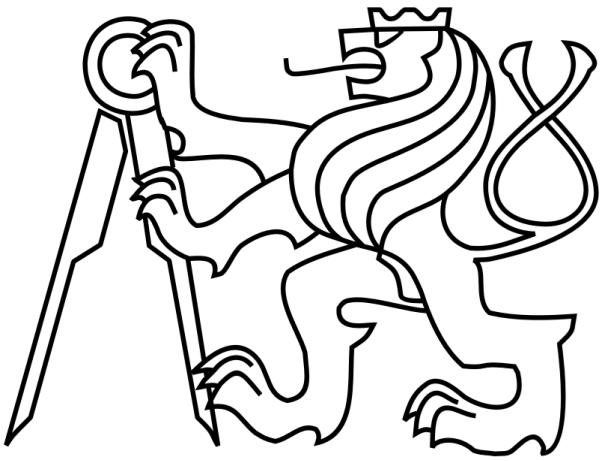
ČESKÉ VYSOKÉ UČENIE TECHNICKÉ V PRAHE

FAKULTA STROJNÍCKA

Ústav výrobných strojov a zariadení



Textové prílohy

Monitoring stavu stroja pomocou vlastnej SCADA aplikácie

2021 **Bc. Dávid Križan**

SKRIPT – scada.py

```
from opcua import Client
from timeit import time, timeit
from datetime import datetime
import pyodbc
import telnetlib
import pandas as pd

#DATA KTERE JE TREBA VYPLNIT

#dataframe vsech nazvu hodnot typu integer, ktere chceme cist ze serveru opc a
 ukladat do SQL
df_int = pd.DataFrame(columns=[#zde_vlozit_ctene_promenne,Timestamp"])

#dataframe vsech nazvu hodnot typu bool, ktere chceme cist ze serveru opc a uk
ladat do SQL
df_bool = pd.DataFrame(columns=[=[#zde_vlozit_ctene_promenne,Timestamp"])

#zadani adresy opc serveru
opcserverstring= #adresa opc serveru
opcstring= 'ns=2;s=Paintshop' #s = nazev slozky na opc serveru

#SQL

#SQL string - konfigurace sql serveru
sqlstring="Driver={SQL Server};"+"Server=DESKTOP-
LMSTPTV\WINCC;"+"Database=PLC;"+"Trusted_Connection=yes;"

#TELNET (is for checking SQL Server avaliability with short timeout)
telnetstring='localhost'
sqlport=49978 #port of sql server connection

#TOHLE PROBIHA SAMOVOLNE - neni treba nic vyplnovat


#predpriprava promenne pro buffer (pri uspesnem nacteni hodnot bude +1, pri
zapsani do SQL -1)
nezapsano = 0

#pripojeni na OPC Server
client = Client(opcserverstring)  # if anonymous authentication is enabled
# client = Client("opc.tcp://user:12345678@1.1.1.53:4840/") #connect using a u
ser
print(client.application_uri)
client.connect()
print("OPC Server uspesne pripojen")
```

```
#pripojeni na SQL Server
conn = pyodbc.connect(sqlstring,timeout=1)


print("SQL Server uspesne pripojen")
print("\n")

blankint=len(df_int.columns)*[0]
blankbool=len(df_bool.columns)*[0]

while True:
    try:
        Timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
        #nacteni postupne vsech bool hodnot promnennych definovanych v df_bool
 z OPC Serveru
        for i in range(len(df_bool.columns)-1):
            node = client.get_node(opcstring+'.%s' %df_bool.columns[i])
            node_value = bool(node.get_value())
            blankbool[i]=node_value

        #nacteni postupne vsech int hodnot promnennych definovanych v df_int z
 OPC Serveru
        for i in range(len(df_int.columns)-1):
            node = client.get_node(opcstring+'.%s' %df_int.columns[i])
            node_value = int(node.get_value())
            blankint[i]=node_value

        blankbool[len(df_bool.columns)-1]=Timestamp
        blankint[len(df_int.columns)-1]=Timestamp
        df_bool=df_bool.append(pd.DataFrame([blankbool],columns=df_bool.column
s),ignore_index=True)
        df_int=df_int.append(pd.DataFrame([blankint],columns=df_int.columns),i
gnore_index=True)

        print("Data uspesne nactena z OPC  " + Timestamp)
        nezapsano = nezapsano +1

        df_int_pro_zapsani=df_int.iloc[-
nezapsano:].to_records(index=False)  #radky dataframe, ktere nebyly zapsany js
ou transf. na list
        df_int_tuple=tuple(u for u in (df_int_pro_zapsani)) #list je pak preme
neny na tuple
        df_bool_pro_zapsani = df_bool.iloc[-
nezapsano:].to_records(index=False) #radky dataframe, ktere nebyly zapsany jso
u transf. na list
        df_bool_tuple = tuple(h for h in (df_bool_pro_zapsani)) #list je pak p
remeneny na tuple
```

```
        try:

            for i in range(nezapsano): #zapisuji se vsechnz dosud nezapsane ra
dky do SQL
                cursor = conn.cursor()
                pro_zapis_int = tuple(df_int_tuple[i-
nezapsano]) #vytvoreni vektoru v spravnem tvaru pro zapis
                cursor.execute('''INSERT INTO PLC.dbo.int VALUES
                               (''' + (len(df_int.columns)-1)*'''?,'''+'''?)
                               ''',pro_zapis_int) #zapisujou se hodnoty odzad
u
                conn.commit()  # poslani to SQL
                #
                cursor = conn.cursor()
                pro_zapis_bool = tuple(df_bool_tuple[i - nezapsano]) #vytvoren
i vektoru v spravnem tvaru pro zapis
                cursor.execute('''INSERT INTO PLC.dbo.bool VALUES
                               (''' + (len(df_bool.columns)-1)*'''?,'''+'''?)
                               ''',pro_zapis_bool)
                conn.commit()  # poslani to SQL

            print("Data uspesne zapsana do SQL " + Timestamp)
            nezapsano=0
            time.sleep(1 )
            print("\n")
        except:
            Timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
            print("SQL nedostupne  " + Timestamp)
            try:
                telnetlib.Telnet(telnetstring, port=sqlport, timeout=1) #zkous
i dostupnost SQL severu (musime zadat spravny port SQL serveru)
                conn = pyodbc.connect(sqlstring, timeout = 1)

            except:
                print("Nepovedlo se obnovit spojeni s SQL  " + Timestamp)
                print("\n")
        time.sleep(4)
    except:
        Timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
        print("OPC nedostupne  " +  Timestamp)
        try:
            client.connect()
            print("Spojeni s OPC obnoveno  " + Timestamp)
            print("\n")
        except:
            print("Nepovedlo se obnovit spojeni s OPC  " + Timestamp)
            print("\n")
        time.sleep(5)
```

