FAKULTA ELEKTROTECHNICKÁ ZÁPADOČESKÉ UNIVERZITY V PLZNI

RICE

Fakulta elektrotechnická
Regionální inovační centrum elektrotechniky

Možnosti tvorby grafického uživatelského rozhraní fyzikálních modelů ze softwaru Dymola

Pracoviště:	RICE - Energetika a průmyslové systémy
Číslo dokumentu:	22190-028-2018
Typ zprávy:	Výzkumná zpráva
Řešitelé:	Ing. Aleš Hromádka, Ing. Martin Sirový, Ph.D., Ing. Martin Vinš
Hlavní řešitel:	Ing. Martin Sirový, Ph.D.
Počet stran:	14
Datum vydání:	24. 9. 2018
Oborové zařazení:	**JF – Jaderná energetika**

Zpracovatel / dodavatel:

Západočeská univerzita v Plzni
Regionální inovační centrum elektrotechniky
Univerzitní 8
306 14 Plzeň

Kontaktní osoba:

Ing. Aleš Hromádka
tel. 377 634 106
aleshrom@rice.zcu.cz

Tato výzkumná zpráva byla vytvořena za podpory projektu CANUT TE01020455 a studentského projektu SGS-2018-009.

Anotace

Tato výzkumná zpráva se zabývá možnosti tvorby grafického uživatelského rozhraní fyzikálních modelů ze softwaru Dymola. Tyto možnosti jsou naznačeny pomocí dvou již vytvořených rozhraní pro fyzikální modely z Dymoly. Zaprvé je zde detailně popsána tvorba rozhraní pro model elektrického odporu v prostředí C#. Jako druhý je zde opět detailně popsána tvorba rozhraní modelu jednostupňové turbíny pomocí aplikace AppDesigner, která je součástí prostředí Matlab/Simulink.

Klíčová slova

Grafický uživatelské rozhraní, Fyzikální model, Matlab/Simulink, #C, AppDesigner.

Report title

Possibilities of creation of graphical user interface for physical models from the software Dymola. These options are shown by the already created Graphical User Interfaces for physical models from the Dymola.

Abstract

This research report concerns the options of the creation of the Graphical User Interface for physical models from the software Dymola. These options are shown by two already created Graphical User Interfaces for certain physical models from the Dymola. Firstly, here is in more details described the creation of the interface for the model of electrical resistance in the language C# . As the second way, here is in more details describes the creation of interface for the model of the one stage turbine by the tool App Designer, which is included to Matlab/Simulink.

Klíčová slova v anglickém jazyce / Keywords

Graphical User Interface, Physical Model, Matlab/Simulink, #C, AppDesigner.

Obsah

ANOTACE 2
KLÍČOVÁ SLOVA 2
REPORT TITLE 2
ABSTRACT 2
KLÍČOVÁ SLOVA V ANGLICKÉM JAZYCE / KEYWORDS 2
SEZNAM SYMBOLŮ A ZKRATEK 4
1 ÚVOD 5
2 MOŽNOSTI EXPORTU ZE SOFTWARU DYMOLA 6
3 GRAFICKÉ UŽIVATELSKÉ PROSTŘEDÍ VYTVOŘENÉ V VISUAL STUDIO C# 7
3.1 POUŽITÝ MODEL 7
3.2 POSTUP TVORBY GUI V JAZYCE C# 8
4 GRAFICKÉ UŽIVATELSKÉ PROSTŘEDÍ VYTVOŘENÉ V APPDESIGNERU (MATLAB/SIMULINK) 9
4.1 POUŽITÝ MODEL 9
4.2 POSTUP TVORBY GUI V APPDESIGNERU 10
5 ZÁVĚR 12
LITERATURA 14
SEZNAM OBRÁZKŮ 14
PŘÍLOHY 15

Seznam symbolů a zkratek

GUI	Graphical User Interface
FMU	Functional Mock-up Unit
Matlab	Matrix laboratories
Dymola	Dynamic Modeling Laboratories
.txt	Textový formát
.mat	Tabulkový formát
.exe	Vykonávací formát

1 Úvod

Software Dymola nabízí celou řadu knihoven, které obsahují fyzikální modely pro různé obory jako je například termodynamika, hydraulika, aerodynamika nebo třeba elektronika. Nicméně, interpretace výsledků simulací pro zadavatele, který nemá licencovaný tento program je velmi obtížná. Z tohoto důvodu vznikla tato výzkumná zpráva zabývající možnostmi a popisem tvorby GUI pro konkrétní fyzikální modely z knihoven softwaru Dymola. Software Dymola nabízí dvě cesty, jak exportovat fyzikální model mimo toto prostředí. Jedná se o export například do formátů:

- FMU (možný export do jiných programů např. Simulink)
- Dymosin.exe (pomocí .txt souborů si přečte vstupy a zapíše výstupy)

Existují i další možnosti exportu ze softwaru Dymola, nicméně tyto cesty podléhají zvláštní licenci. Pro tuto výzkumnou zprávu byly vybrány obě zmíněné cesty exportu fyzikálních modelů. První cesta exportu je přes již zmíněný exportovaný soubor s názvem dymosim.exe, který obsahuje kompilovaný fyzikální model v jazyce C včetně hlavičkových souborů (.h). Druhou cestou exportu z Dymoly je přes modelový balíček FMU, který umožňuje převedení model do jiného prostředí například do Matlabu/Simulinku. Export do těchto formátu není nikterak složitý a bude v následujícím textu.

