https://www.halvorsen.blog

Industry 4.0 and Cyber Security

Hans-Petter Halvorsen

Table of Contents

- Introduction
- Industry 4.0
  - DAQ
  - OPC
- Control Engineering
  - Air Heater System
  - Frequency Response
  - Stability Analysis
  - PID Controller
- Database Systems
- ASP.NET Core
- Microsoft Azure
- Cyber Security
  - Authentication
  - SQL Injection
  - Risk Analysis
  - Cyber Security Test Tools

Introduction

- Cloud services and IoT solutions are becoming increasingly popular.
- Even the industry embrace IoT as Industrial Internet of Things (IIoT), which is part of the next generation Automation Systems.
- Industry 4.0 is the new buzzword for the combination of industry, automation and the current Internet of Things (IoT) technology.
- We will focus on Web Technology and modern Cloud Platforms like Microsoft Azure.

Topics

- IIoT and Industry 4.0 (The Next Generation Industry)
- Control Engineering
- OPC; OPC UA is the Industry 4.0 implementation of OPC
- Database Systems; SQL Server
- Web Technology and ASP.NET Core
- Microsoft Azure (Cloud Platform)
- Cyber Security

Delivery

- Control System Design: Perform **Frequency Response** and Stability Analysis on the Control System
- Create a **Control System** in either LabVIEW, C# or Python. Implement a discrete PID controller from scratch.
- Start by creating a Simulator.
- When the simulator is working properly, start using the real Air Heater system.
- Store data in a local **SQL Server**.
- OPC, preferably **OPC UA** should be used.
- Create an **ASP.NET Core** Web Application for Monitoring your Data.
- **Microsoft Azure**: deploy your system to Microsoft Azure, i.e., the database and the web application should be running in Microsoft Azure
- **Cyber Security**: SQL Injection: Make sure your system is secure when it comes to SQL Injection issues. Create Login functionality

For more details, see the web site

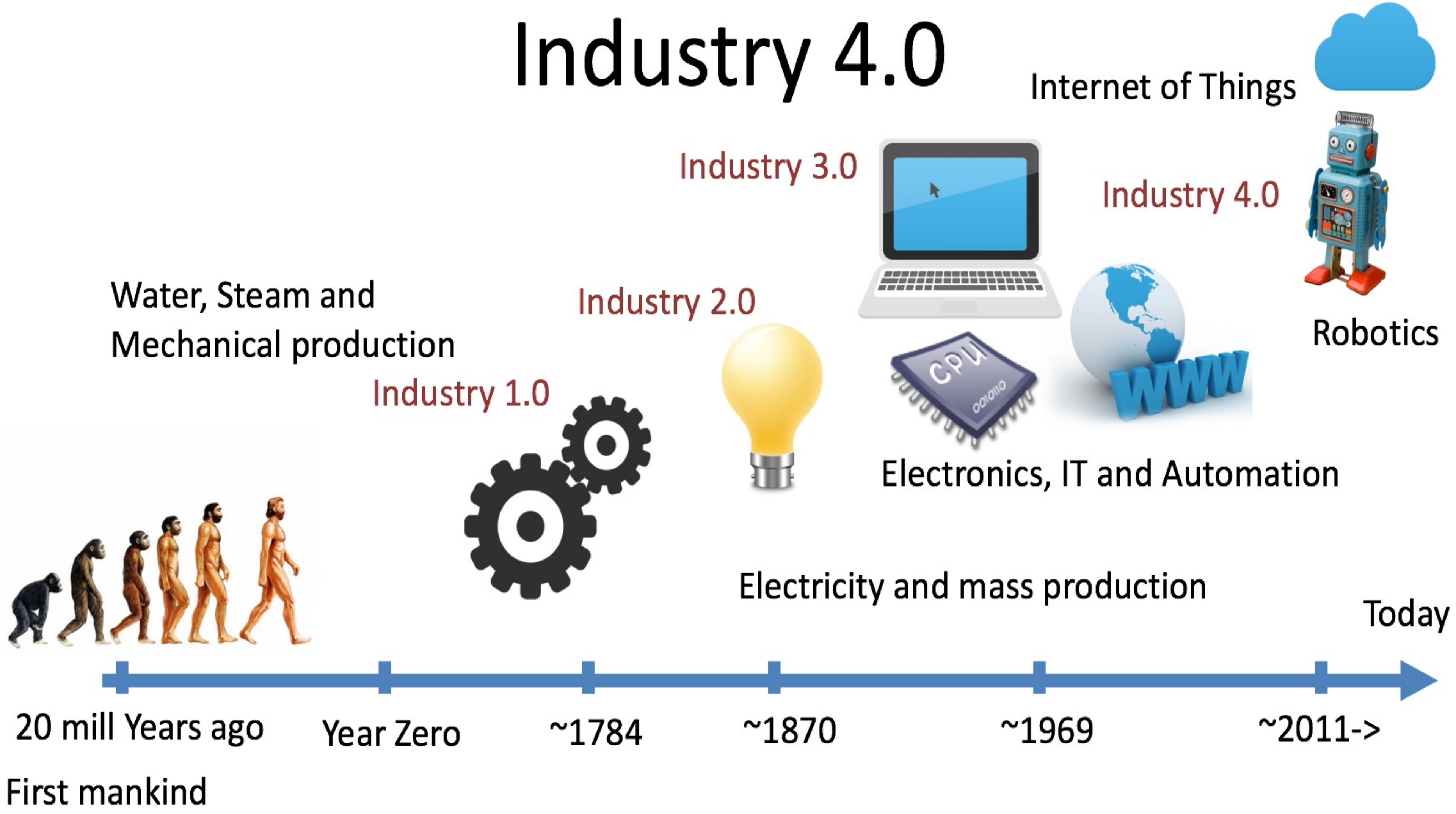
Industry 4.0

Table of Contents

Industry 4.0

- Industry 4.0 is the new buzzword for the combination of industry, automation and the current Internet of Things (IoT) technology
- IIoT – Industrial use of IoT Technology. Industrial Internet of Things (IIoT) is another word for Industry 4.0.
- You could say that IoT is more consumer oriented with applications like Smart Home, Home Automation, etc., while IIoT has more industrial focus and applications.
- The term "Industrie 4.0" was first used in 2011 in Germany.
- Industry 4.0 is also called the fourth industrial revolution.

Industry 4.0



Internet of Things

Industry 3.0

Industry 4.0

Water, Steam and Mechanical production

Industry 2.0

Robotics

Industry 1.0

Electronics, IT and Automation

Electricity and mass production

Today

20 mill Years ago

First mankind

Year Zero

~1784

~1870

~1969

~2011->

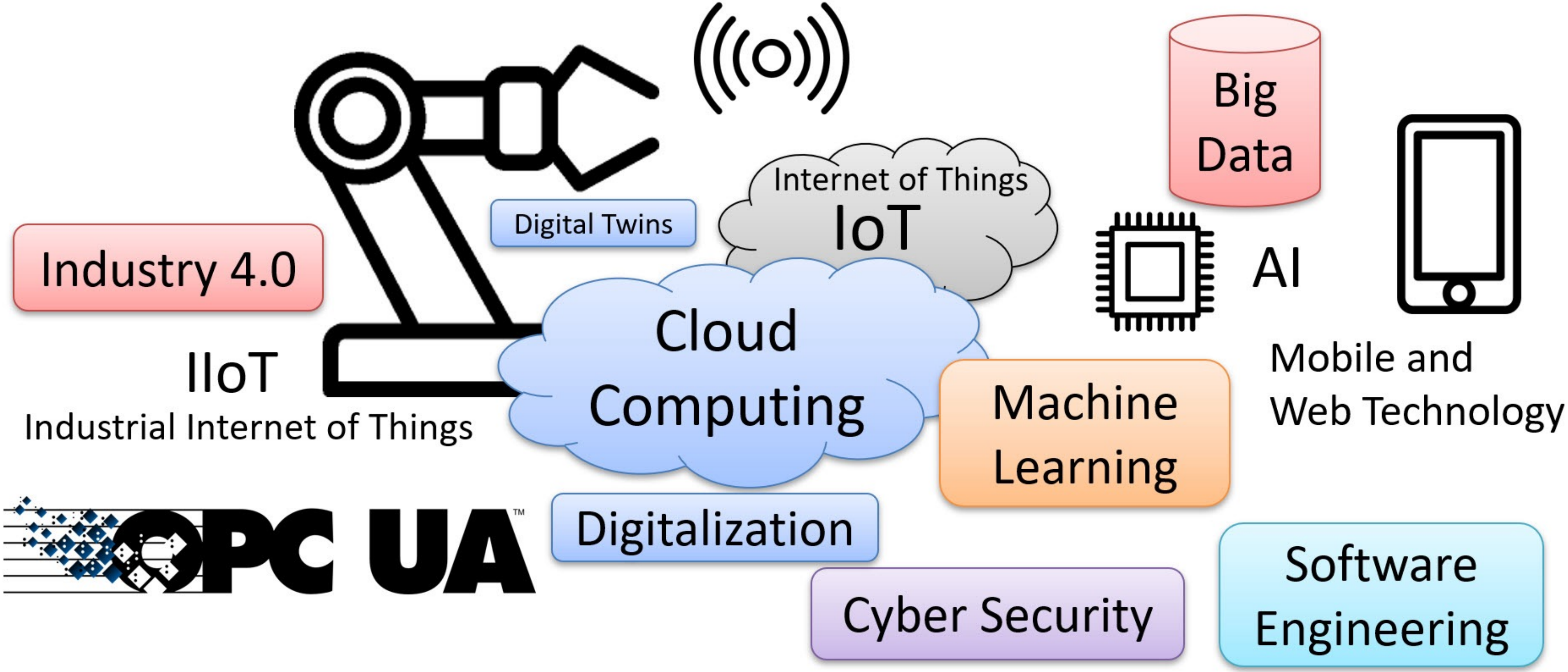
The 4. Industrial Revolution

Industry 4.0 is also called the fourth industrial revolution.

- **Industry 1.0**: Mechanization of production using Water and Steam Power.
- **Industry 2.0**: Mass production with the help of Electric Power.
- **Industry 3.0**: The Digital Revolution. From Analog to Digital Devices and Signals. Use of Electronics and IT to further Automate Production
- **Industry 4.0**: The combination of industry, automation, digitalization and the current Internet of Things (IoT) technology.

Focus on Next Generation Industry

We will learn the latest technology and terms used in the industry today and tomorrow



Industry 4.0

IIoT

Industrial Internet of Things

Digital Twins

Internet of Things

IoT

Cloud Computing

Digitalization

OPC UA™

Big Data

AI

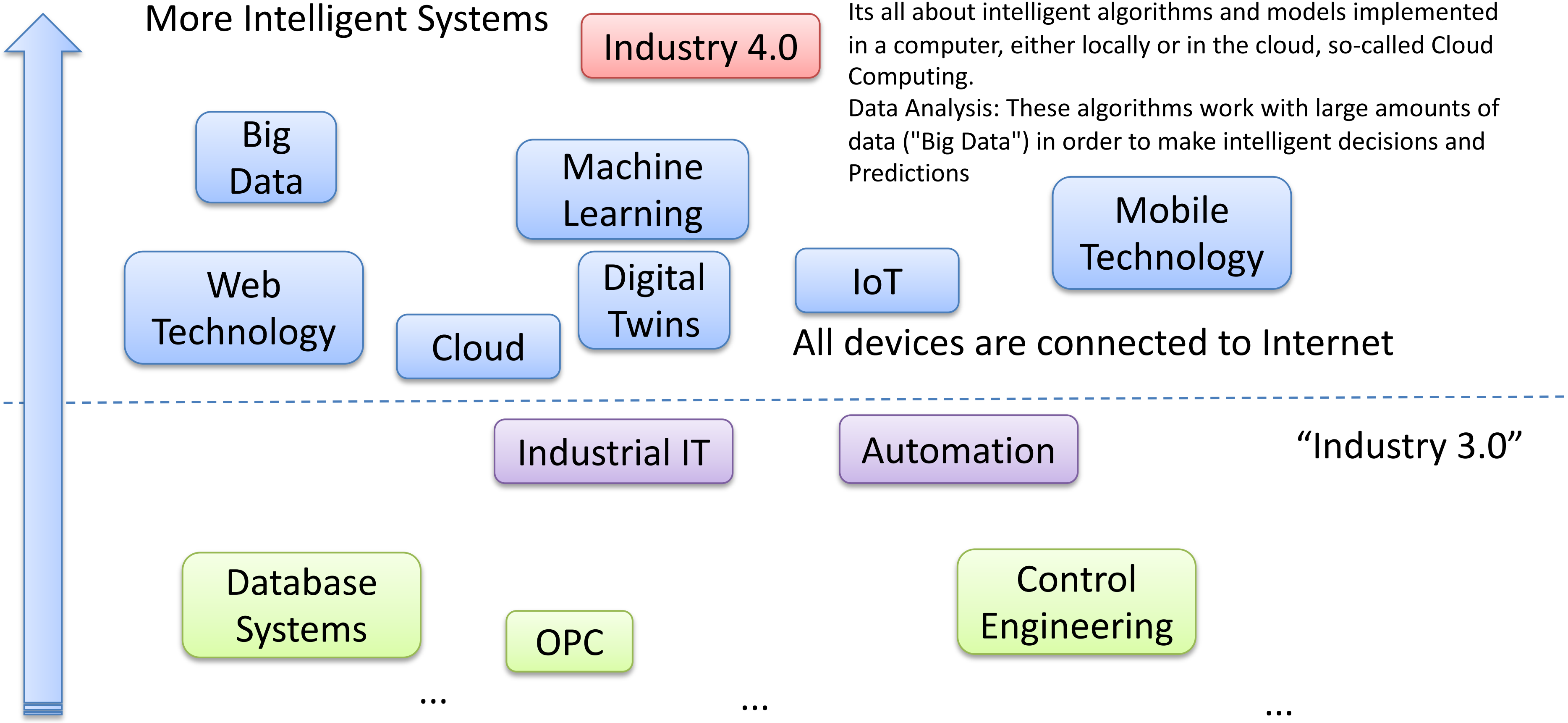
Mobile and Web Technology

Machine Learning

Cyber Security

Software Engineering

Moving forward to Industry 4.0



More Intelligent Systems

Industry 4.0

Its all about intelligent algorithms and models implemented in a computer, either locally or in the cloud, so-called Cloud Computing.
Data Analysis: These algorithms work with large amounts of data ("Big Data") in order to make intelligent decisions and Predictions