SKRIPT – app.py

```
from mypythontools import pyvueeel
from mypythontools.pyvueeel import expose
import pyodbc
import mypythontools
from opcua import Client, ua
import telnetlib
import pandas as pd
import datetime

try:
    client = Client("opc.tcp://localhost:49580")  # if anonymous authenticatio
n is enabled
    client.connect()
    print("OPC Server uspesne pripojen")

except:
    print("OPC Server nedostupny")

try:
    telnetlib.Telnet('localhost', port=49978, timeout=1)

except:
    print("SQL Server nedostupny")

@expose
def set_switch_value(variable, value):
    variable_node = client.get_node('ns=2;s=Paintshop.%s' %variable)
    variable_node.set_value(value)
    print("zapsano na opc")

@expose
def set_slider_value(variable, value):
    variable_node = client.get_node('ns=2;s=Paintshop.%s' %variable)
    datavalue = ua.DataValue(ua.Variant(value, ua.VariantType.Int16))
    variable_node.set_data_value(datavalue)

    #except:
        #try:
            #client = Client("opc.tcp://localhost:49580")  # if anonymous auth
entication is enabled
            #client.connect()
        #except:
            #print("Nedari se navazat spojeni s OPC - nelze ovladat pres web")
```

```
@expose
def get_plot(promenna, pocet_zaznamu,casovani):

    import pandas as pd

    conn = pyodbc.connect(
        "Driver={SQL Server};"  # napojeni na server SQL
        "Server=DESKTOP-LMSTPTV\WINCC;"
        "Database=PLC;"
        "Trusted_Connection=yes;",
        timeout=1,
    )

    query='''SELECT TOP %d %s, %s FROM PLC.dbo.int ORDER BY Timing DESC'''% (p
ocet_zaznamu, promenna,casovani)
    df_query = pd.read_sql_query(query,conn)
    df = pd.DataFrame(df_query, columns=[promenna,casovani])
    return mypythontools.pyvueeel.to_vue_plotly(df)


def get_alarm_df(promenna):

    conn = pyodbc.connect(
        "Driver={SQL Server};"  # napojeni na server SQL
        "Server=DESKTOP-LMSTPTV\WINCC;"
        "Database=PLC;"
        "Trusted_Connection=yes;",
        timeout=1,
    )

      #zdvojnasobi zadany pocet zaznamu, protoze potrebuju zacatek i konec ala
rmu

    query="""select %s,Timing
    from (select *,lead(%s) over(order by Timing desc) as prev_value
    from PLC.dbo.bool
    ) l where prev_value <> %s""" %(promenna,promenna,promenna)

    df_query = pd.read_sql_query(query,conn,parse_dates=["Timing"]) #nacte z S
QL, kde "Timing" je datetime
    df = pd.DataFrame(df_query, columns=[promenna,"Timing"]) #vytvoreni df

    selected_columns = df[[promenna,"Timing","Timing"]] #zdvojnasobi se sloupc
e s casem
    new_df = selected_columns.copy()                    #zdvojnasobi se sloupc
e s casem
    new_df.insert(3, "Duration", int(0), allow_duplicates = True)
```

```
    new_df.columns = ['Location', 'Start time', 'Finish time', 'Duration']  #p
rejmenujou se sloupce

    number_of_rows = len(df.index) #zjisti se pocet radku

    if (new_df.iloc[0]['Location'] == True):
        new_df = new_df.drop(0)
        new_df = new_df.reset_index()

    for i in range(int(number_of_rows/2)):
        new_df.at[i*2,'Start time'] = new_df.iloc[2*i+1]['Start time']    #pro
pisuje se hodnota zacatku naspolecny radek jako hodnota konce alarmu
        try:
            new_df.at[i*2,'Duration'] = datetime.timedelta.total_seconds(new_d
f.iloc[2*i]['Finish time'] - new_df.iloc[2*i]['Start time'])    #propisuje se
hodnota zacatku naspolecny radek jako hodnota konce alarmu
        except: 0

    new_df['Location'] = new_df['Location'].replace([False],promenna)
    new_df=new_df.iloc[::2, :] #odstrani se kazdy druhy radek z dataframu


    new_df['Start time'] = new_df['Start time'].astype(str)
    new_df['Finish time'] = new_df['Finish time'].astype(str)

    return new_df


@expose
def get_table(list_of_alarm_variables):

    final_df = pd.DataFrame()

    for i in range(len(list_of_alarm_variables)):
        df=get_alarm_df(list_of_alarm_variables[i])
        final_df = final_df.append(df,ignore_index = True)

    return mypythontools.pyvueeel.to_table(final_df)

@expose
def load_switch(
    promenna1,
    promenna2,
    promenna3,
    promenna4,
    promenna5,
    promenna6,
    promenna7,
```

```
):
    conn = pyodbc.connect(
        "Driver={SQL Server};"  # napojeni na server SQL
        "Server=DESKTOP-LMSTPTV\WINCC;"
        "Database=PLC;"
        "Trusted_Connection=yes;",
        timeout=1,
    )

    cursor = conn.cursor()
    cursor.execute(
        """SELECT TOP 1 %s,%s,%s,%s,%s,%s,%s FROM PLC.dbo.bool ORDER BY Timing
 DESC"""
        % (
            promenna1,
            promenna2,
            promenna3,
            promenna4,
            promenna5,
            promenna6,
            promenna7,
        )
    )
    rows = cursor.fetchone()
    conn.close()
    return list(rows)

@expose
def load_scada(
    scadacontrol,
):
    conn = pyodbc.connect(
        "Driver={SQL Server};"  # napojeni na server SQL
        "Server=DESKTOP-LMSTPTV\WINCC;"
        "Database=PLC;"
        "Trusted_Connection=yes;",
        timeout=1,
    )

    cursor = conn.cursor()
    cursor.execute(
        """SELECT TOP 1 %s FROM PLC.dbo.bool ORDER BY Timing DESC"""
        % (
            scadacontrol,
        )
    )
    rows = cursor.fetchone()
    conn.close()
```

```
    return list(rows)


@expose
def load_slider(
    promenna1,
    promenna2,
    promenna3,
    promenna4,
    promenna5,
    promenna6,
    promenna7,
):
    conn = pyodbc.connect(
        "Driver={SQL Server};"  # napojeni na server SQL
        "Server=DESKTOP-LMSTPTV\WINCC;"
        "Database=PLC;"
        "Trusted_Connection=yes;",
        timeout=1,
    )

    cursor = conn.cursor()
    cursor.execute(
        """SELECT TOP 1 %s,%s,%s,%s,%s,%s,%s FROM PLC.dbo.int ORDER BY Timing
DESC"""
        % (
            promenna1,
            promenna2,
            promenna3,
            promenna4,
            promenna5,
            promenna6,
            promenna7,
        )
    )
    rows = cursor.fetchone()
    conn.close()
    return list(rows)

@expose
def connect_opc():
    try:
        telnetlib.Telnet('localhost', port=49580, timeout=1)
        client = Client("opc.tcp://localhost:49580")  # if anonymous authentic
ation is enabled
        client.connect()
        boolean = True
```

```
    except:
        boolean = False



    return boolean


# End of file
if __name__ == "__main__":
    pyvueeel.run_gui()
```

