSocket = new DataSocket();

if (dataSocket.IsConnected)
        dataSocket.Disconnect();

dataSocket.Connect(opcUrl, AccessMode.Read);
dataSocket.Update();
opcValue = Convert.ToDouble(dataSocket.Data.Value);
```

We also need to add the following **Namespaces**:

```
using NationalInstruments;
using NationalInstruments.Net;
```

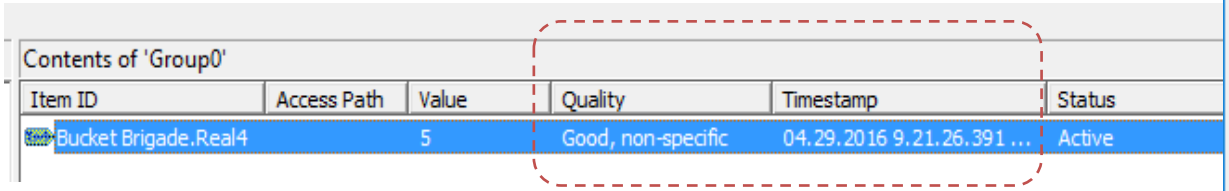
We also need to add **References** to the following **DLL** Files:

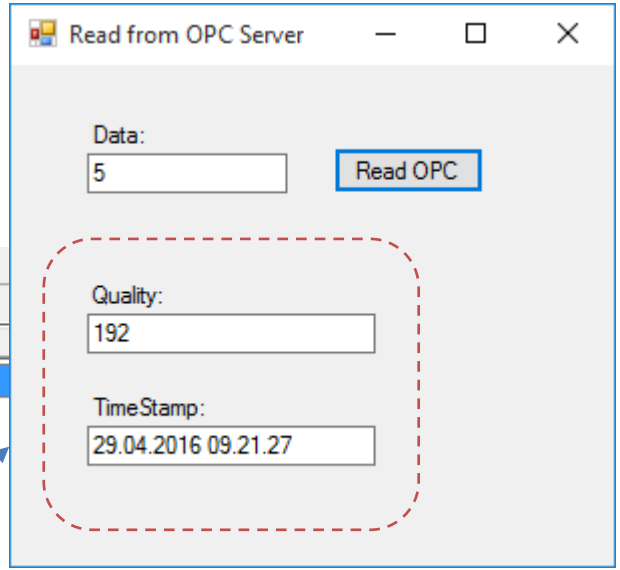
```
NationalInstruments.Common.dll
NationalInstruments.Net.dll
```

Note! This Code Snippet reads only one value once, you can use e.g. a **Timer** in order to read values at specific intervals

Get OPC Attributes

Matrikon OPC Server:





Visual Studio/C# Code:

```
...

 dataSocket.Update();

txtReadOpcValue.Text = dataSocket.Data.Value.ToString();

txtAttributeQuality.Text = dataSocket.Data.Attributes["Quality"].Value.ToString();
txtAttributeTimeStamp.Text = dataSocket.Data.Attributes["Timestamp"].Value.ToString();
```

Stored Procedure Example

Create Stored Procedure:

```
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;
            }
}
```

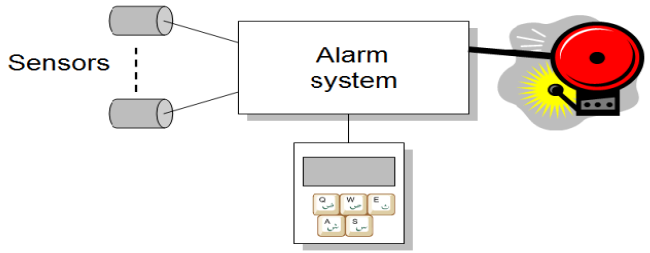
It is recommended to create and use a **Stored Procedure**.
It is also recommended that the Connection String is stored in **App.config**

Congratulations! - You are finished with the Task

Task F

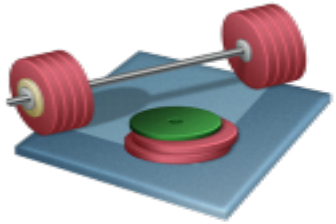
Alarm System

Alarm Generation and Alarm Monitoring

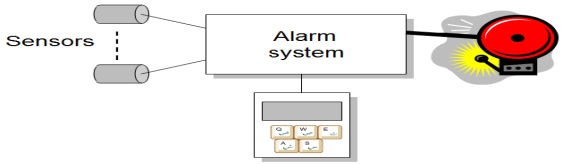


Hans-Petter Halvorsen, M.Sc.

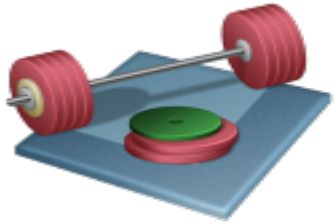
Alarm System



- The Alarm System should check for Alarms and saves the Alarm information in your Database.
- In addition you should have a User Interface that shows the Alarms (Alarm List).
- You could implement Alarm Logging in your SCADA system by using a Database **Trigger** on the table that stores the Tag Data.



Alarm System



- Create an **Alarm Application** (C# WinForm App, ASP.NET App, WPF or UWP App) that shows all the Alarms in the system.