Big Data

Machine Learning

Mobile Technology

Web Technology

Cloud

Digital Twins

IoT

All devices are connected to Internet

Industrial IT

Automation

“Industry 3.0”

Database Systems

OPC

Control Engineering

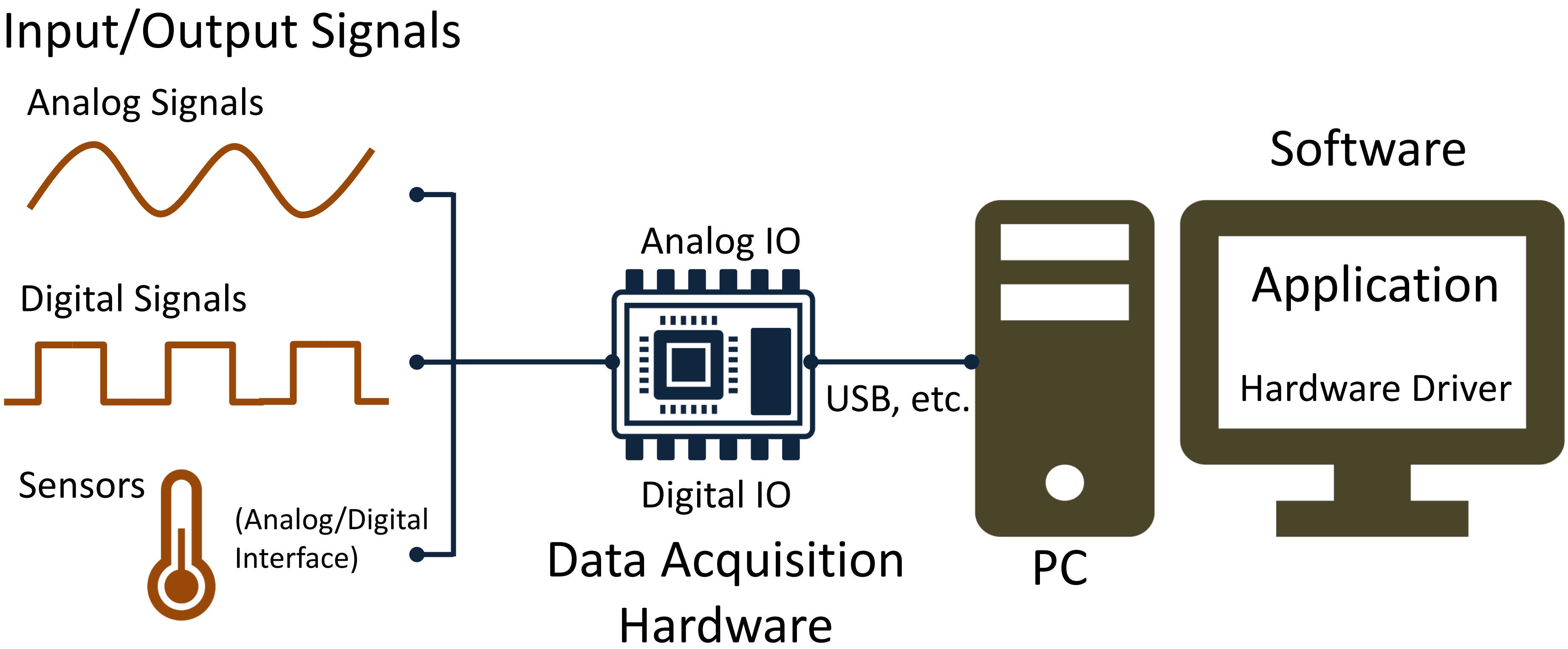
...

...

...

DAQ

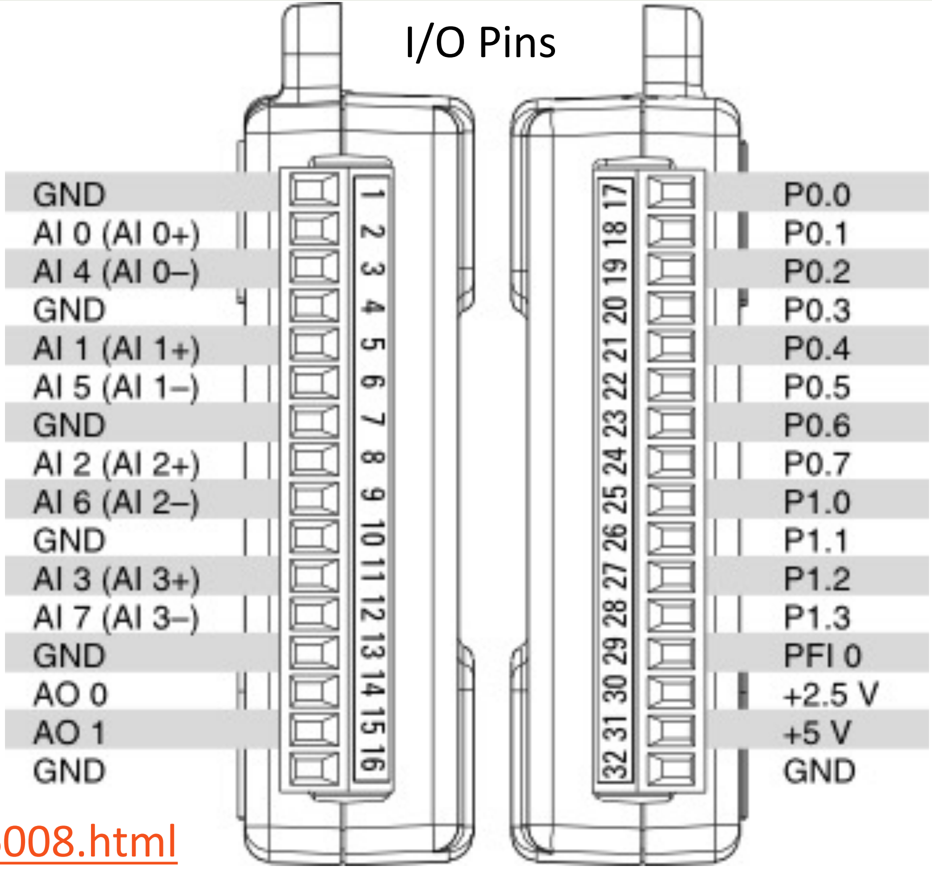
DAQ System


Input/Output Signals
Analog Signals
Digital Signals
Sensors
(Analog/Digital
Interface)
Analog IO
Digital IO
Data Acquisition
Hardware
USB, etc.
PC
Software
Application
Hardware Driver

NI USB-6008

We will use NI USB-6008 in our examples

I/O Pins



GND	1	17	P0.0
AI 0 (AI 0+)	2	18	P0.1
AI 4 (AI 0–)	3	19	P0.2
GND	4	20	P0.3
AI 1 (AI 1+)	5	21	P0.4
AI 5 (AI 1–)	6	22	P0.5
GND	7	23	P0.6
AI 2 (AI 2+)	8	24	P0.7
AI 6 (AI 2–)	9	25	P1.0
GND	10	26	P1.1
AI 3 (AI 3+)	11	27	P1.2
AI 7 (AI 3–)	12	28	P1.3
GND	13	29	PFI 0
AO 0	14	30	+2.5 V
AO 1	15	31	+5 V
GND	16	32	GND

http://www.ni.com/en-no/support/model.usb-6008.html

NI USB-6008/DAQmx

We can use NI USB-6008 (or similar) in many different Programming Languages, such as

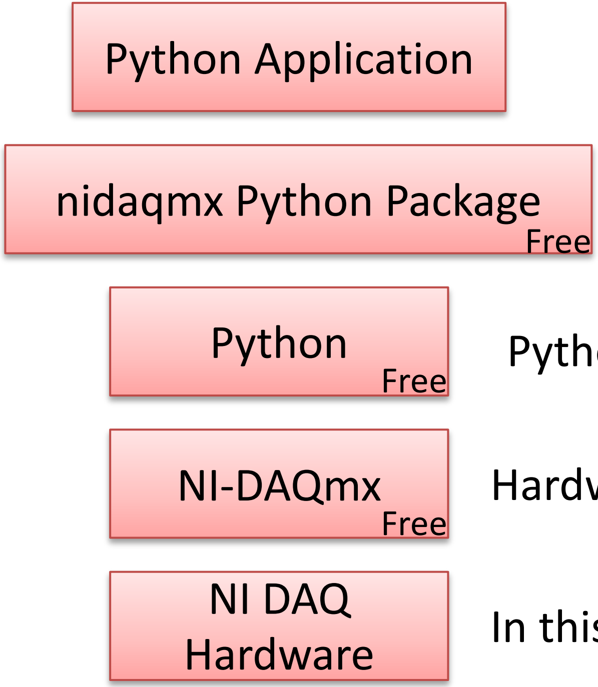
- LabVIEW
- Visual Studio/C#
- Python

In all cases we need to install the **NI-DAQmx Driver**.

LabVIEW/C# examples have been introduced earlier. Here, some basic Python examples will be provided.

NI DAQ Device with Python

How to use a NI DAQ Device with Python



Python Application

nidaqmx Python Package
Free

Python
Free

NI-DAQmx
Free

NI DAQ Hardware

Your Python Program

Python Library/API for Communication with NI DAQmx Driver

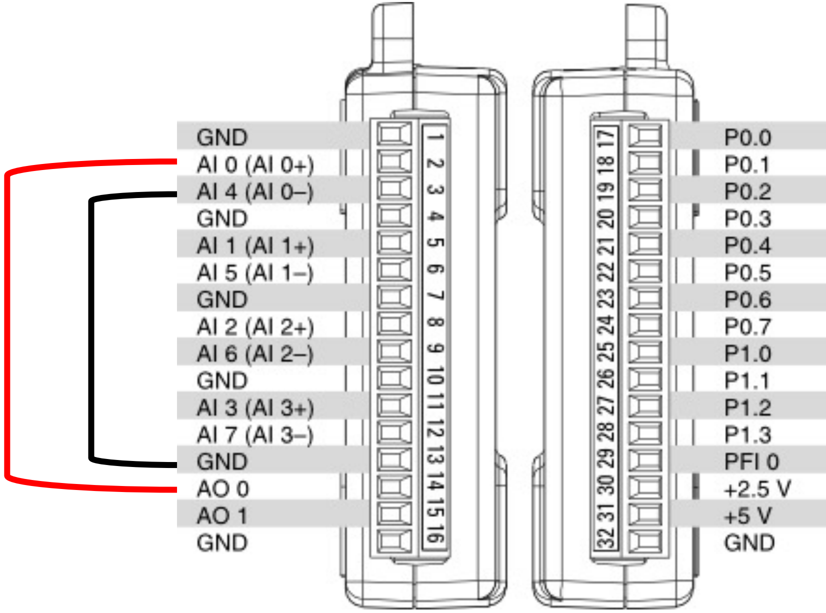
Python Programming Language

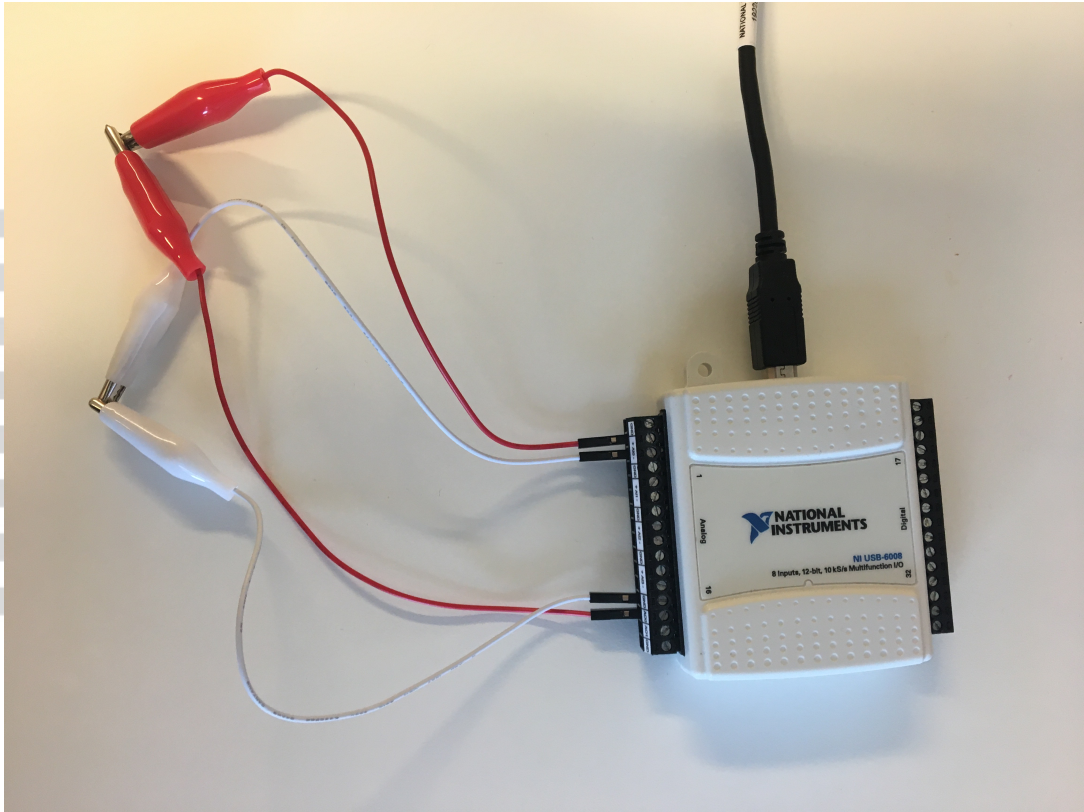
Hardware Driver Software

In this Tutorial we will use USB-6008

Test of DAQ Device (Loopback Test)

Connect Analog Out connectors on DAQ device to the Analog In connectors:





Python - Analog In (Read)

```
import nidaqmx

task = nidaqmx.Task()
task.ai_channels.add_ai_voltage_chan("Dev1/ai0")
task.start()

value = task.read()
print(value)

task.stop
task.close()
```

This example can be used as a foundation for reading the Temperature value from the real Air Heater System

Python - Analog Out (Write)

```
import nidaqmx

task = nidaqmx.Task()
task.ao_channels.add_ao_voltage_chan('Dev1/ao0','mychannel',0,5)
task.start()

value = 2
task.write(value)

task.stop()
task.close()
```

This example can be used as a foundation for sending the Controller output to the real Air Heater System

You can, e.g., use a **Multimeter** in order to check if the the program outputs the correct value

Discrete Lowpass Filter

Lowpass Filter:

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

We can find the Differential Equation for this filter using Inverse Laplace:

$$T_f \dot{y}_f + y_f = y$$

We use Euler Backward method: $\dot{x} \approx \frac{x(k)-x(k-1)}{T_s}$

Then we get:

$$T_f \frac{y_f(k) - y_f(k-1)}{T_s} + y_f(k) = y(k)$$

This gives: $y_f(k) = \frac{T_f}{T_f+T_s} y_f(k-1) + \frac{T_s}{T_f+T_s} y(k)$

We define:

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

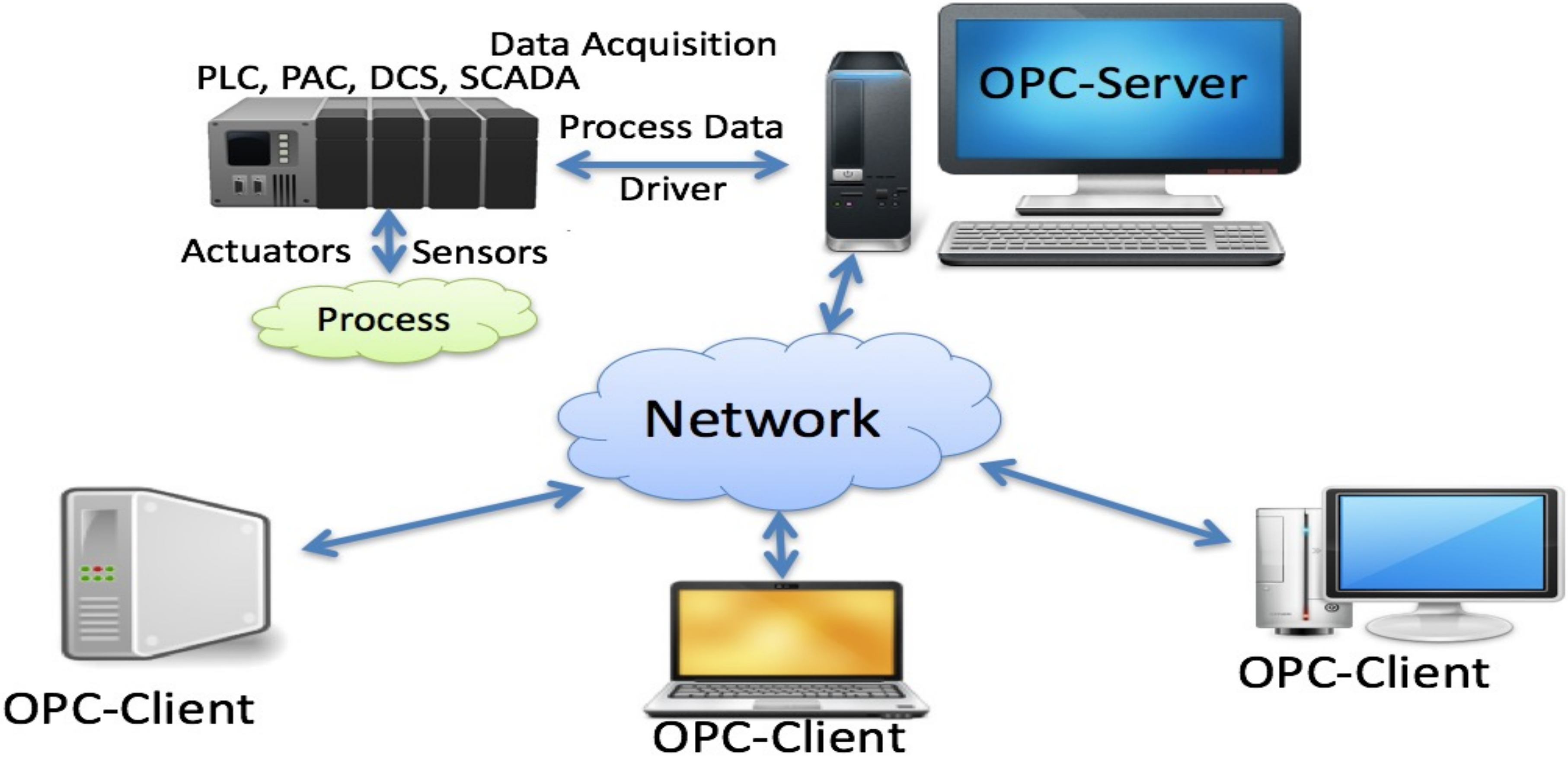
Finally, we get the following discrete version of the Lowpass Filter:

$$y_f(k) = (1-a)y_f(k-1) + ay(k)$$

This equation can easily be implemented using Python or another programming language

OPC

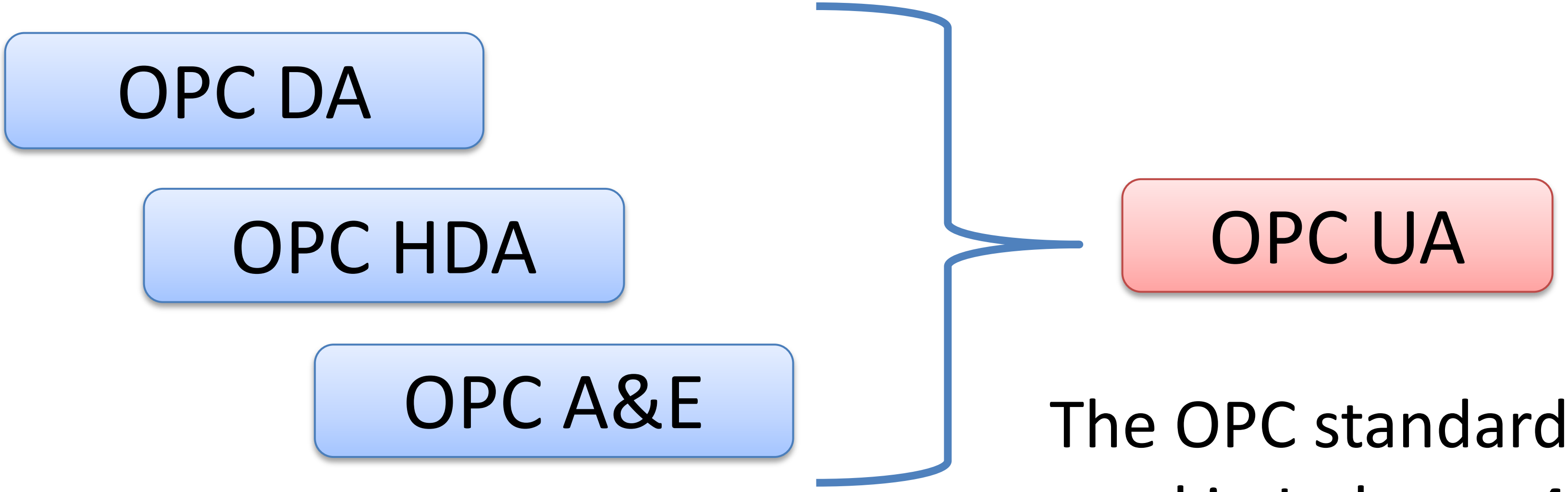
OPC


Data Acquisition
PLC, PAC, DCS, SCADA
Process Data
Driver
OPC-Server
Actuators
Sensors
Process
Network
OPC-Client
OPC-Client
OPC-Client

OPC UA

“Classic” OPC

“Next Generation” OPC



OPC DA

OPC HDA

OPC A&E

OPC UA

… (Many others)

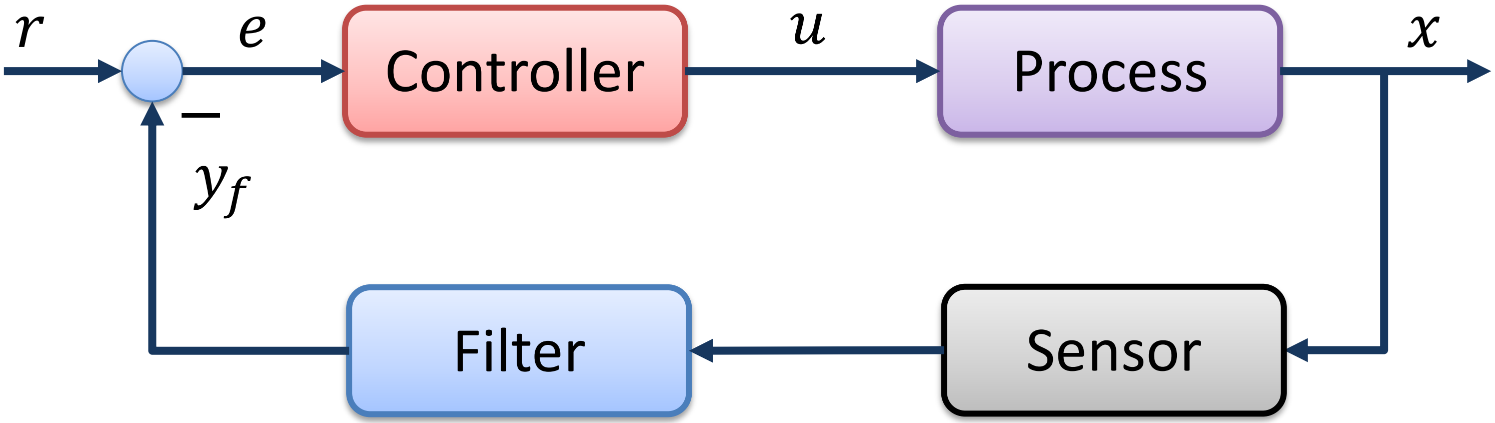
The OPC standard used in Industry 4.0

Control Engineering

Hans-Petter Halvorsen

Table of Contents

Control System

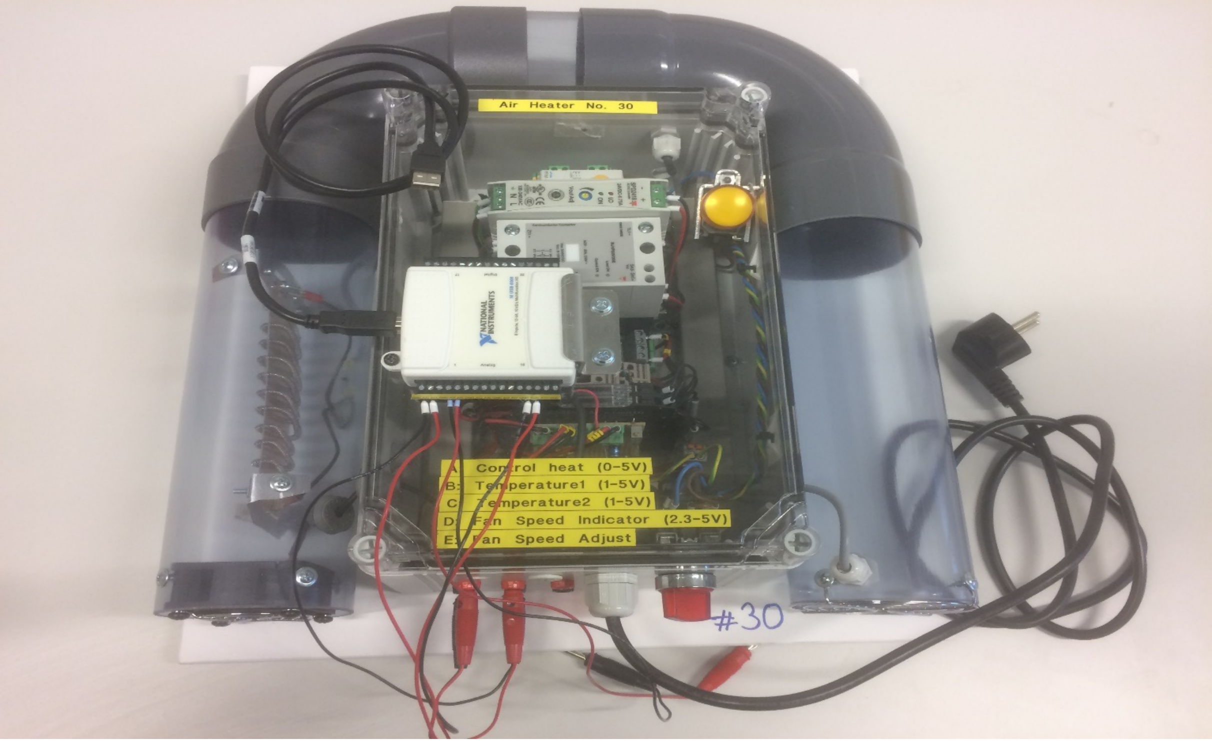


Air Heater System

Hans-Petter Halvorsen

Table of Contents

Air Heater System



We can, e.g., use the following values in the simulation:

$$\theta_t = 22\ s$$

$$\theta_d = 2\ s$$

$$K_h = 3.5\ \frac{°C}{V}$$

$$T_{env} = 21.5\ °C$$

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

Discrete Air Heater

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:

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

We can then implement the discrete model in any programming language

Simulation Ex.