Dále je pro tvorbu GUI použito dvou grafických prostředí. První z nich bylo využito prostředí Visual studio C#, kde byl řešen příklad elementárního elektrického odporu, u kterého byl výstupem ztrátový odpor. Jako druhé prostředí pro tvorbu GUI aplikace bylo zvoleno prostředí AppDesigner, který je rozšířením novějších verzí Matlabu/Simulinku. Pro ukázku v tomto prostředí jsme zvolili o něco složitější příklad jednostupňové turbíny, u které jsou voleny jako vstupy vstupní a výstupní termodynamické parametry a výstupem je elektrický výkon generátoru.

Závěrem této výzkumné zprávy posouzení výhodnosti použitých prostředí pro různé fyzikální modely a jejich omezení.

2 Možnosti exportu ze softwaru Dymola

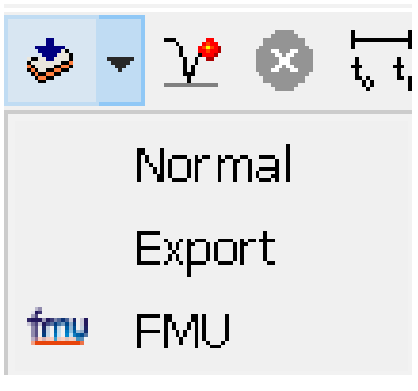
Software Dymola nabízí dva základní módy, kterými jsou viz 1.1.1Obr. 1: [1]

- „Modeling“
- „Simulation“

Obr. 1. *Základní módy v prostředí Dymola [1]*

Pro ukázání možností exportu fyzikálních modelů je nezbytné se nacházet v módu „Simulation“. V módu „Simulation“ se nachází horní lišta s možnostmi a mimo jiné také se záložkou „Translate“. Záložka „Translate“ zahrnuje následující možnosti, viz Obr. 2:

- „Normal“ – tři textové soubory dsin, dslog a dsout.txt + dymosin.exe.
- „Export“ – cesta export podléhající zvláštní licenci.
- „FMU“ – modelový balíček, který lze využít v jiných prostředích např. Matlab/Simulink. [1]



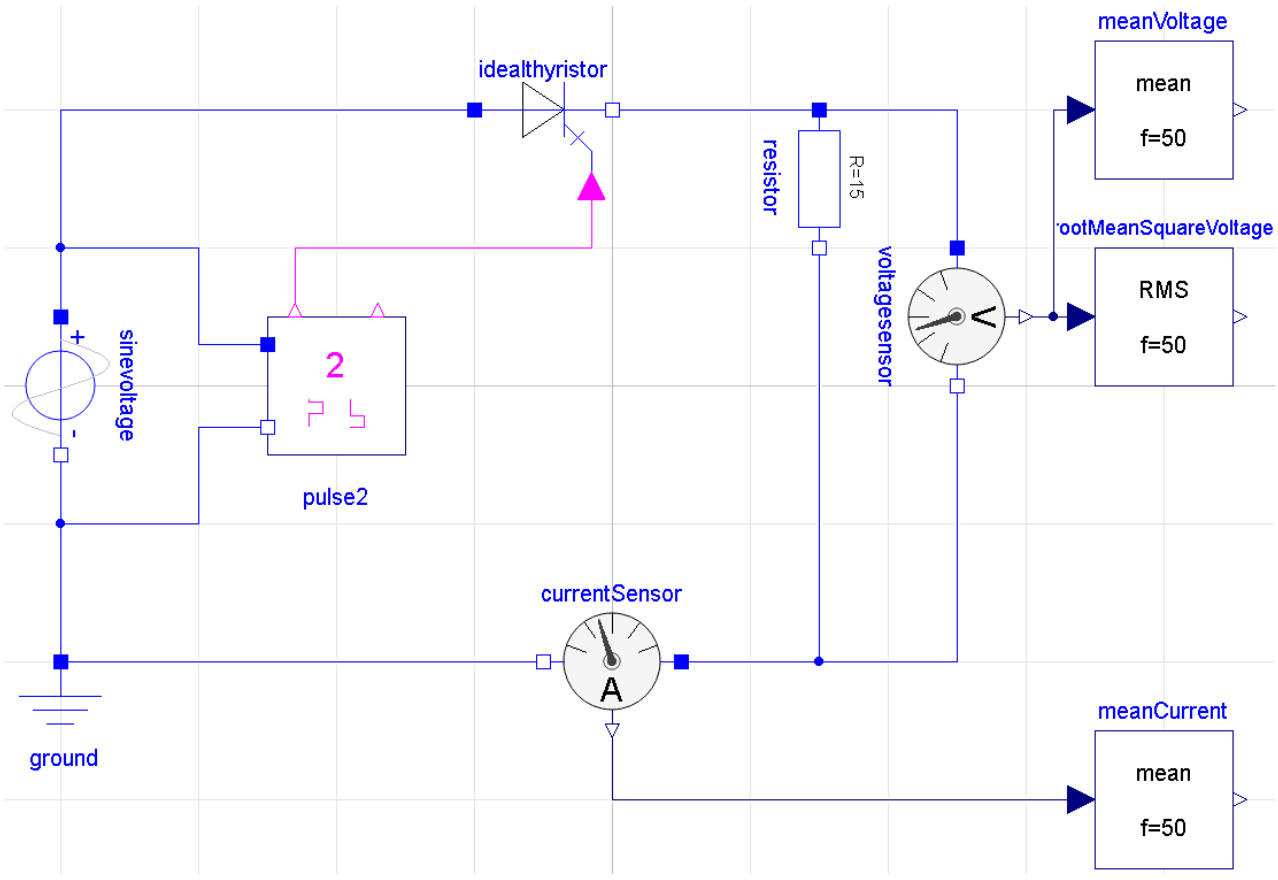
Obr. 2. *Možnosti překladu fyzikálního modelu v prostředí Dymola [1]*