SKRIPT – index.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,initial-scale=1.0">
    <link rel="icon" href="<%= BASE_URL %>favicon.ico">
    <script type="text/javascript" src="<%= VUE_APP_EEL %>"></script>

    <title>Moje lakovna GUI</title>
    <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Robot
o:100,300,400,500,700,900">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/@mdi/font@latest
/css/materialdesignicons.min.css">
</head>

<body>
    <noscript>
        <strong>We're sorry but <%= htmlWebpackPlugin.options.title %> doesn't
 work properly without JavaScript enabled.
                Please enable it to continue.</strong>
    </noscript>
    <div id="app"></div>
    <!-- built files will be auto injected -->
</body>

</html>
```

SKRIPT – sliders.vue

```
<template>
  <v-container fluid>
    <v-row>
      <v-slider
        class="slider"
        v-model="D1Speed"
        thumb-label="always"
        step="10"
        min="0"
        max="7000"
        hint="D1 Speed"
        persistent-hint
        @end="ChangeSlider('D1Speed', D1Speed)"
      ></v-slider>
    </v-row>
    <v-row>
      <v-slider
        class="slider"
        v-model="D2Speed"
        thumb-label="always"
        step="10"
        min="0"
        max="7000"
        hint="D2 Speed"
        persistent-hint
        @end="ChangeSlider('D2Speed', D2Speed)"
      ></v-slider>
    </v-row>
    <v-row>
      <v-slider
        class="slider"
        v-model="D3Speed"
        thumb-label="always"
        step="10"
        min="0"
        max="7000"
        hint="D3 Speed"
        persistent-hint
        @end="ChangeSlider('D3Speed', D3Speed)"
      ></v-slider>
    </v-row>
    <v-row>
      <v-slider
        class="slider"
        v-model="D4Speed"
```

```
      thumb-label="always"
      step="10"
      min="0"
      max="7000"
      hint="D4 Speed"
      persistent-hint
      @end="ChangeSlider('D4Speed', D4Speed)"
    ></v-slider>
  </v-row>
  <v-row>
    <v-slider
      class="slider"
      v-model="D5Speed"
      thumb-label="always"
      step="10"
      min="0"
      max="7000"
      hint="D5 Speed"
      persistent-hint
      @end="ChangeSlider('D5Speed', D5Speed)"
    ></v-slider>
  </v-row>
  <v-row>
    <v-slider
      class="slider"
      v-model="P2Speed"
      thumb-label="always"
      step="10"
      min="0"
      max="7000"
      hint="P2 Speed"
      persistent-hint
      @end="ChangeSlider('P2Speed', P2Speed)"
    ></v-slider>
  </v-row>
  <v-row>
    <v-slider
      class="slider"
      v-model="P3Speed"
      thumb-label="always"
      step="10"
      min="0"
      max="7000"
      hint="P3 Speed"
      persistent-hint
      @end="ChangeSlider('P3Speed', P3Speed)"
    ></v-slider>
  </v-row>
```

```
  </v-container>
</template>

<script>
export default {
  data() {
    return {
      D1Speed: 6000,
      D2Speed: 6000,
      D3Speed: 6000,
      D4Speed: 6000,
      D5Speed: 6000,
      P2Speed: 6000,
      P3Speed: 6000,
      OpcArray: {
        a: false,
        b: false,
        c: false,
        d: false,
        e: false,
        f: false,
        g: false,
      },
      TimerVar: "",
    };
  },

  methods: {
    LoadOpcState() {
      window.eel.load_slider(
        "D1Speed",
        "D2Speed",
        "D3Speed",
        "D4Speed",
        "D5Speed",
        "P2Speed",
        "P3Speed"
      )((result) => {
        this.OpcArray.a = result[0];
        this.OpcArray.b = result[1];
        this.OpcArray.c = result[2];
        this.OpcArray.d = result[3];
        this.OpcArray.e = result[4];
        this.OpcArray.f = result[5];
        this.OpcArray.g = result[6];
      });
    },
```

```
    UpdateSliders() {
      this.D1Speed = this.OpcArray.a;
      this.D2Speed = this.OpcArray.b;
      this.D3Speed = this.OpcArray.c;
      this.D4Speed = this.OpcArray.d;
      this.D5Speed = this.OpcArray.e;
      this.P2Speed = this.OpcArray.f;
      this.P3Speed = this.OpcArray.g;
    },

    ChangeSlider(variable, value) {
      window.clearTimeout(this.TimerVar);
      window.eel.set_slider_value(variable, value);
      this.TimerVar = window.setTimeout(() => {
        this.UpdateSliders();
      }, 5000);
    },
  },

  mounted: function () {
    this.LoadOpcState();
    this.UpdateSliders();
    window.setInterval(() => {
      this.LoadOpcState();
    }, 1000);
  },

  watch: {
    OpcArray: {
      handler: function () {
        this.UpdateSliders();
      },
      deep: true,
    },
  },
};
</script>

<style scoped>
.slider {
  height: 66px;
}
</style>
```