1. Order System: $\dot{y} = ay + bu$

Discretization:

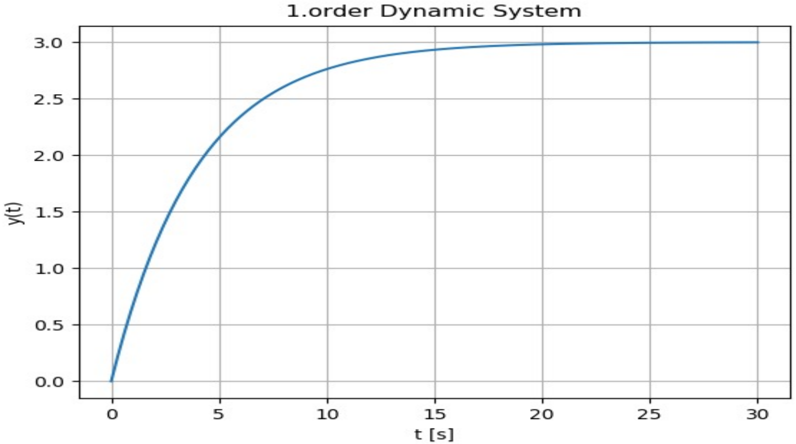
$$y_{k+1} = (1 + T_s a)y_k + T_s b u_k$$

Where $a = -\frac{1}{T}$ and $b = \frac{K}{T}$

In the Python code we can set:

$K = 3$

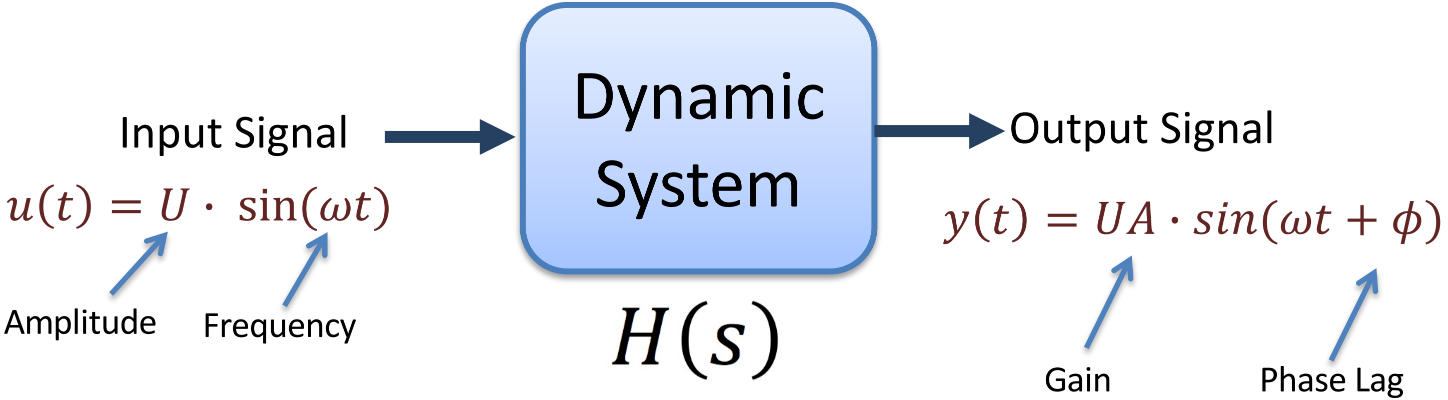
$T = 4$



```
import numpy as np
import matplotlib.pyplot as plt

# Model Parameters
K = 3
T = 4

a = -1/T
b = K/T

# Simulation Parameters
Ts = 0.1
Tstop = 30
uk = 1 # Step Response
yk = 0 # Initial Value
N = int(Tstop/Ts) # Simulation length
data = []
data.append(yk)

# Simulation
for k in range(N):
   yk1 = (1 + a*Ts) * yk  + Ts * b * uk
   yk = yk1
   data.append(yk1)

# Plot the Simulation Results
t = np.arange(0,Tstop+Ts,Ts)

plt.plot(t,data)
plt.title('1.order Dynamic System')
plt.xlabel('t [s]')
plt.ylabel('y(t)')
plt.grid()
```

Frequency Response

Frequency Response



Input Signal

$u(t) = U \cdot \sin(\omega t)$

Amplitude

Frequency

Dynamic System

$H(s)$

Output Signal

$y(t) = UA \cdot sin(\omega t + \phi)$

Gain

Phase Lag

The frequency response of a system expresses how a **sinusoidal** signal of a given frequency on the system input is transferred through the system. The only difference in the signal is the **gain** and the **phase lag**.

Bode Diagram

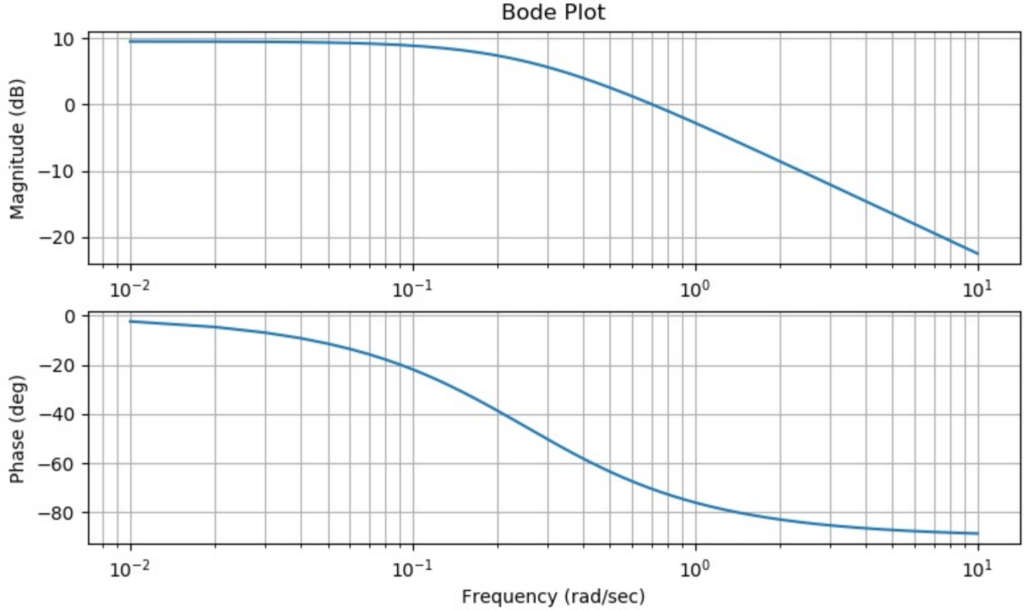
- The Bode diagram gives a simple Graphical overview of the Frequency Response for a given system.
- The Bode Diagram is tool for Analyzing the Stability properties of the Control System.
- You can find the Bode diagram from experiments on the physical process or from the transfer function (the model of the system). We will use the Transfer Function

Bode Diagram Explained

The y-scale is in $[dB]$

Magnitude/ Gain Plot

Phase Plot



Normally, the unit for frequency is Hertz [Hz], but in frequency response and Bode diagrams we use radians ω $[rad/s]$.

The y-scale is in $[degrees]$

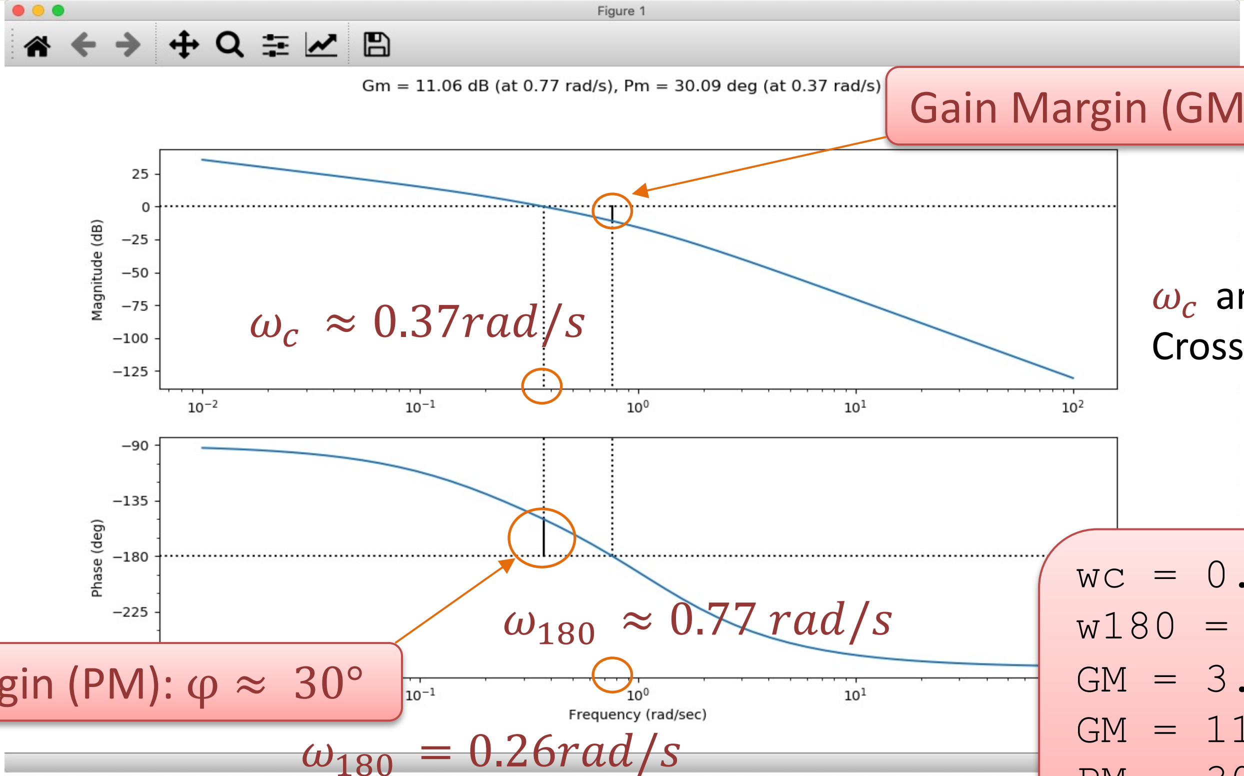
The x-scale is in radians ω $[rad/s]$

The x-scale is logarithmic

Frequency Response Analysis Example

$K_p = 0.4$

$T_i = 2s$



Gm = 11.06 dB (at 0.77 rad/s), Pm = 30.09 deg (at 0.37 rad/s)

Gain Margin (GM): $\Delta K \approx 11.dB$

$\omega_c \approx 0.37 rad/s$

ω_c and ω_{180} are called Crossover Frequencies

$\omega_{180} \approx 0.77\ rad/s$

Phase Margin (PM): $\varphi \approx 30°$

$\omega_{180} = 0.26 rad/s$

```
wc = 0.37 rad/s
w180 = 0.77 rad/s
GM = 3.57
GM = 11.06 dB
PM = 30.09 deg
Kc = 1.43
```

As you see we have an Asymptotically Stable System

The Critical Gain is $K_c = K_p \times \Delta K = 1.43$

Stability Analysis

Hans-Petter Halvorsen

Table of Contents

Stability Analysis

How do we figure out that the Feedback System is stable before we test it on the real System? We have 3 different methods:

1. Poles
2. Frequency Response/Bode
3. Simulations (Step Response)

We will do all these things using, e.g., MATLAB or Python

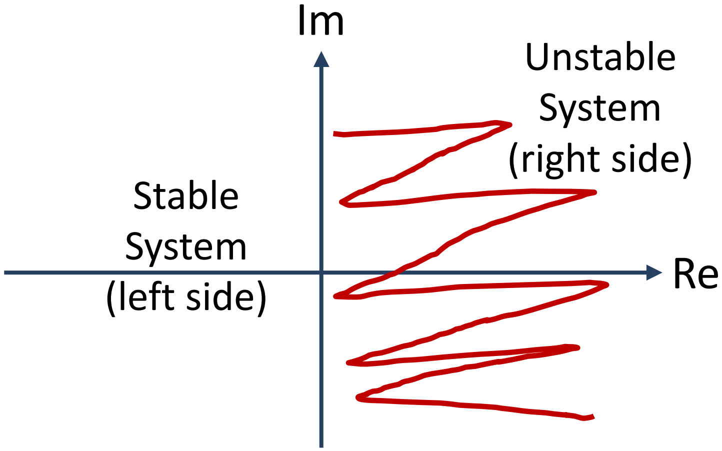
Stability Analysis

It is important to check the Stability properties of a given Control System and perform simulations before applied to the real process

- In the complex domain we can check the stability of the control system by the placements of the poles
- In the time domain we can simulate the system, e.g., performing a simple step response
- In the frequency domain we can check stability properties using, e.g., a Bode diagram

Poles and Stability of the System

The poles are important when analyzing the stability of a system. The Figure below gives an overview of the poles impact on the stability of a system.