- 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** should be used in order to generate Alarms in the Database
- A Database **View** should be created and used to retrieve Alarm Data from the Database

Create the Trigger:

Trigger Example

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)

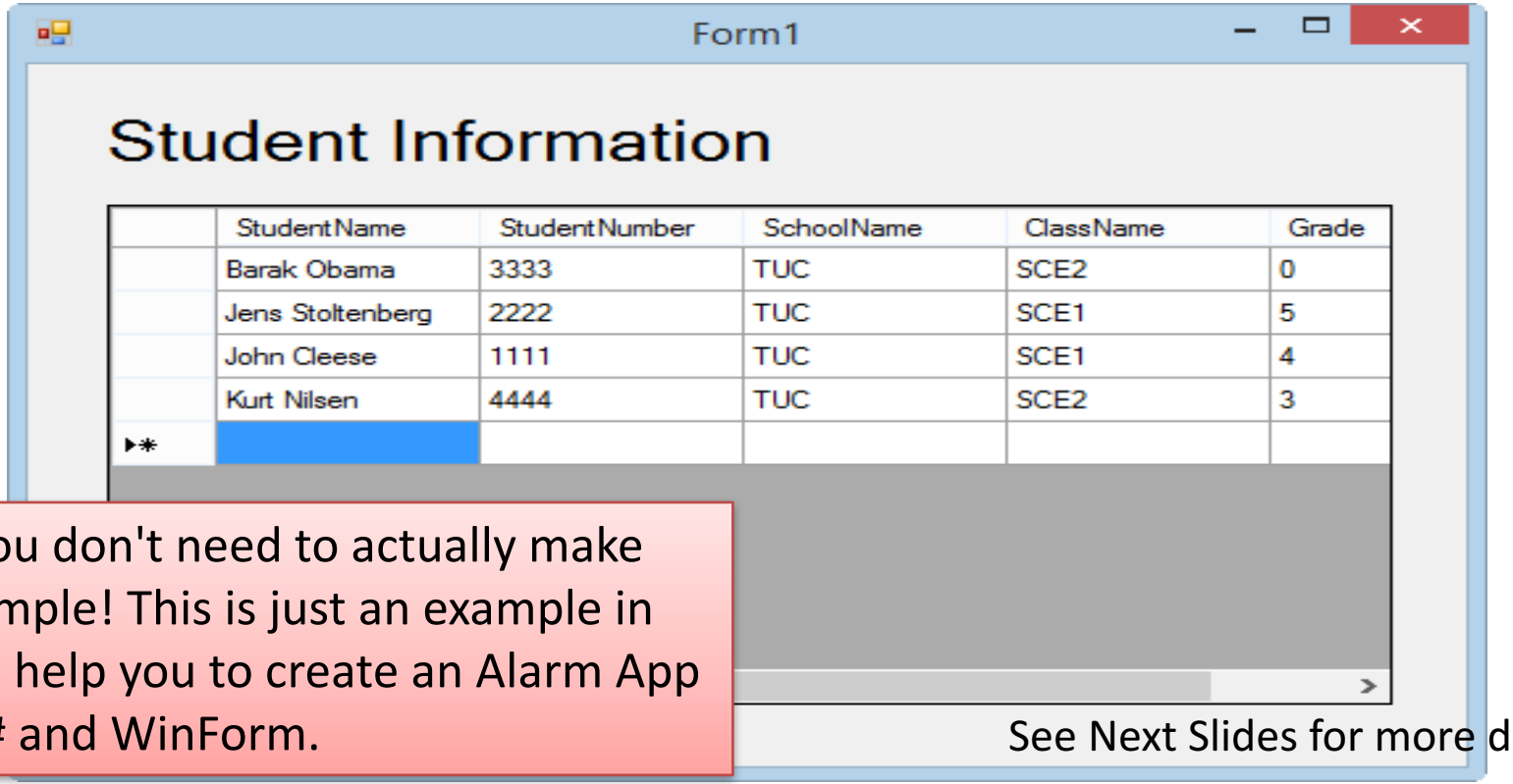
Note! "INSERTED" is a temporarily table containing the latest inserted data, and it is very handy to use inside a trigger

WinForm C# Example

(or you can create an ASP:NET Web App)

Hans-Petter Halvorsen, M.Sc.

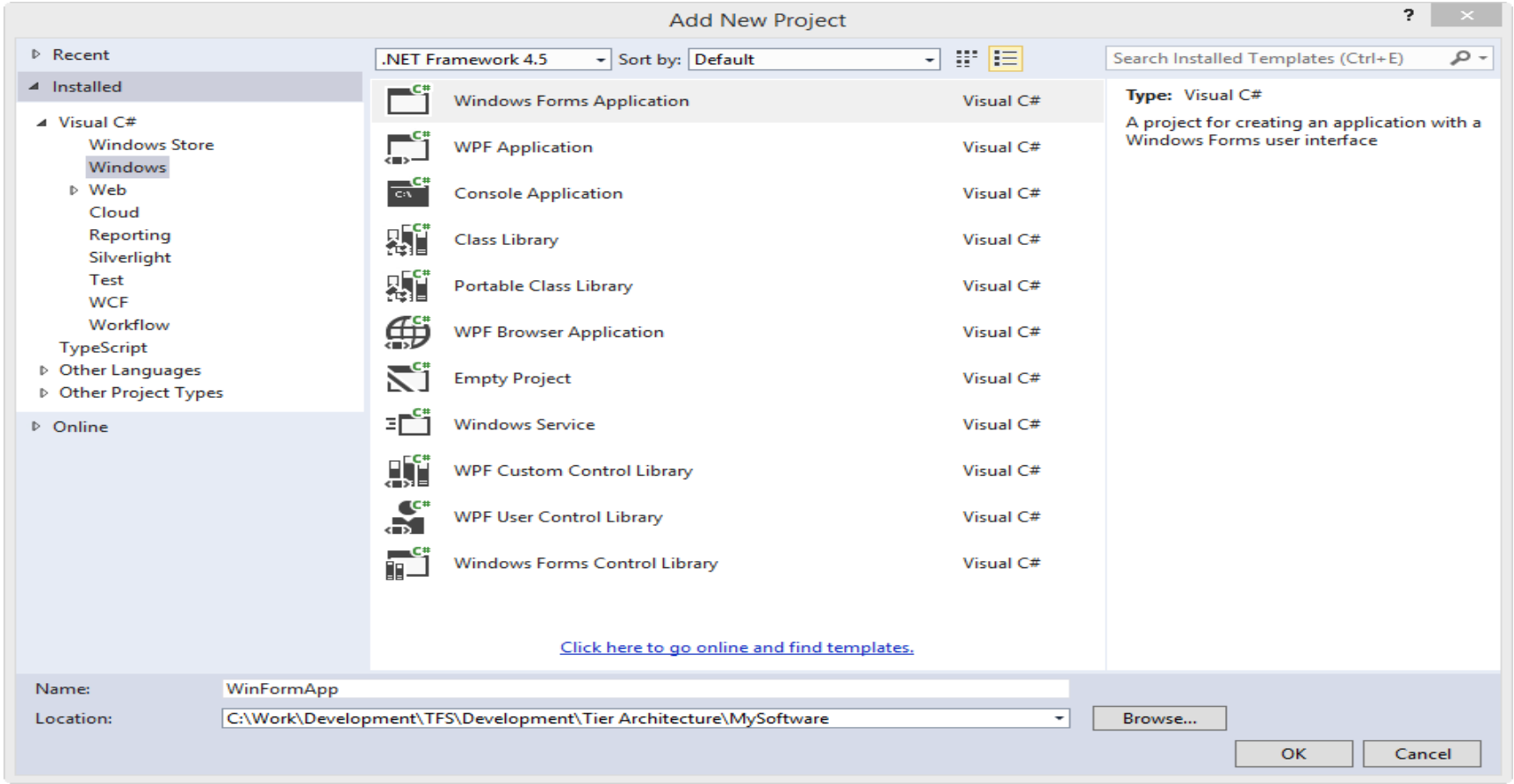
This Example shows how to create a WinForm App that gets Data from a Database into a DataGridView



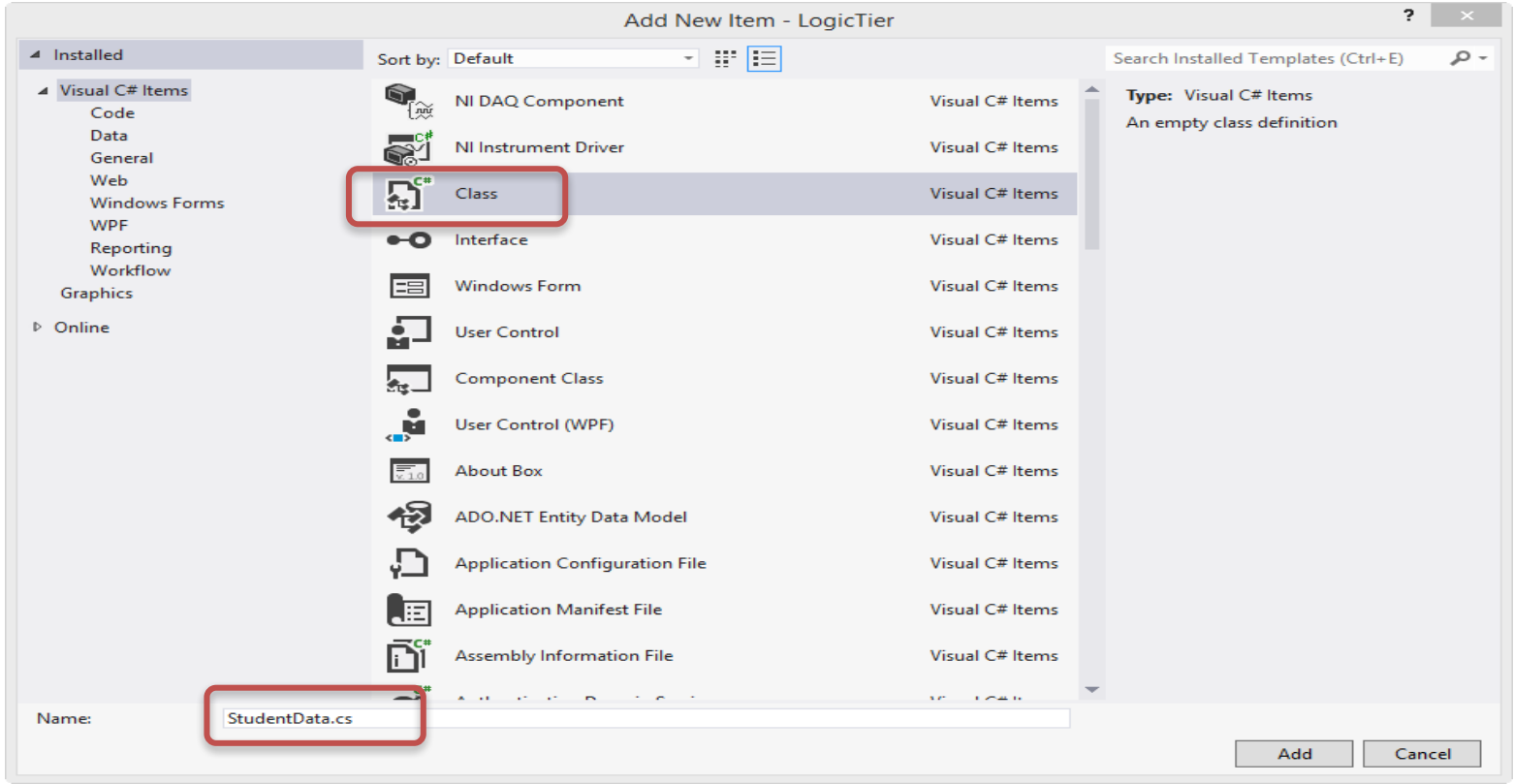
Note! You don't need to actually make this example! This is just an example in order to help you to create an Alarm App using C# and WinForm.

See Next Slides for more details