Pro další zkoumání jsme využili obou dosažitelných exportů ze softwaru Dymola tedy možnosti „Normal“ a „FMU“ . [1]

3 Grafické uživatelské prostředí vytvořené v Visual studio C#

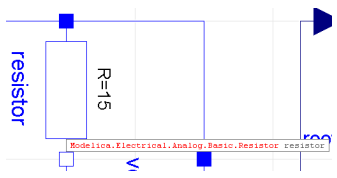
3.1 Použitý model

Pro maximální jednoduchost byl zvolen příklad jednocestného usměrňovače jdoucí přes čistě ohmický odpor, viz Obr. 3.



Obr. 3. *Využitý model jednocestného řízeného usměrňovače pro tvorbu GUI v C#*

Nicméně, pro tvorbu GUI byl tento model zredukován na samotný ohmický odpor, což vstupem je ohmický odpor a výstupem je ztrátový výkon tohoto odporu, viz Obr. 4.



Obr. 4. *Ohmický odpor pro tvorbu GUI v prostředí C#*

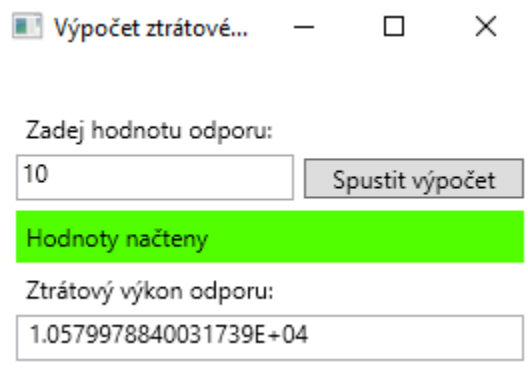
3.2 Postup tvorby GUI v jazyce C#

Jako první krok je nutné vyexportovat fyzikální model mimo prostředí Dymola. To bylo provedeno v souladu s bodem 2. Respektive byl použit typ exportu „Normal“ a byly vytvořeny příslušné soubory: [1]

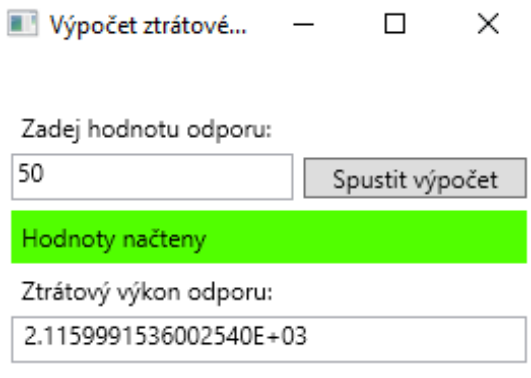
- Dymosin.exe,
- dsin.txt,
- dlog.txt,
- dsout.txt.

Pomocí těchto souborů jsme schopni libovolně měnit údaje v simulaci, respektive když změníme libovolný údaj v souboru dsin.txt a spustíme soubor Dymosin.exe výsledky přepočítaných výstupů se uloží do souboru dsout.txt.

Tohoto mechanismu využívá vytvořené GUI v Visual studiu C#. Základní myšlenka vytvořeného GUI je načíst originální dsin.txt a měnit konkrétní parametr přepsáním stávajícího do nového souboru. Po tomto kroku spustit samotnou simulaci pomocí souboru Dymosim.exe a z nově vzniklého souboru dsout.txt vyčíst hledaný parametr a vypsat. Celý skript GUI v rámci prostředí Visual studio C# je přiložen k této výzkumné zprávě jako příloha č. 1.