We have 3 different Alternatives:

1. Asymptotically Stable System
2. Marginally Stable System
3. Unstable System

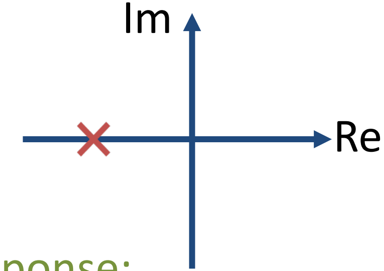
Stability Analysis

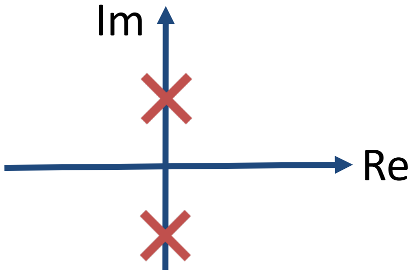
Asymptotically Stable System

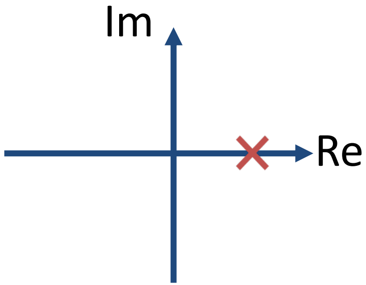
Marginally Stable System

Unstable System

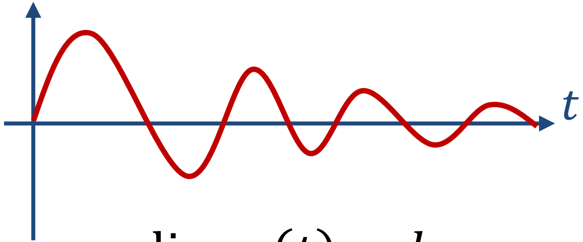
Poles:



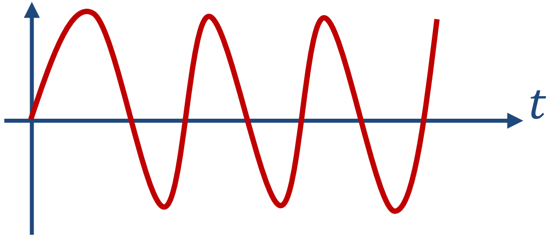




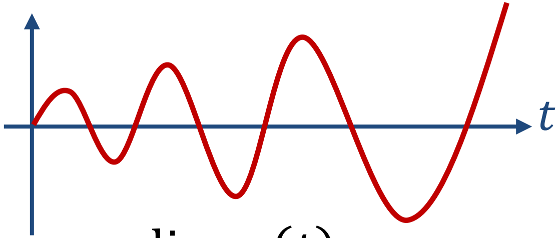
Step Response:



$$\lim_{t\to\infty} y(t) = k$$



$$0 < \lim_{t\to\infty} y(t) < \infty$$



$$\lim_{t\to\infty} y(t) = \infty$$

Frequency Response:

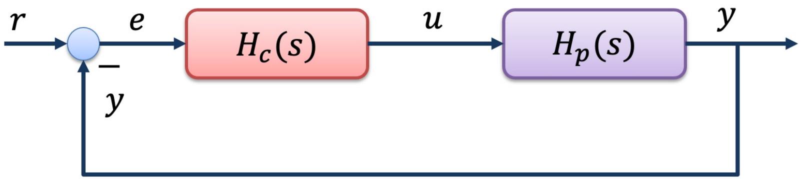
$$\omega_c < \omega_{180}$$

$$\omega_c = \omega_{180}$$

$$\omega_c > \omega_{180}$$

Loop Transfer Function

Control System:



PI Controller:

$$H_c(s) = \frac{K_p(T_i s + 1)}{T_i s}$$

Process (random example):

$$H_p(s) = \frac{2}{3s + 1}$$

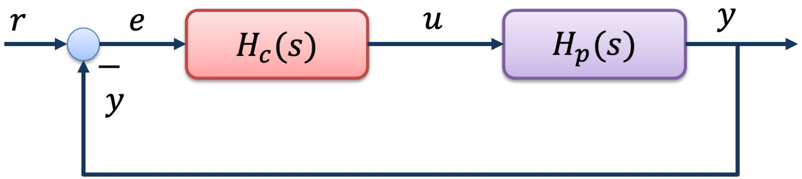
Loop Transfer Function:

$$L(s) = H_c(s)H_p(s)$$

```
import numpy as np
import control

# Controller
Kp = 0.4
Ti = 2
num = np.array ([Kp*Ti, Kp])
den = np.array ([Ti , 0])
Hc = control.tf(num , den)

# Process
num = np.array ([2])
den = np.array ([3 , 1])
Hp = control.tf(num , den)

L = control.series(Hc, Hp)
print(L)
```

Tracking Transfer Function

Control System:



PI Controller:

$$H_c(s) = \frac{K_p(T_i s + 1)}{T_i s}$$

Process (random example):

$$H_p(s) = \frac{2}{3s + 1}$$

Loop Transfer Function:

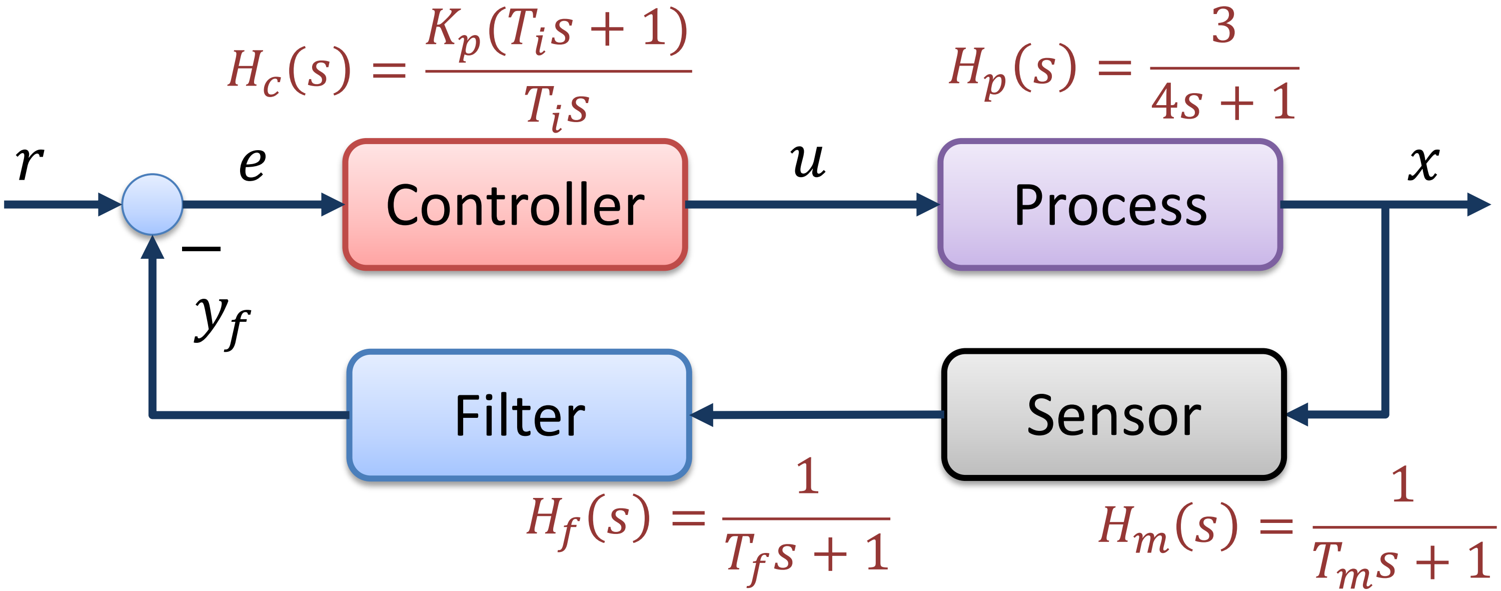
$$L(s) = H_c(s)H_p(s)$$

Tracking Transfer Function:

$$T(s) = \frac{y(s)}{r(s)} = \frac{L(s)}{1 + L(s)}$$

```python
import numpy as np
import control

# Controller
Kp = 0.4
Ti = 2
num = np.array ([Kp*Ti, Kp])
den = np.array ([Ti , 0])
Hc = control.tf(num , den)

# Process
num = np.array ([2])
den = np.array ([3 , 1])
Hp = control.tf(num , den)

L = control.series(Hc, Hp)
print(L)

T = control.feedback(L,1)
print(T)
```

Stability Analysis Example



$H_c(s) = \frac{K_p(T_i s + 1)}{T_i s}$
$H_p(s) = \frac{3}{4s + 1}$
r
e
Controller
u
Process
x
$-$
y_f
Filter
Sensor
$H_f(s) = \frac{1}{T_f s + 1}$
$H_m(s) = \frac{1}{T_m s + 1}$

In Stability Analysis we use the following Transfer Functions:

Loop Transfer Function: $\boxed{L(s) = H_c(s)H_p(s)H_m(s)H_f(s)}$

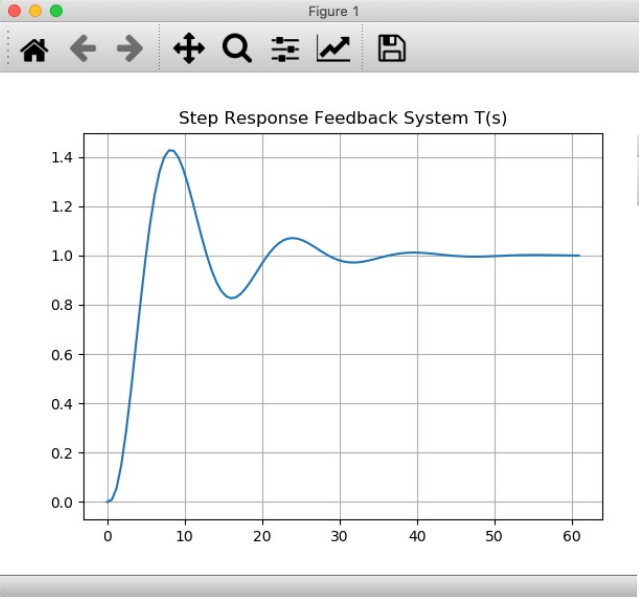
Tracking Transfer Function: $\boxed{T(s) = \frac{y(s)}{r(s)} = \frac{L(s)}{1+L(s)}}$

```python
import numpy as np
import matplotlib.pyplot as plt
import control

# Transfer Function Process
K = 3; T = 4
num_p = np.array ([K])
den_p = np.array ([T , 1])
Hp = control.tf(num_p , den_p)
print ('Hp(s) =', Hp)

# Transfer Function PI Controller
Kp = 0.4
Ti = 2
num_c = np.array ([Kp*Ti, Kp])
den_c = np.array ([Ti , 0])
Hc = control.tf(num_c, den_c)
print ('Hc(s) =', Hc)

# Transfer Function Measurement
Tm = 1
num_m = np.array ([1])
den_m = np.array ([Tm , 1])
Hm = control.tf(num_m , den_m)
print ('Hm(s) =', Hm)

# Transfer Function Lowpass Filter
Tf = 1
num_f = np.array ([1])
den_f = np.array ([Tf , 1])
Hf = control.tf(num_f , den_f)
print ('Hf(s) =', Hf)

# The Loop Transfer function
L = control.series(Hc, Hp, Hf, Hm)
print ('L(s) =', L)
```

```python
# Tracking transfer function
T = control.feedback(L,1)
print ('T(s) =', T)

# Step Response Feedback System (Tracking System)
t, y = control.step_response(T)
plt.figure(1)
plt.plot(t,y)
plt.title("Step Response Feedback System T(s)")
plt.grid()

# Bode Diagram with Stability Margins
plt.figure(2)
control.bode(L, dB=True, deg=True, margins=True)

# Poles and Zeros
control.pzmap(T)
p = control.pole(T)
z = control.zero(T)
print("poles = ", p)

# Calculating stability margins and crossover frequencies
gm , pm , w180 , wc = control.margin(L)

# Convert gm to Decibel
gmdb = 20 * np.log10(gm)

print("wc =", f'{wc:.2f}', "rad/s")
print("w180 =", f'{w180:.2f}', "rad/s")

print("GM =", f'{gm:.2f}')
print("GM =", f'{gmdb:.2f}', "dB")
print("PM =", f'{pm:.2f}', "deg")

# Find when Sysem is Marginally Stable (Kritical Gain -  Kc)
Kc = Kp*gm
print("Kc =", f'{Kc:.2f}')
```

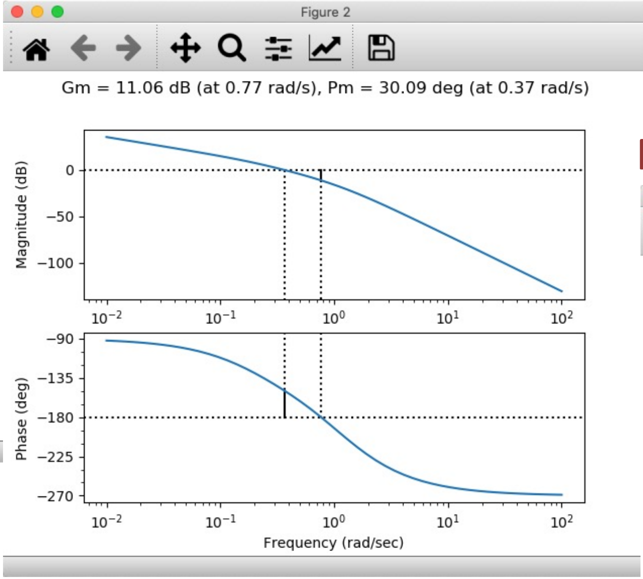
$K_p \; = \; 0.4$
$T_i \; = 2s$

Results



Step Response

Frequency Response



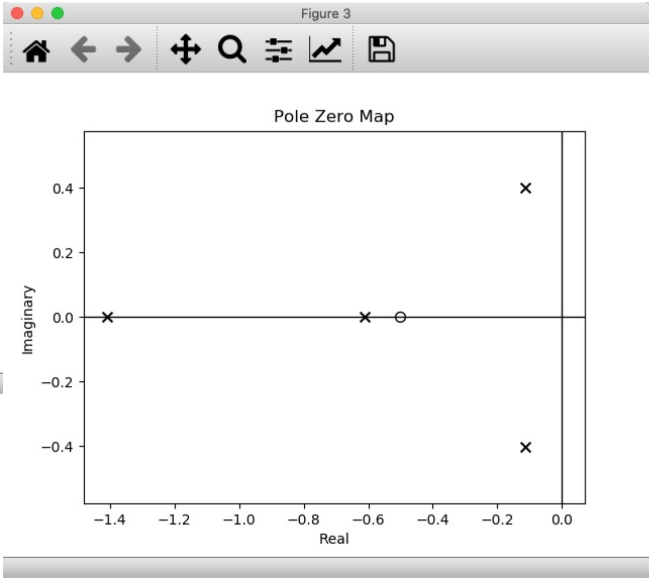
Gm = 11.06 dB (at 0.77 rad/s), Pm = 30.09 deg (at 0.37 rad/s)

Gain Margin (GM): $\Delta K \; \approx \; 11. dB$

Phase Margin (PM): $\varphi \; \approx \; 30°$

This means that we can increase K_p a bit without problem

Poles



As you see we have an Asymptotically Stable System

The Critical Gain is $K_c \; = K_p \times \Delta K = 1.43$

https://www.halvorsen.blog

PID Controller

Hans-Petter Halvorsen

Table of Contents

PID Controller

$$\boxed{u(t) = K_p e + \frac{K_p}{T_i}\int_0^t e\,d\tau + K_p T_d \dot{e}}$$

Where u is the controller output and e is the control error:

$$e(t) = r(t) - y(t)$$

r is the Reference Signal or Set-point

y is the Process value, i.e., the Measured value

Tuning Parameters:

K_p	Proportional Gain
T_i	Integral Time $[sec.]$
T_d	Derivative Time $[sec.]$

Discrete PI controller

We start with the continuous PI Controller:

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

We derive both sides in order to remove the Integral:

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

We can use the Euler Backward Discretization method:

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

Where T_s is the Sampling Time

Then we get:

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

Finally, we get:

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

Where $e_k = r_k - y_k$

Alternative PI controller

We can also put the PI Controller on Transfer Function form (we use Laplace):

$$u(s) = K_p e(s) + \frac{K_p}{T_i s} e(s)$$

We can set $z = \frac{1}{s} e \Rightarrow sz = e \Rightarrow \dot{z} = e$

This gives:

$\dot{z} = e$

$u = K_p e + \frac{K_p}{T_i} z$

This is the PI controller on State-space form

Using Euler, we get the following discrete PI controller:

$$e_k = r_k - y_k$$

$$u_k = K_p e_k + \frac{K_p}{T_i} z_k$$

$$z_{k+1} = z_k + T_s e_k$$

This algorithm can easily be implemented in the Arduino software.