Step 1: Add a **WinForm Project**



Step 2a: Add a New **Class** to the Project



"StudentData.cs"

Step 2b: Create the **Code**:

StudentData.cs

Tuc.School.LogicTier.StudentData | GetStudentDB(string connectionString)

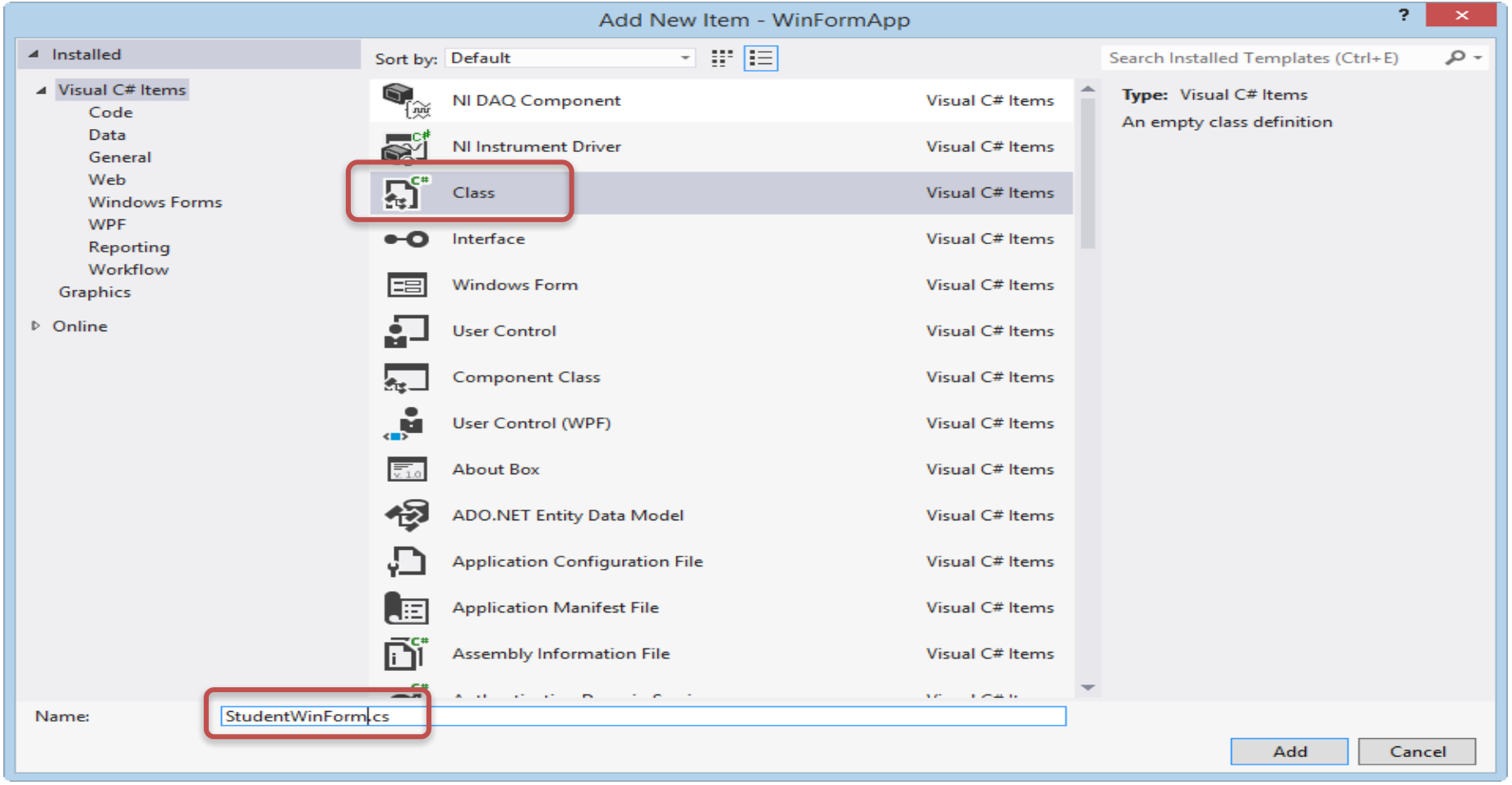
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using System.Data.SqlClient;
using System.Data.SqlTypes;
using System.Data;

namespace Tuc.School.LogicTier
{
    public class StudentData
    {

        public DataSet GetStudentDB(string connectionString)
        {

            string selectSQL = "select StudentName, StudentNumber, SchoolName, ClassName, Grade from StudentData order by StudentName";

            // Define the ADO.NET objects.
            SqlConnection con = new SqlConnection(connectionString);

            SqlDataAdapter da = new SqlDataAdapter(selectSQL, con);

            DataSet ds = new DataSet();
            da.Fill(ds);

            return ds;

        }

    }
}
```