Obr. 5. *Ukázka GUI vytvořeného v jazyce C#*

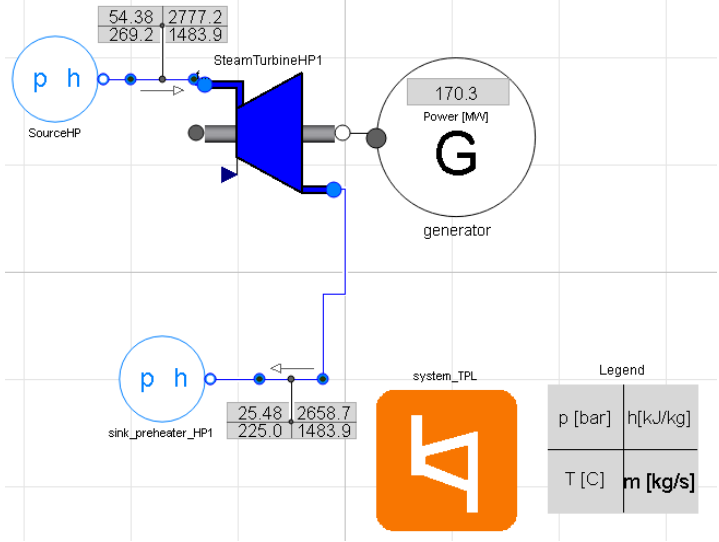


Obr. 6. *Ukázka GUI vytvořeného v jazyce C#*

4 Grafické uživatelské prostředí vytvořené v AppDesigneru (Matlab/Simulink)

4.1 Použitý model

Je nutné říci, že toto prostředí AppDesigneru umožňuje velice i pokročilejší funkce než předchozí případ kde jsme se omezili pouze na vypisování prosté hodnoty. Zde byl pro ukázku přiložen také graf, do kterého lze vykreslovat hodnoty. Proto byl zvolen o něco složitější model jednostupňové turbíny, viz Obr. 7.



Obr. 7. *Využitý model jednostupňové turbíny pro tvorbu GUI v AppDesigneru*

Pro uvažovaný model turbíny bylo zvoleno sedm vstupních parametrů, které jsou zaměnitelné v GUI, kterými jsou:

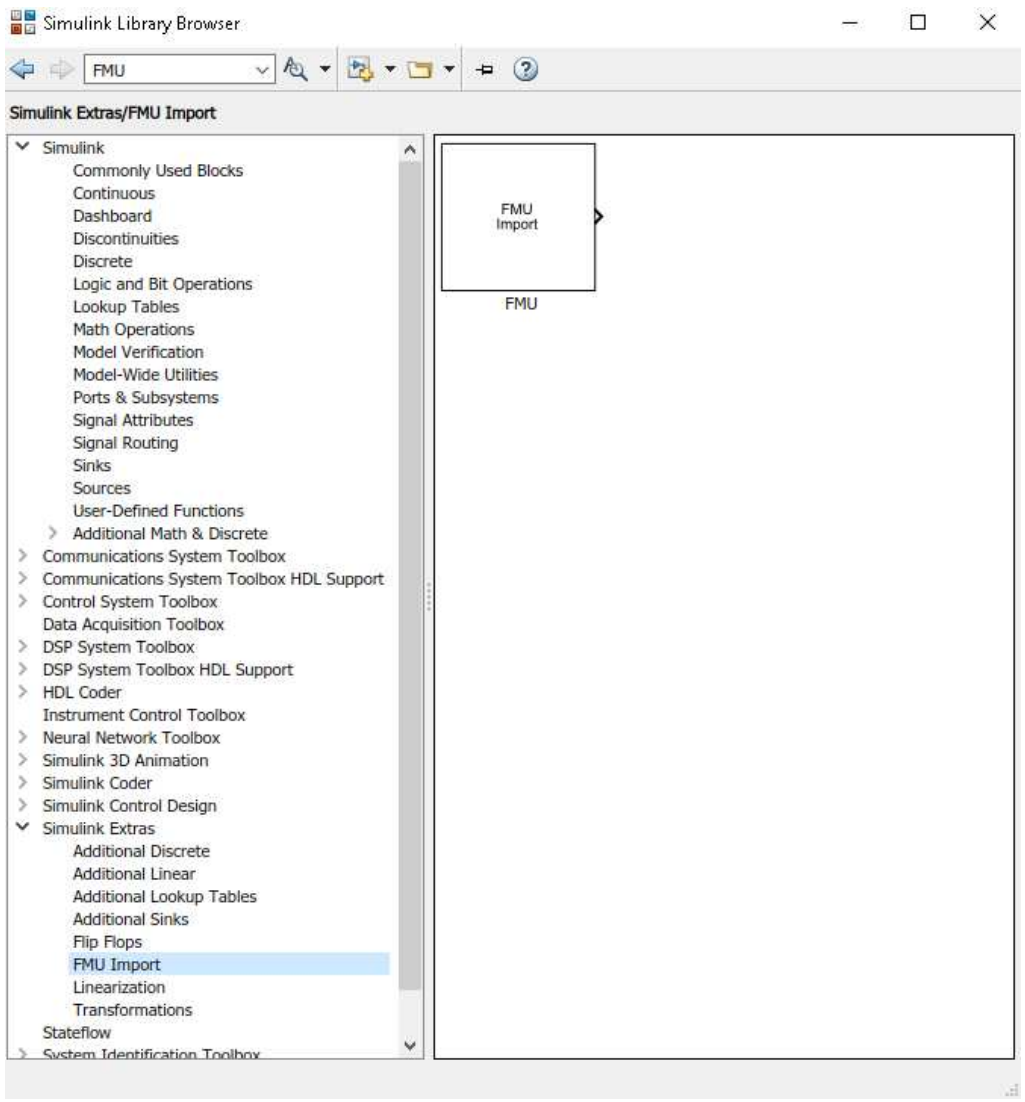
- vstupní tlak [bar],
- vstupní entalpie [kJ/kg],
- mechanická účinnost [-],
- termodynamická účinnost [-],
- výstupní tlak [bar],
- výstupní entalpie [kJ/kg].