Control System

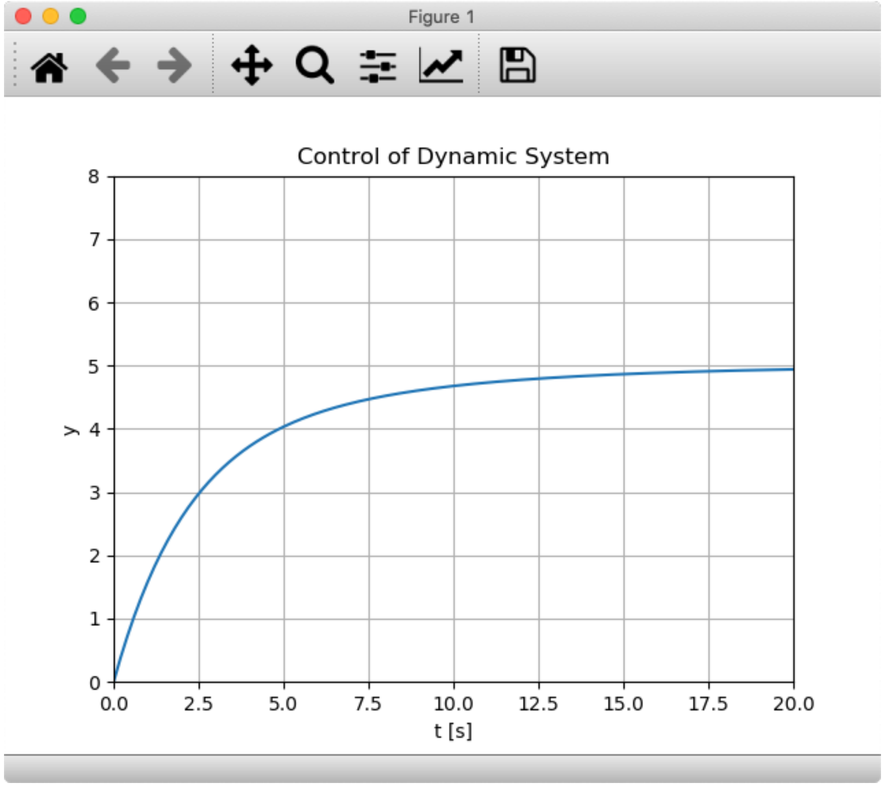
$$\boxed{\dot{y} = ay + bu}$$

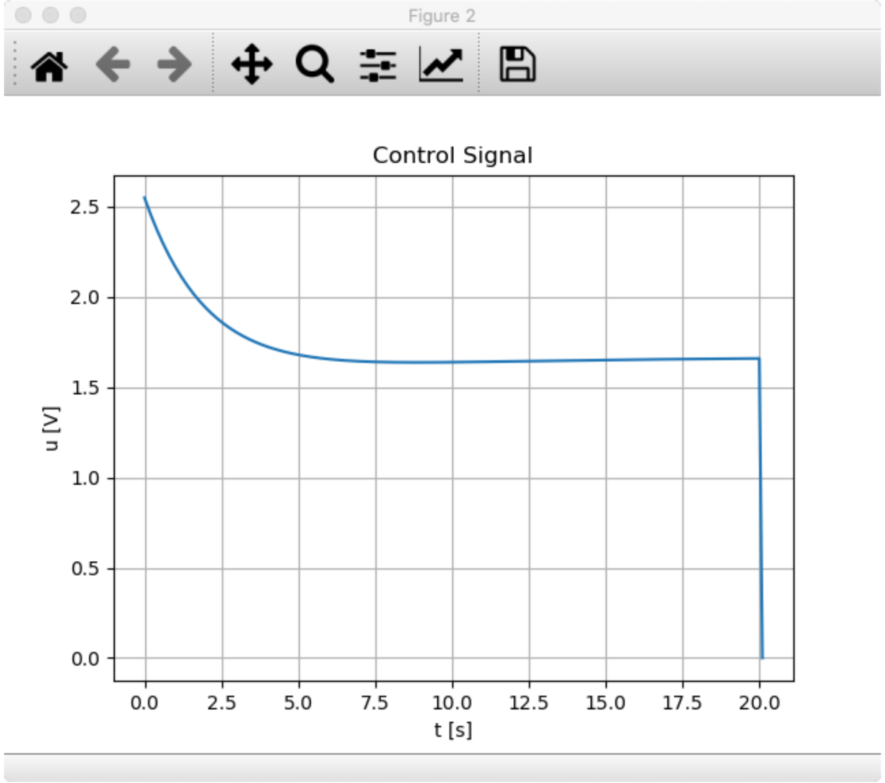
```python
import numpy as np
import matplotlib.pyplot as plt

# Model Parameters
K = 3
T = 4
a = -(1/T)
b = K/T

# Simulation Parameters
Ts = 0.1 # Sampling Time
Tstop = 20 # End of Simulation Time
N = int(Tstop/Ts) # Simulation length
y = np.zeros(N+2) # Initialization the Tout vector
y[0] = 0 # Initial Vaue

# PI Controller Settings
Kp = 0.5
Ti = 5

r = 5 # Reference value
e = np.zeros(N+2) # Initialization
u = np.zeros(N+2) # Initialization

# Simulation
for k in range(N+1):
    e[k] = r - y[k]
    u[k] = u[k-1] + Kp*(e[k] - e[k-1]) + (Kp/Ti)*Ts*e[k]
    y[k+1] = (1+Ts*a)*y[k] + Ts*b*u[k]

# Plot the Simulation Results
t = np.arange(0,Tstop+2*Ts,Ts) #Create the Time Series
```

```python
# Plot Process Value
plt.figure(1)
plt.plot(t,y)

# Formatting the appearance of the Plot
plt.title('Control of Dynamic System')
plt.xlabel('t [s]')
plt.ylabel('y')
plt.grid()
xmin = 0
xmax = Tstop
ymin = 0
ymax = 8
plt.axis([xmin, xmax, ymin, ymax])
plt.show()

# Plot Control Signal
plt.figure(2)
plt.plot(t,u)

# Formatting the appearance of the Plot
plt.title('Control Signal')
plt.xlabel('t [s]')
plt.ylabel('u [V]')
plt.grid()
```

Control System

$$\dot{y} = ay + bu$$





Database Systems

Hans-Petter Halvorsen

Table of Contents

Database Systems

- There exists lots of Database Systems today, we will focus on Microsoft SQL Server
- We can communicate with SQL Server using Programming Languages like LabVIEW, C#, Python, etc.
- Here I will focus on Python

Python Drivers for SQL Server

- There are several python SQL drivers available:
  - pyodbc
  - pymssql
- These Drivers are not made made Microsoft but the Python Community.
- However, Microsoft places its testing efforts and its confidence in pyodbc driver.
- Microsoft contributes to the pyODBC open-source community and is an active participant in the repository at GitHub

https://docs.microsoft.com/sql/connect/python/python-driver-for-sql-server

Connect to Database from Python

Example:

Server Name

If Server is on your local PC, you can use LOCALHOST

```
import pyodbc

driver = "{ODBC Driver 17 for SQL Server}"
server = “TESTPC\\SQLEXPRESS"
database = "BOOKSTORE"
username = "sa"
password = "Test123"
conn = pyodbc.connect("DRIVER=" + driver
                      + ";SERVER=" + server
                      + ";DATABASE=" + database
                      + ";UID=" + username
                      + ";PWD=" + password )
```

Instance Name (you can have multiple instances of SQL Server on the same computer)

Here is the built-in “sa” user (System Administrator) used to connect to the Database. In general, you should use another user than the sa user. The sa user is used here for simplicity. You can easily create new user in SQL Server Management Studio

Using Parameters- Avoid SQL Injection

- ODBC supports parameters using a question mark as a place holder in the SQL. You provide the values for the question marks by passing them after the SQL
- This is safer than putting the values into the string because the parameters are passed to the database separately, protecting against SQL injection attacks.
- It is also be more efficient if you execute the same SQL repeatedly with different parameters.

https://github.com/mkleehammer/pyodbc/wiki/Getting-started

Retrieving Data from Database

Example:

```
import pyodbc
import database

connectionString = database.GetConnectionString()

conn = pyodbc.connect(connectionString)

cursor = conn.cursor()

query = "select BookId, Title, Author, Category from BOOK where Category=?"

parameters = ['Data']

for row in cursor.execute(query, parameters):
    print(row.BookId, row.Title, row.Author, row.Category)
```

Insert Data into Database

In this example, you see how to run an INSERT statement safely, and pass parameters. The parameters protect your application from SQL injection.

```
import pyodbc
import database

connectionString = database.GetConnectionString()

conn = pyodbc.connect(connectionString)

cursor = conn.cursor()

query = "INSERT INTO BOOK (Title, Author, Category) VALUES (?,?,?)"

parameters = 'Python for Beginners', 'Hans-Petter Halvorsen', 'Data'

count = cursor.execute(query, parameters).rowcount
cursor.commit()

print('Rows inserted: ' + str(count))
```

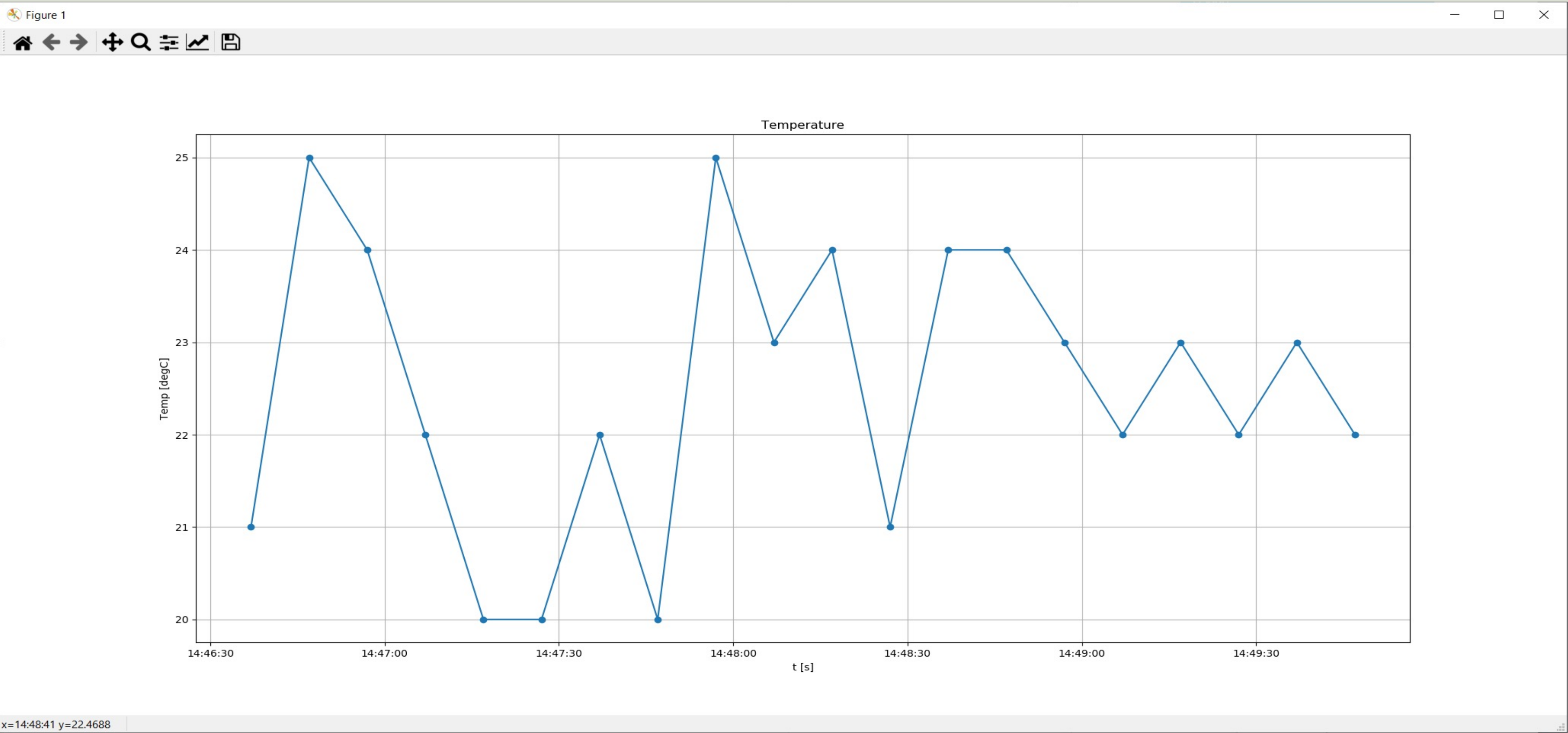
Logging Data

```
import pyodbc
import random
import time
from datetime import datetime
import database

# Connect to Database
connectionString = database.GetConnectionString()
conn = pyodbc.connect(connectionString)
cursor = conn.cursor()
query = "INSERT INTO MEASUREMENTDATA (SensorName, MeasurementValue, MeasurementDateTime) VALUES (?,?,?)"

sensorName = "Temperature"
Ts = 10 # Sampling Time
N = 20
for k in range(N):
    # Generate Random Data
    LowLimit = 20
    UpperLimit = 25
    measurementValue = random.randint(LowLimit, UpperLimit)

    #Find Date and Time
    now = datetime.now()
    datetimeformat = "%Y-%m-%d %H:%M:%S"
    measurementDateTime = now.strftime(datetimeformat)

    # Insert Data into Database
    parameters = sensorName, measurementValue, measurementDateTime
    cursor.execute(query, parameters)
    cursor.commit()

    # Wait
    time.sleep(Ts)
```

Plotting Data

```python
import pyodbc
import matplotlib.pyplot as plt
import database

sensorName = "Temperature"

# Connect to Database
connectionString = database.GetConnectionString()
conn = pyodbc.connect(connectionString)
cursor = conn.cursor()
query = "SELECT MeasurementValue, MeasurementDateTime FROM MEASUREMENTDATA WHERE SensorName=?"
parameters = [sensorName]

t = []; data = []

# Retrieving and Formatting Data
for row in cursor.execute(query, parameters):
    measurementValue = row.MeasurementValue
    measurementDateTime = row.MeasurementDateTime

    data.append(measurementValue)
    t.append(measurementDateTime)

# Plotting
plt.plot(t, data, 'o-')
plt.title('Temperature')
plt.xlabel('t [s]')
plt.ylabel('Temp [degC]')
plt.grid()
plt.show()
```

Plotted Data



ASP.NET Core

ASP.NET

- 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

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

More resources ASP.NET Core: https://www.halvorsen.blog/documents/programming/web/aspnet

Web Programming
ASP.NET Core

Hans-Petter Halvorsen

https://www.halvorsen.blog

ASP.NET Core Resources

- Textbook
- Videos
- Tutorials
- Example Code

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

Web Pages and Real-time Monitoring?