SQL Query

A View (created inside SQL Server) that collects data from several tables

Improvements: Use Try... Catch ...

Step 3a: Add a New **Class** (“StudentWinForm.cs”)



Step 3b: Add **Code** in Class

StudentWinForm.cs

Tuc.School.WinFormApp.StudentWinForm

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using System.Data;

using Tuc.School.LogicTier;

namespace Tuc.School.WinFormApp
{
    class StudentWinForm
    {

        public DataSet GetStudent(string connectionString)
        {
            StudentData studentData = new StudentData();

            return studentData.GetStudentDB(connectionString);
        }

    }
}
```

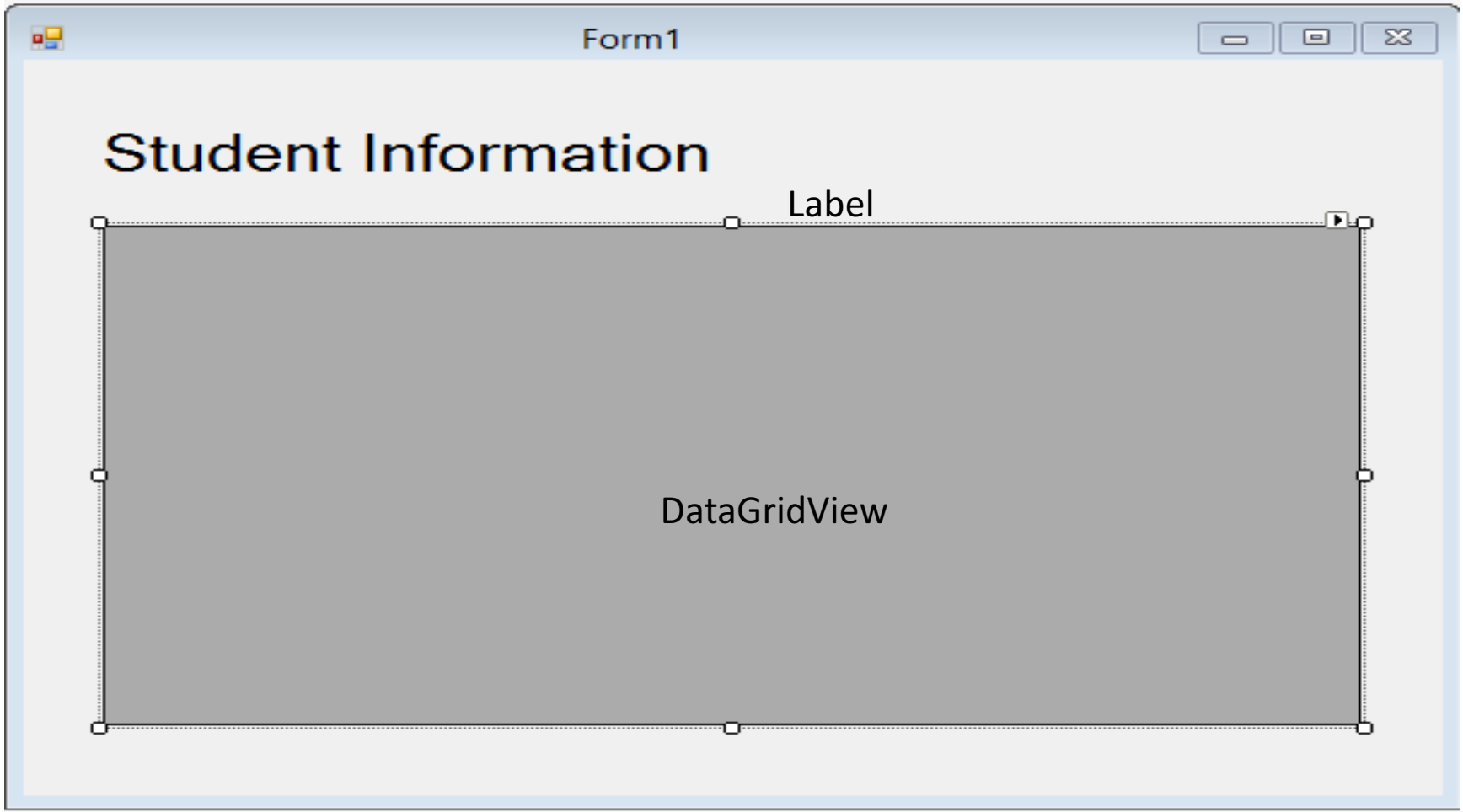
Add a Reference to your Class created in Step 2b

Code for Class “StudentWinForm.cs”

```
using System.Data;

using Tuc.School.LogicTier;

namespace Tuc.School.WinFormApp
{
    class StudentWinForm
    {
        public DataSet GetStudent(string connectionString)
        {
            StudentData studentData = new StudentData();

            return studentData.GetStudentDB(connectionString);
        }
    }
}
```

Since we are using the DataSet Class

Step 4a: Create Form



Step 4b: Create Form Code

Form1.cs | Form1.cs [Design]

WinFormApp.Form1 | connectionString

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

using System.Configuration;

using Tuc.School.WinFormApp;


namespace WinFormApp
{
    public partial class Form1 : Form
    {

        private string connectionString = ConfigurationManager.ConnectionStrings["SCHOOLConnectionString"].ConnectionString;

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            FillStudentGrid();

        }


        private void FillStudentGrid()
        {

            DataSet ds = new DataSet();

            StudentWinForm studentList = new StudentWinForm();

            ds = studentList.GetStudent(connectionString);

            dataGridViewStudentInformation.DataSource = ds.Tables[0];

        }
    }
}
```