Výstupem tohoto modelu je potom vypočítaný elektrický výkon na svorkách generátoru v megawattech. Tento výstup je také vyobrazen v grafu, nicméně vzhledem k tomu, že vždy uvažujeme jeden časový okamžik, zobrazovaný výkon se vždy projeví jako vodorovná čára. Na druhou stranu pro názornost tohoto příkladu je toto zjednodušení nepodstatné.

4.2 Postup tvorby GUI v AppDesigneru

Stejně jako v předchozím případě prvním krokem je nutné vyexportovat fyzikální model mimo prostředí Dymola. To bylo opět provedeno v souladu s bodem 2. Nicméně v tomto případě byl použit typ exportu „FMU“. Export do formátu „FMU“ umožňuje pracovat s celistvým modelem v rámci přepisovatelných struktur jako vstupních parametrů. Výstupní parametry jsou vypisovány do souboru s příponou .mat.

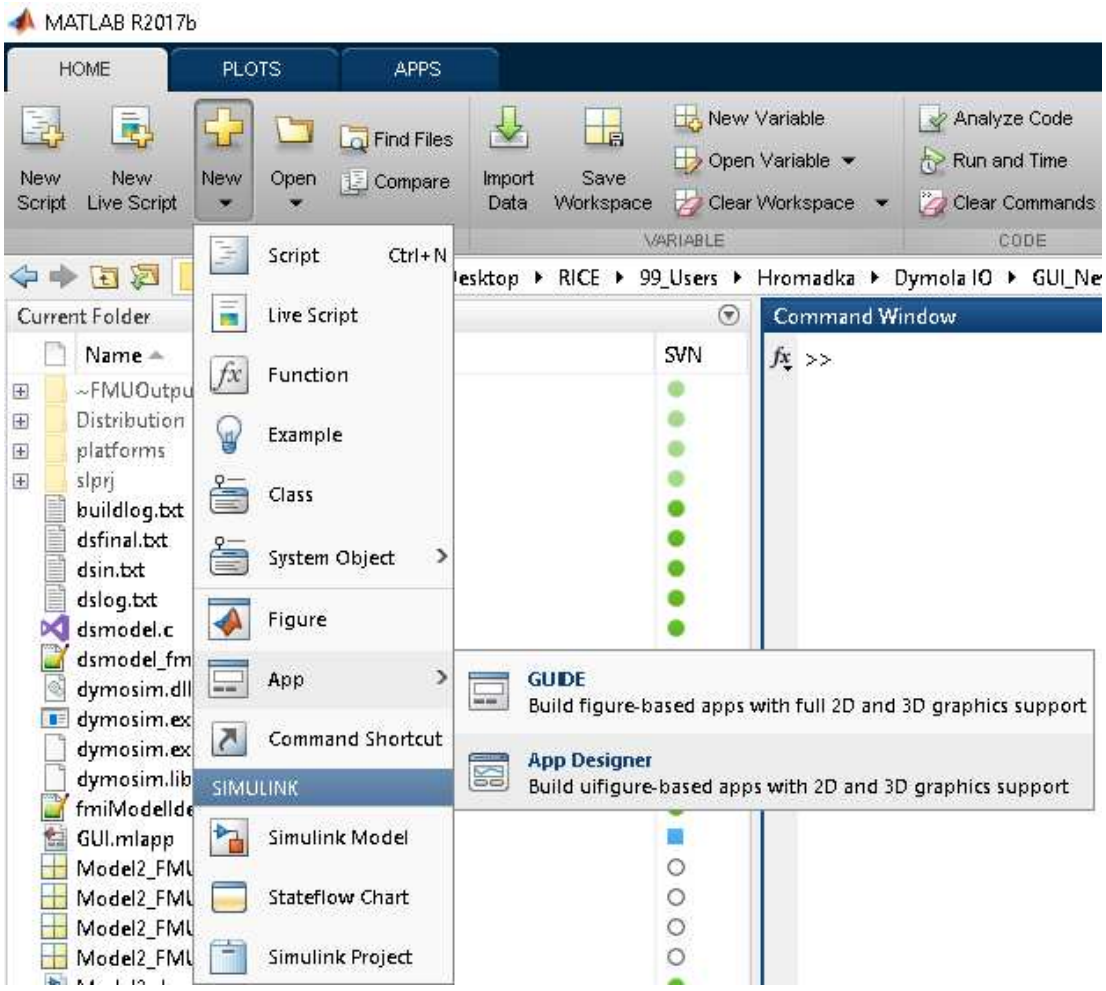
První nezbytným krokem bylo nalézt cestu, jak v prostředí Matlab přečíst soubor „FMU“. Bylo zjištěno, že nových verzí Matlabu myšleno 2017b a novější nabízí Simulink (nadstavba Matlabu) velice jednoduchý nástroj jak importovat fyzikální model z Dymoly pomocí vloženého bloku s názvem „Import FMU“ V tomto bloku lze pomocí struktur zaměňovat libovolně parametry simulovaného modelu.