- Web Pages are typically not used for Real-time Monitoring, and **not** necessary to to implement in this assignment.
- A simple solution though is to put like this in your web page:

```
<html>
<head>
  <title>Data Monitoring</title>
  <meta http-equiv="refresh" content=”30"/>
</head>
<body>
  ..
</body>
</html>
```

This line refreshes the web page every 30 seconds

Note! For more advanced Real-time updates of Web pages, you typically use something called AJAX and JavaScript – but that is really NOT part of this assignment!

Microsoft Azure

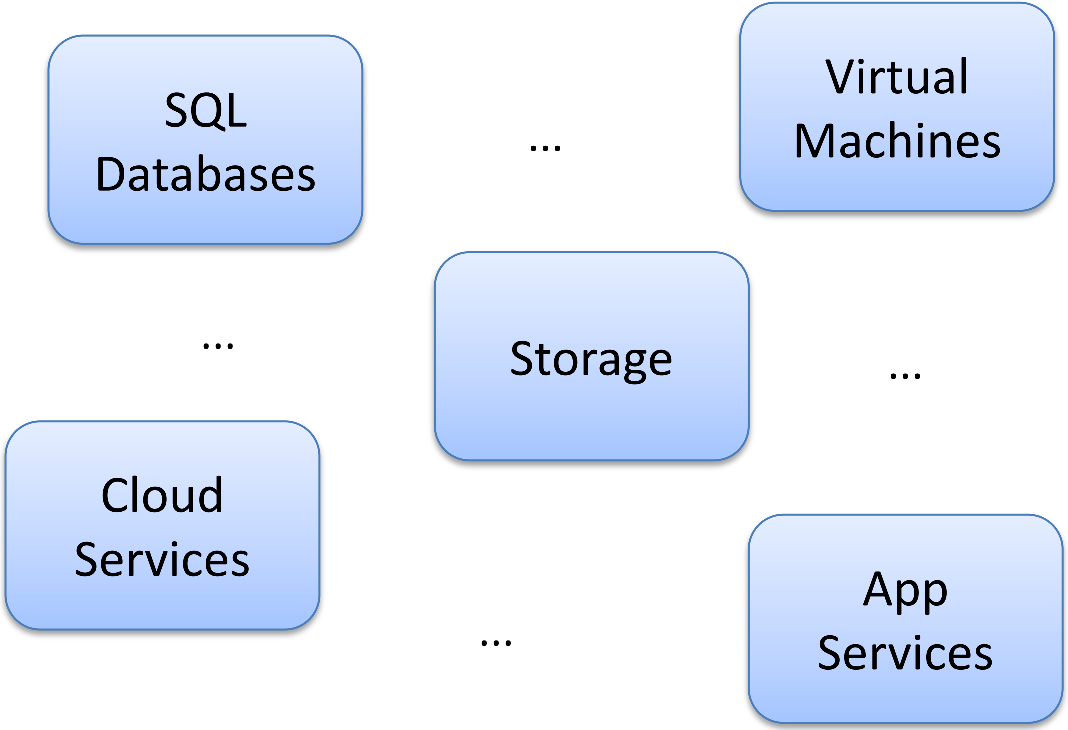
Table of Contents

Cloud Platforms

You can rent Cloud based services like Virtual Machines (Computers with OS running in the Cloud), Web Servers, Database Systems on Monthly fees and usage

Microsoft Azure

“Windows running in the Cloud”