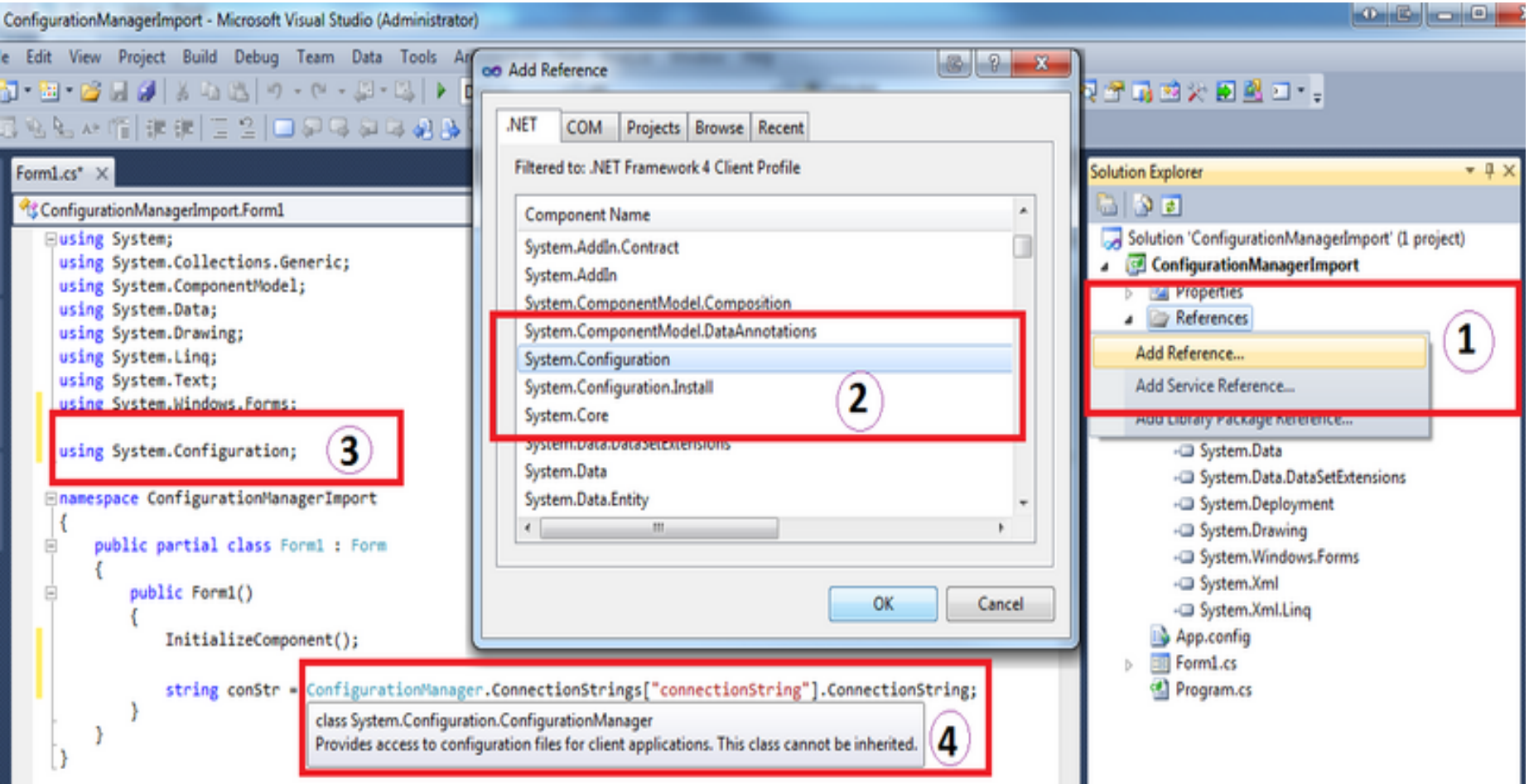
WinForm Code

```
using System.Configuration;
using Tuc.School.WinFormApp;

namespace WinFormApp
{
    public partial class Form1 : Form
    {

        private string connectionString =
                ConfigurationManager.ConnectionStrings["SCHOOLConnectionString"].ConnectionString;

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            FillStudentGrid();

        }

        private void FillStudentGrid()
        {

            DataSet ds = new DataSet();

            StudentWinForm studentList = new StudentWinForm();

            ds = studentList.GetStudent(connectionString);

            dataGridViewStudentInformation.DataSource = ds.Tables[0];

        }
    }
}
```

Note!