Obr. 8. *Ukázka cesty k „Import FMU“*

Pomocí tohoto nástroje jsme schopni importovat celou simulaci uvažovaného modelu do prostředí Matlabu/Simulinku.

Nyní se zaměříme na samotnou tvorbu GUI v další nadstavbě Matlabu pro tvorbu uživatelských rozhraní s názvem AppDesigner.



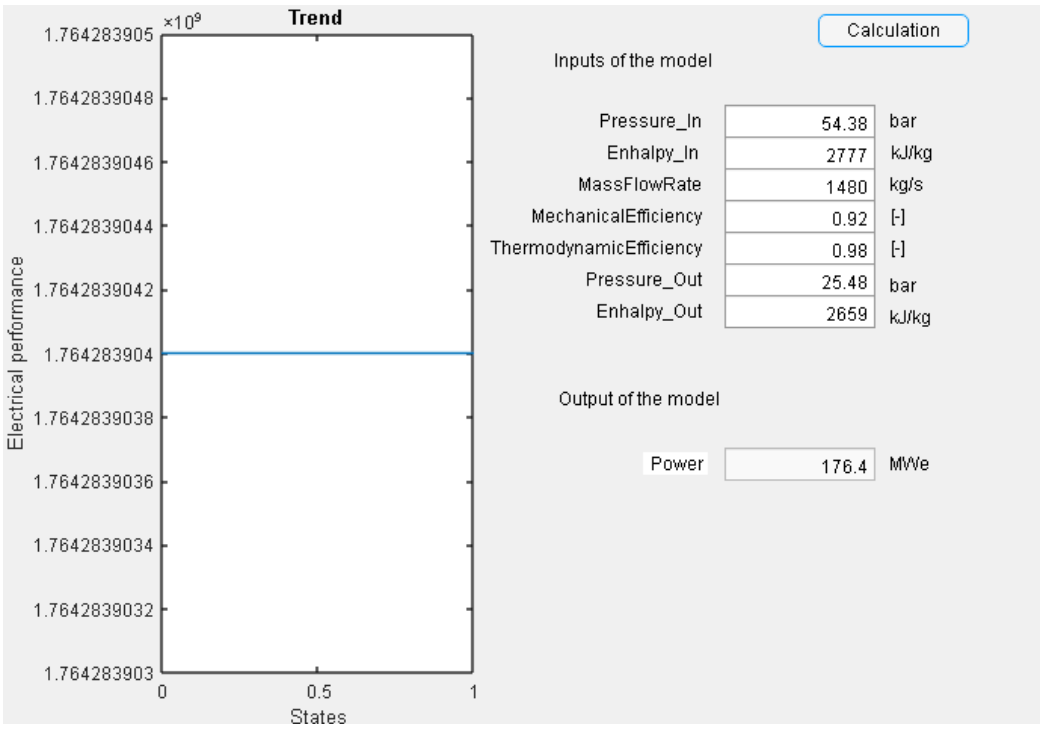
Obr. 9. *Ukázka cesty k nástroji AppDesigner*

Pomocí tohoto nástroje jsme schopni velmi jednoduše sestavit uživatelské rozhraní pro libovolný exportovaný model z Dymoly.

Základní myšlenka tvořeného GUI pro uvažovaný model je, že se přepisovatelné struktury v importovaném modelu v Simulinku nahradí strukturami uloženými v GUI. Tyto struktury v GUI mají přepisovatelné pozice, pomocí kterých jsou zadávány proměnné vstupní do modelu.