SQL Databases

…

Virtual Machines

…

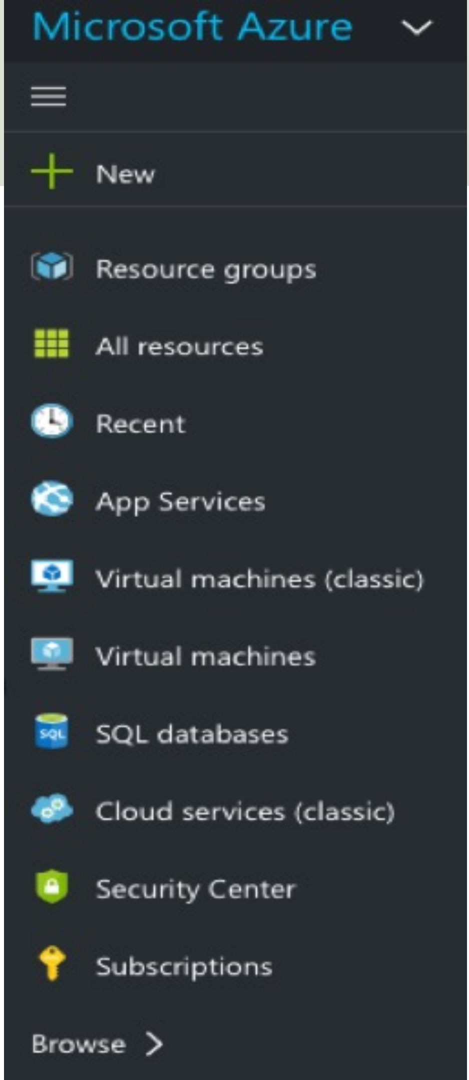
Storage

…

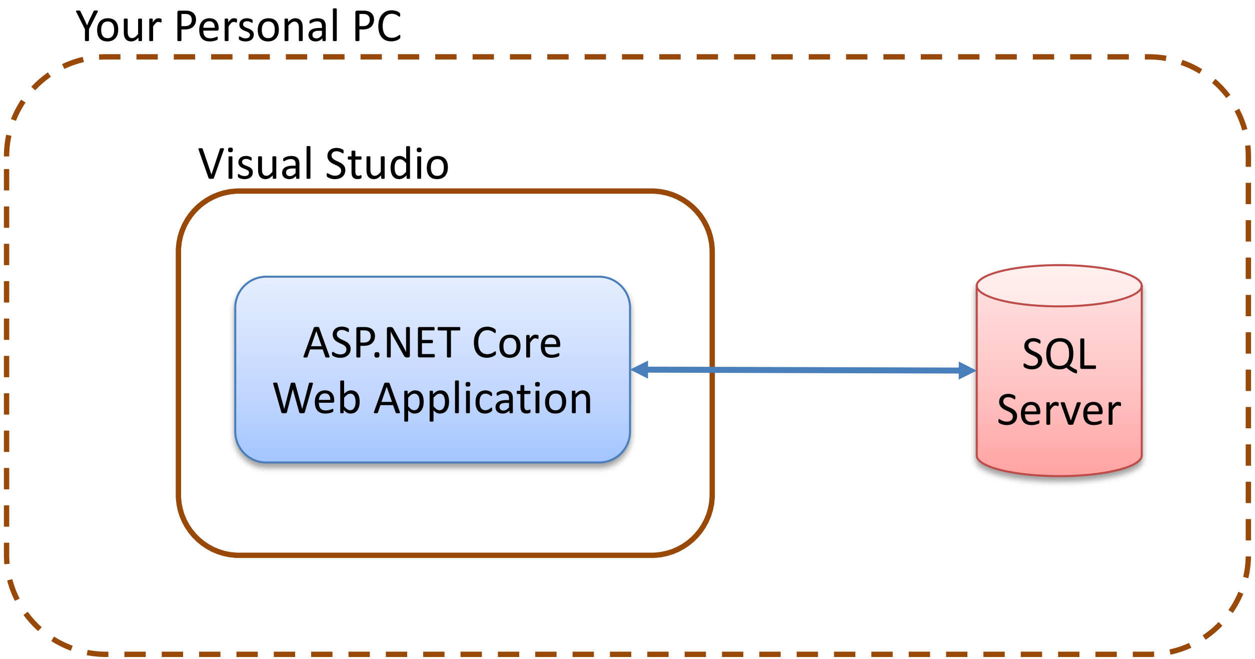
Cloud Services

…

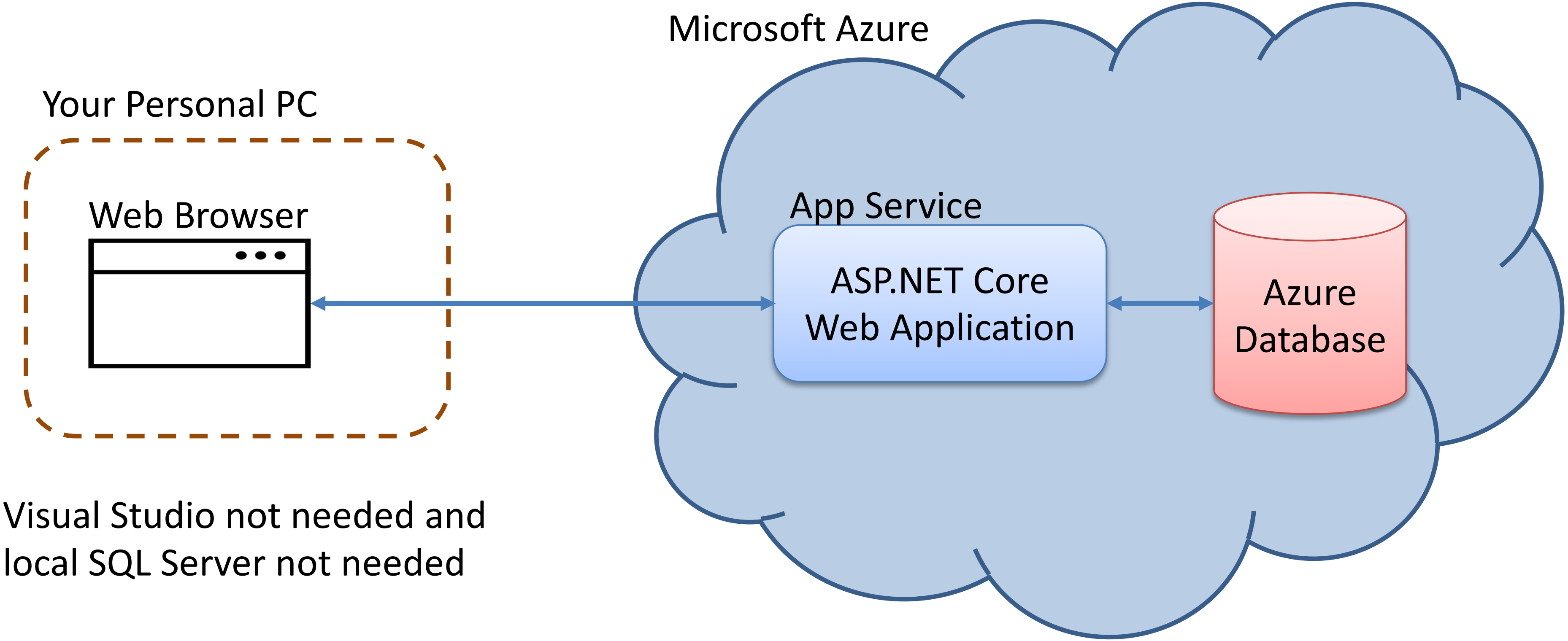
App Services



Development



Deployment to Azure



Visual Studio not needed and local SQL Server not needed

Cyber Security

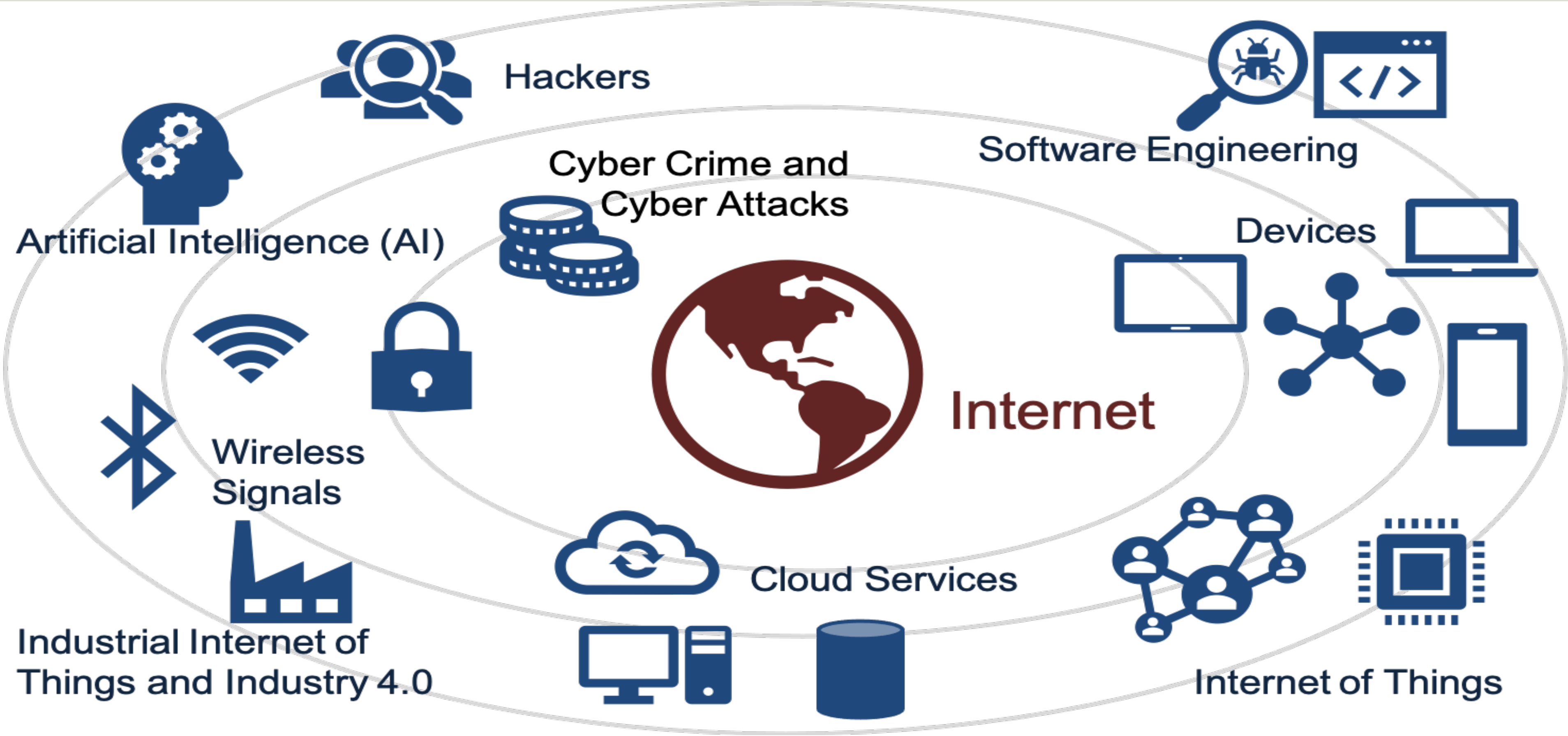
Hans-Petter Halvorsen

Table of Contents

Cyber Security

- IIoT (Industrial Internet of Things) solutions and Data Security?
- How can we make sure our applications and data are safe?
- Security is crucial in IoT/IIoT Applications

Cyber Security


Hackers
Software Engineering
Cyber Crime and Cyber Attacks
Artificial Intelligence (AI)
Devices
Internet
Wireless Signals
Cloud Services
Industrial Internet of Things and Industry 4.0
Internet of Things

Cyber Security and GDPR

- Data Security in Automation Systems?
- IoT solutions and Data Security?
- Data Security in Cloud Storage and Cloud Services?
- GDPR?
- What can be done to protect the system (and data) you have created?

Cyber Security in IACS Systems

IACS – Industrial Automation and Control Systems

- IEC62443 – Cyber Security standard for IACS systems

Cyber Security Examples

- Authentication (Login ..)
- SQL Injection
- Risk Analysis
- Cyber Security Test Tools
- ..

Authentication

Login and Authentication

- See if you are you able to create Login functionality to make the application more secure
- Are you able to implement Two-factor Authentication

SQL Injection

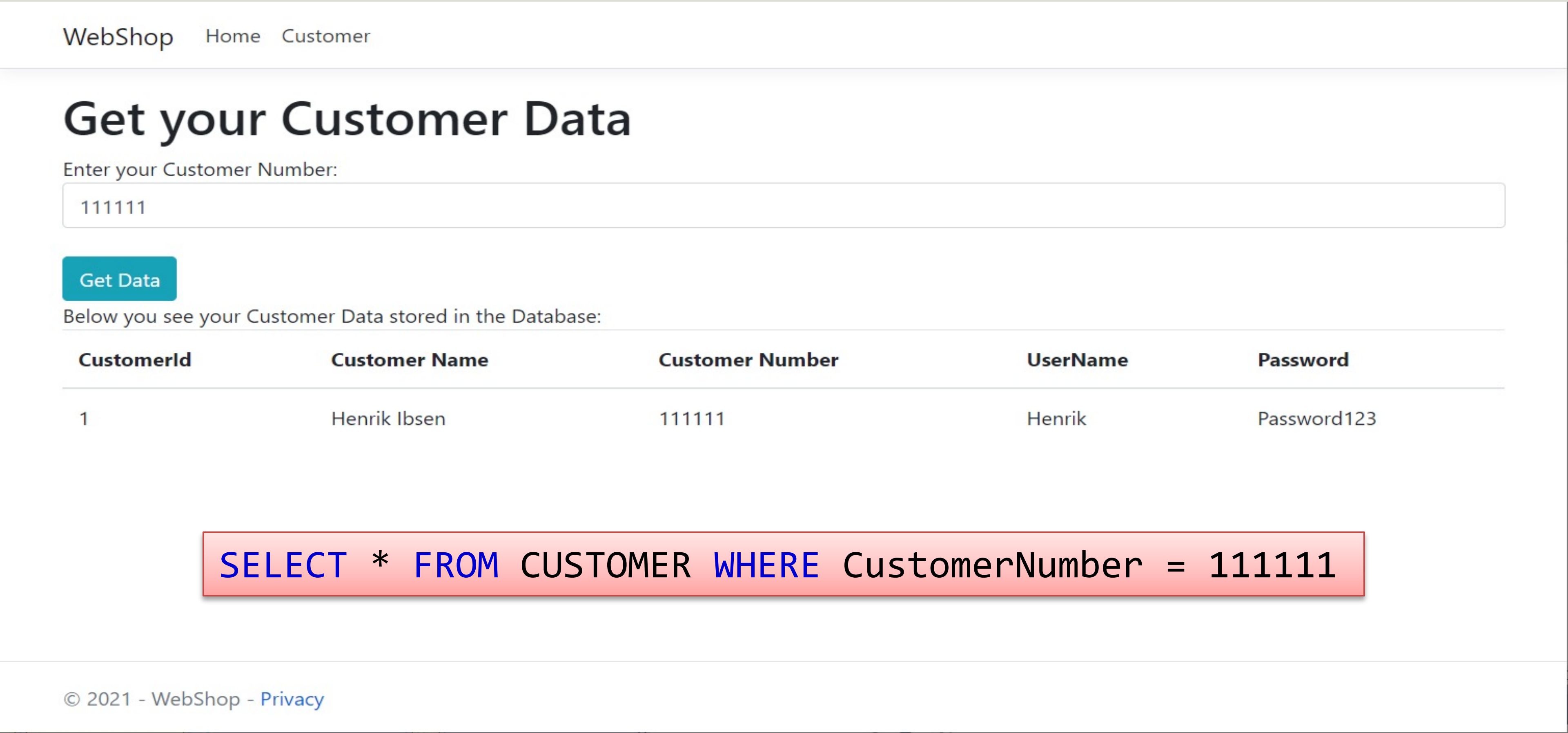
Hans-Petter Halvorsen

Table of Contents

SQL Injection

- SQL injection is a code injection technique that might destroy your database or expose information, such as passwords, etc.
- SQL injection is one of the most common web hacking techniques.
- A Structured Query Language (SQL) injection occurs when an attacker inserts malicious code into a server that uses SQL and forces the server to reveal information it normally would not.
- An attacker could carry out a SQL injection simply by submitting malicious code into a vulnerable website search box.

Web Application Example



Get your Customer Data

Enter your Customer Number:

111111

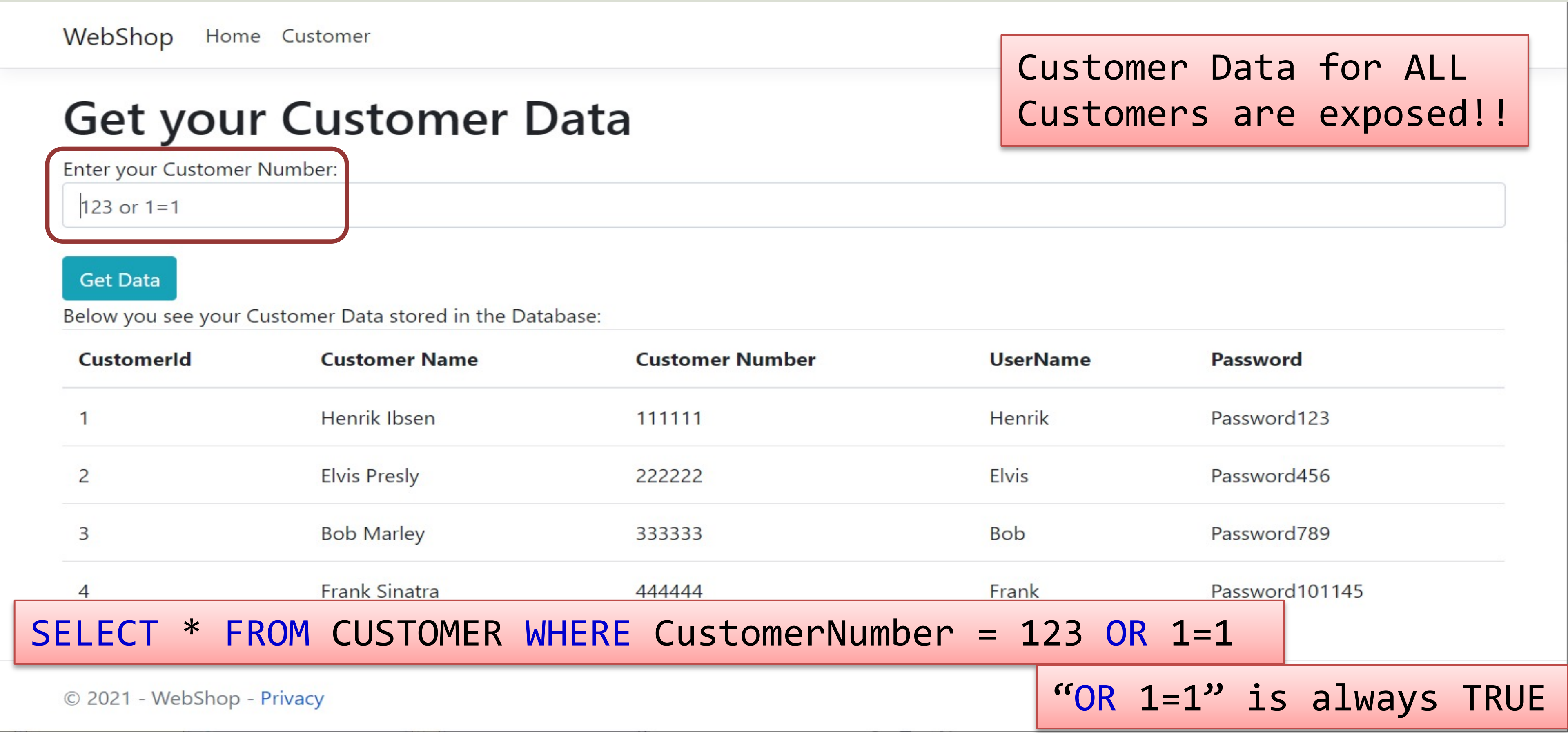
Below you see your Customer Data stored in the Database:

CustomerId	Customer Name	Customer Number	UserName	Password
1	Henrik Ibsen	111111	Henrik	Password123

```
SELECT * FROM CUSTOMER WHERE CustomerNumber = 111111
```

© 2021 - WebShop - Privacy

SQL Injection Example



Get your Customer Data

Enter your Customer Number:

123 or 1=1

Below you see your Customer Data stored in the Database:

CustomerId	Customer Name	Customer Number	UserName	Password
1	Henrik Ibsen	111111	Henrik	Password123
2	Elvis Presly	222222	Elvis	Password456
3	Bob Marley	333333	Bob	Password789
4	Frank Sinatra	444444	Frank	Password101145

© 2021 - WebShop - Privacy

Customer Data for ALL Customers are exposed!!

```
SELECT * FROM CUSTOMER WHERE CustomerNumber = 123 OR 1=1
```

“OR 1=1” is always TRUE

Prevent SQL Injection

- Use proper Data Types in your Code, i.e., int, etc. instead of string for everything
- Check Input in GUI if proper Data Type
- Use SQL Parameters
- Test your Software Properly

Risk Analysis

Risk Matrix (Severity vs Probability)

		Severity		
		Low	Medium	High
Probability	Very Likely	🟧	🟥	🟥
	Likely	🟩	🟧	🟥
	Unlikely	🟩	🟩	🟧

Critical to deal with

🟧	Low Criticality
🟩	Medium Criticality
🟥	High Criticality

Risk Analysis Example

#	Issue	Probability	Severity
1	The RFID reader has not the necessary range	Unlikely	Medium
2	What if an ID Card has been stolen?	Likely	Medium
3	Hacker attacks	Very Likely	High
4	Existing Access Card cannot be used because they don’t have RFID or different standard is used	Unlikely	Low
5	..		
6	..		
7	..		
8	..		
..	..		

Risk Matrix Example

Fill the Results from the Risk Analysis into the Risk Matrix:

		Severity		
		Low	Medium	High
Probability	Very Likely	🟨	🟥	🟥 x
	Likely	🟩	🟨 x	🟥
	Unlikely	🟩 x	🟩 x	🟨

High Criticality

Low Criticality

=> In the Development of the System focus on solving/preventing the issues with High Criticality

🟨	Low Criticality
🟩	Medium Criticality
🟥	High Criticality

https://www.halvorsen.blog

Cyber Security Test Tools

Hans-Petter Halvorsen

Table of Contents

Kali Linux

- Test if your system is secure using, e.g., Kali Linux
- Kali Linux has hundreds of pre-installed penetration-testing programs (tools).

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

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

Web: https://www.halvorsen.blog