ConnectionString is stored in App.config

Note! Add “System.Configuration” Reference



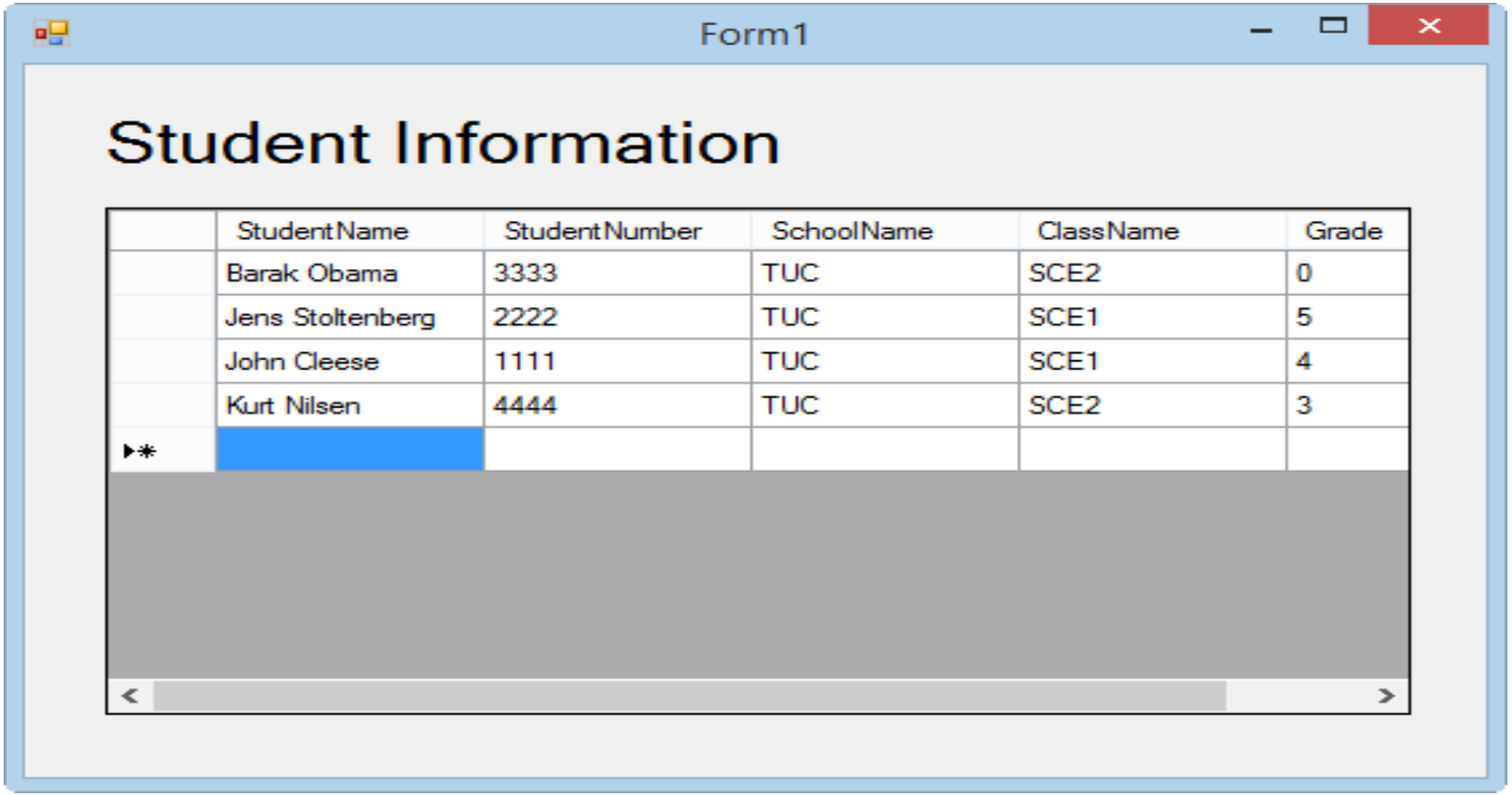
Step 5: Create DB ConnectionString in **App.config**

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>

    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
    </startup>


  <connectionStrings>
    <add name="SCHOOLConnectionString" connectionString="Data Source=macwin8;Initial Catalog=SCHOOL;Persist Security
Info=True;User ID=sa;Password=xxxxxx"
      providerName="System.Data.SqlClient" />
  </connectionStrings>
```

App.config

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>


    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
    </startup>


  <connectionStrings>
    <add name="SCHOOLConnectionString" connectionString="Data Source=macwin8;Initial Catalog=SCHOOL;Persist Security Info=True;User ID=sa;Passwor
      providerName="System.Data.SqlClient" />
  </connectionStrings>



</configuration>
```

Test it



It works!!!

Congratulations! - You are finished with the Task

Task G

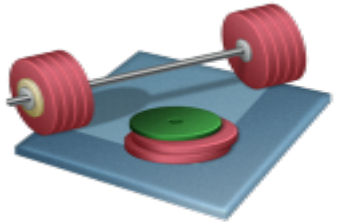
Network

Testing the Final SCADA System

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

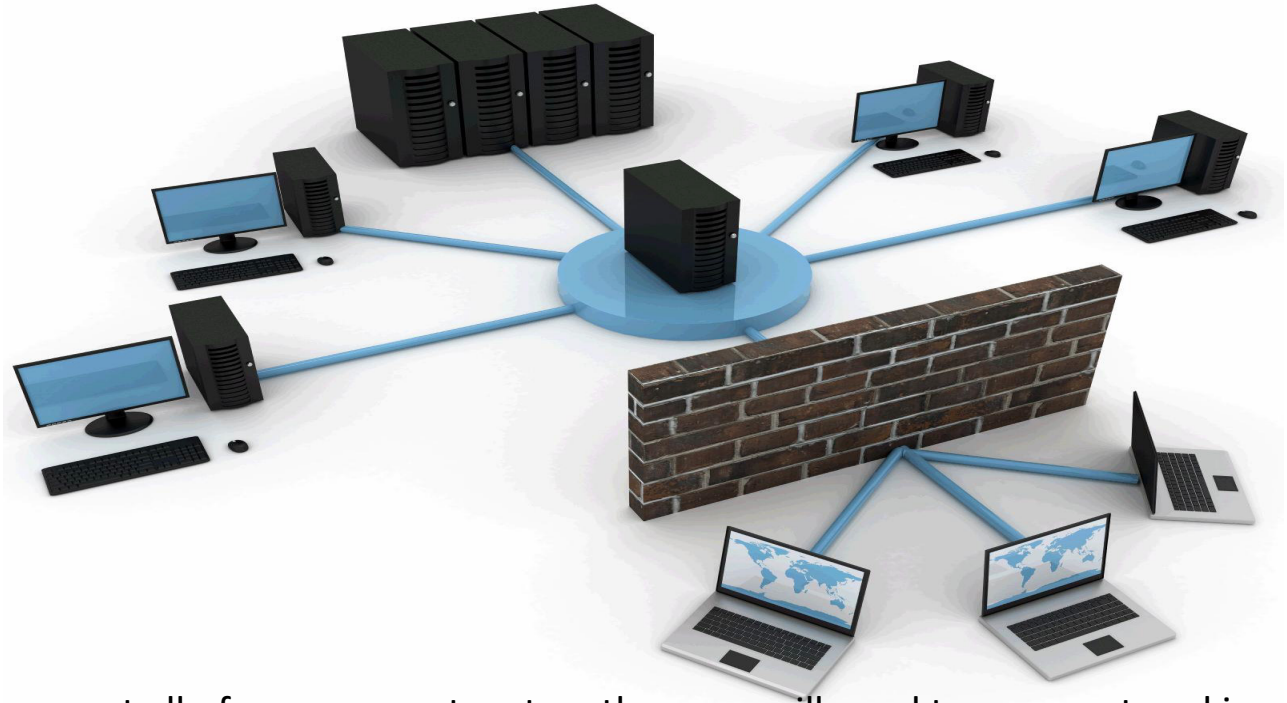
Hans-Petter Halvorsen, M.Sc.