Po zadání vstupních proměnných pomocí GUI zavoláme importovaný model ze Simulinku se zadanými parametry a necháme spočítat. Výsledky modelu se zadanými parametry jsou uloženy do již míněného souboru s příponou .mat. Z tohoto soboru lze velmi jednoduše vyčítat výsledky modelu stačí najít příslušný řádek, který odpovídá konkrétnímu výsledku a zobrazit v okně hledaného

výsledku. Ukázka GUI tvořené pomocí AppDesigneru je zobrazena níže viz Obr. 10. Celistvý kód skriptu, který reprezentuje tvorbu GUI v prostředí AppDesigner viz příloha č.2.



Obr. 10. *Ukázka GUI vytvořeného v prostředí AppDesigneru (Matlab)*

5 Závěr

Díky zde ukázaným možnostem tvorby GUI jsme schopni relativně snadno pracovat exportovanými z Dymoly na jiných platformách. Což je i hlavní cílem této výzkumné zprávy.

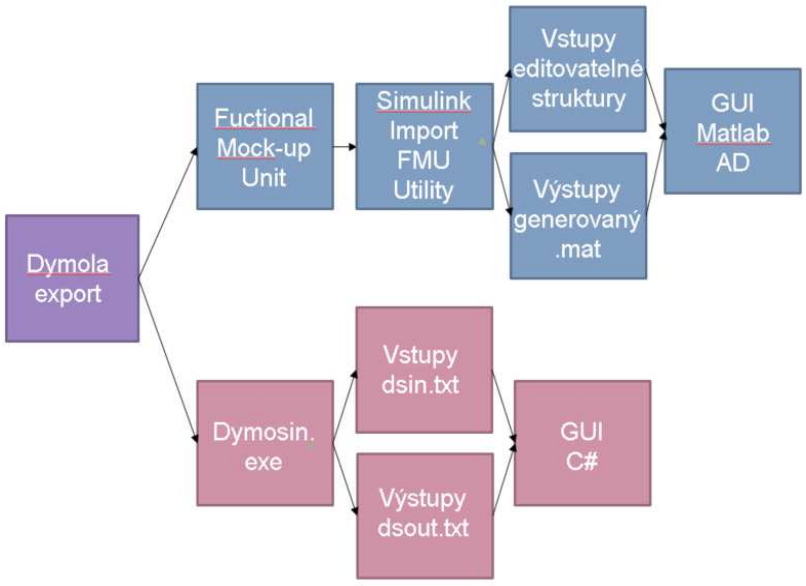
Pro upřesnění obou vytvořených GUI je zde ještě ve zkratce popsán postup tvorby a hlavní myšlenky:

Nejprve bude bodově popsána tvorba GUI v jazyce C#:

- Využití exportu dymosimu.exe
  - .exe Soubor
    - Kompilovaný model z Dymoly do jazyka C
    - Vstupy pro model .txt a výstupy zapisované také do .txt
  - Využití jazyka C# pro tvorbu jednoduchého rozhraní
    - Vstupy zapisovány do .txt a z výstupního .txt vyčítány výsledky
    - Pozor omezení pouze na hodnoty!!! Grafické znázornění je relativně složitě vytvářeno třetími programy.

Druhé GUI bylo vytvářeno pomocí AppDesigneru (Matlab/Simulink) a opět bude bodově popsáno:

- Využití exportu FMU
  - Functional Mock-up Unit (generovaný Dymolou)
  - Matlab/Simulink 2017b (podpora pro čtení FMU)
    - Výsledky každé FMU simulace zapsány v .mat souboru
  - Tvorba samotného GUI Matlab AppDesigneru
    - Vstupy řešeny zaměnitelnými strukturami a výstupy čteny z .mat souboru.
    - Lze snadno zakreslovat výsledky i do grafů, což je velká výhoda.



Obr. 11. *Zjednodušené grafické znázornění obou cest tvorby GUI [1]*

Literatura

[1] *Dymola Release Notes*, Manheim 2017.

Seznam obrázků

Obr. 1. Základní módy v prostředí Dymola 6
Obr. 2. Možnosti překladu fyzikálního modelu v prostředí Dymola 6
Obr. 3. Využitý model jednocestného řízeného usměrňovače pro tvorbu GUI v prostředí C#7
Obr. 4. Ohmický odpor pro tvorbu GUI v prostředí C# 7
Obr. 5. Ukázka GUI vytvořeného v jazyce C# 8
Obr. 6. Ukázka GUI vytvořeného v jazyce C# 9
Obr. 7. Využitý model jednostupňové turbíny pro tvorbu GUI v AppDesigneru 9
Obr. 8. Ukázka cesty k „Import FMU“ 10
Obr. 9. Ukázka cesty k nástroji AppDesigner 11
Obr. 10. Ukázka GUI vytvořeného v prostředí AppDesigneru (Matlab) 12
Obr. 11. Doby odstávek v JETE za jednotlivé zkoumané roky [1] 13