Network



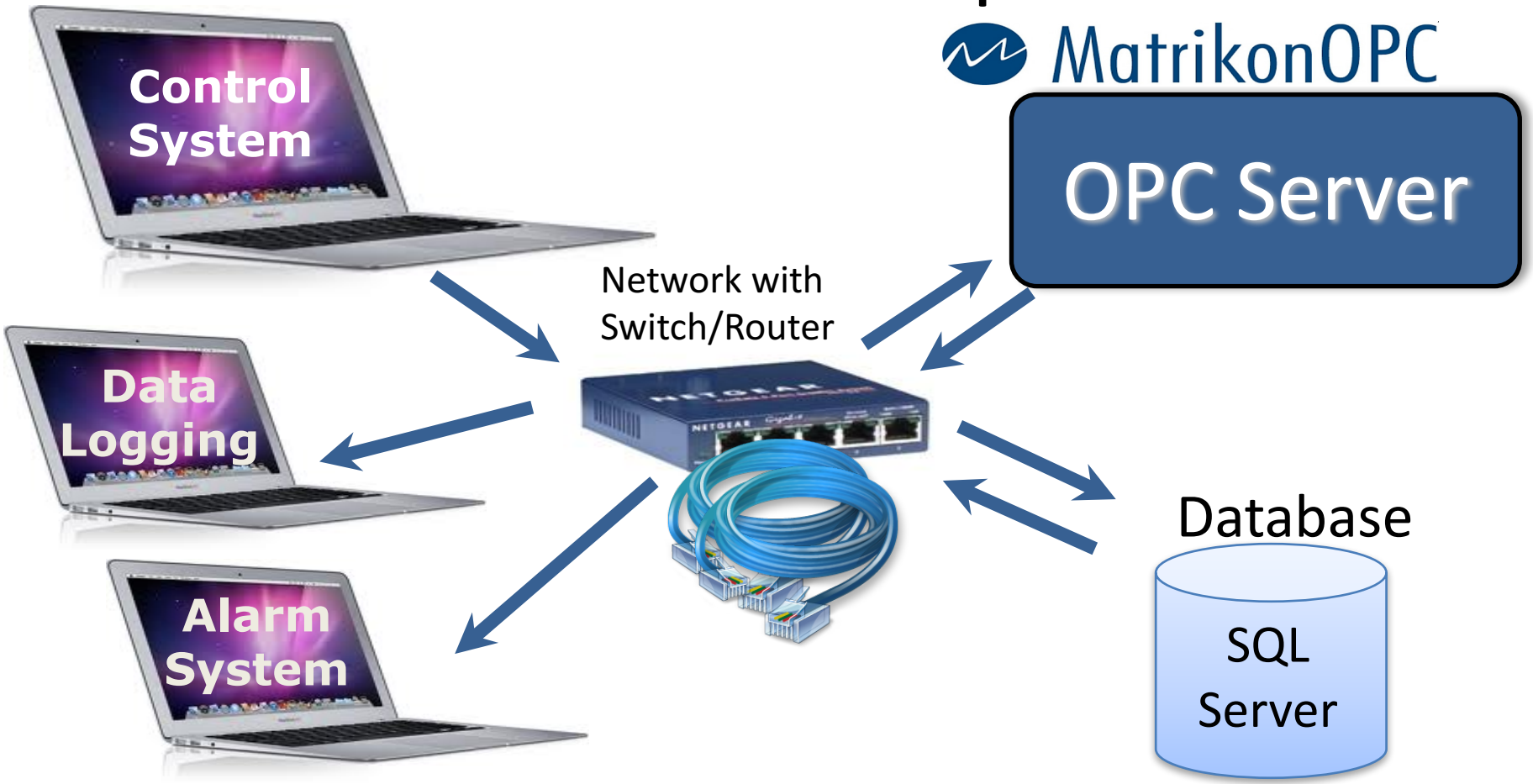
- The SCADA system should be **distributed**, meaning that the different components could be located on different computers in a network
- Connect the different components (Database, OPC Server, ...) together in a network using a switch or a router – and/or the built-in lab network we have in room C-222/C-139a
- Use your personal computers that are available within the group

Network



If you want to connect all of your computers together, you will need to use a networking device. While routers are the most common networking option, a network switch is a less expensive choice that can work just as well. Although a network switch lacks the built-in firewall of a router, it is a more than adequate choice for most small-scale networks in a home or office setting.

SCADA Example



OPC in Network

- OPC uses COM/DCOM -> Complicated to make it work in a modern Network!!
- Solution: Use an **OPC Tunneller Software**
  - OPC Tunneller from MatrikonOPC (30 days free trial)
  - Cogent DataHub Tunnelling Software (The Trial software works only 1 hour, then it needs to be restarted)

[Figure: http://www.cogentdatahub.com]

Congratulations! - You are finished with the Task

Congratulations! - You are finished with all the Tasks in the Assignment!

Hans-Petter Halvorsen, M.Sc.

University College of Southeast Norway

www.usn.no

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

Blog: http://home.hit.no/~hansha/