Přílohy

Příloha č.1: Skript pro GUI v prostředí C#

```
using Microsoft.Win32;
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace DymolaImport
{/// <summary>
 /// Interakční logika pro MainWindow.xaml
    public partial class MainWindow : Window
    {
        string slozka;
        public MainWindow()
        {
            InitializeComponent();
        }
             private void Button_Click(object sender, RoutedEventArgs e)
        {
            OpenFileDialog soubor = new OpenFileDialog();
            soubor.FileName = "*sablona*";
            soubor.ShowDialog();
            slozka = System.IO.Path.GetDirectoryName(soubor.FileName);
            //string text = File.ReadAllText("dsin.txt");
            string text = File.ReadAllText(soubor.FileName);
            string input = txtInputResist.Text;
            input = " " + input;
            int DelkaInputu = input.Length;
            while (DelkaInputu < 26)
            {
                input = input + " ";
                DelkaInputu = input.Length;
                Debug.Write(DelkaInputu);
            }

            string vystupniSoubor = slozka + "\\dsin.txt";

            text = text.Replace("      TADYTOTO            ", input);
            File.WriteAllText(vystupniSoubor, text);
            //MessageBox.Show("Hotovo");
            string cestaKDymosimu = slozka + "\\dymosim.exe";
            Debug.Write(cestaKDymosimu);
            //System.Diagnostics.Process.Start(cestaKDymosimu, "> dymosim-
debug.txt");
            Process process = new Process();
            process.StartInfo.FileName = cestaKDymosimu;
            //process.StartInfo.Arguments = "/c DIR"; // Note the /c command (*)
            process.StartInfo.UseShellExecute = false;
            process.StartInfo.RedirectStandardOutput = true;
            process.StartInfo.RedirectStandardError = true;
            process.StartInfo.WorkingDirectory = slozka;
```

```
            process.Start();//* Read the output (or the error)
            string output = process.StandardOutput.ReadToEnd();
            //MessageBox.Show(output);
            string err = process.StandardError.ReadToEnd();
            Console.WriteLine(err);
            process.WaitForExit();
            lblReady.Visibility = Visibility.Visible;
            string cilovaRadka = "";
            string cestaKDsfinal = slozka + "\\dsfinal.txt";
            using (StreamReader sr = File.OpenText(cestaKDsfinal))
            {
                string s = String.Empty;

                while ((s = sr.ReadLine()) != null)
                {
                    if (s.Contains("resistor.LossPower"))
                    {
                        cilovaRadka = s;
                    }
                }
            }
            string vystup = cilovaRadka.Substring(4, 24);
            lblOutputDescription.Visibility = Visibility.Visible;
            txtOutput.Visibility = Visibility.Visible;
            txtOutput.Text = vystup;
            txtOutput.IsReadOnly = true;
        }

        private void Button_Click_1(object sender, RoutedEventArgs e)
        {
            string cilovaRadka = "";
            string cestaKDsfinal = slozka + "\\dsfinal.txt";
            using (StreamReader sr = File.OpenText(cestaKDsfinal))
            {
                string s = String.Empty;

                while ((s = sr.ReadLine()) != null)
                {
                    if (s.Contains("resistor.LossPower"))
                    {
                        cilovaRadka = s;

                    }
                }
            }
            string vystup = cilovaRadka.Substring(4, 24);
            MessageBox.Show(vystup);
        }

        private void TextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
        }
    }
}
```

Příloha č. 2: Skript v AppDesigneru (Matlab)

```
classdef GUI < matlab.apps.AppBase
    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                        matlab.ui.Figure
        UIAxes                          matlab.ui.control.UIAxes
        CalculationButton_4             matlab.ui.control.Button
        PowerEditFieldLabel             matlab.ui.control.Label
        PowerEditField                  matlab.ui.control.NumericEditField
        Pressure_InEditFieldLabel       matlab.ui.control.Label
        Pressure_InEditField            matlab.ui.control.NumericEditField
        Enhalpy_InEditFieldLabel        matlab.ui.control.Label
        Enhalpy_InEditField             matlab.ui.control.NumericEditField
        OutputofthemodelLabel           matlab.ui.control.Label
        MWeLabel                        matlab.ui.control.Label
        barLabel                        matlab.ui.control.Label
        kJkgLabel                       matlab.ui.control.Label
        barLabel_2                      matlab.ui.control.Label
        kJkgLabel_2                     matlab.ui.control.Label
        Pressure_OutEditFieldLabel      matlab.ui.control.Label
        Pressure_OutEditField           matlab.ui.control.NumericEditField
        Enhalpy_OutEditFieldLabel       matlab.ui.control.Label
        Enhalpy_OutEditField            matlab.ui.control.NumericEditField
        InputsofthemodelLabel           matlab.ui.control.Label
        kgsLabel                        matlab.ui.control.Label
        Label                           matlab.ui.control.Label
        MassFlowRateEditFieldLabel      matlab.ui.control.Label
        MassFlowRateEditField           matlab.ui.control.NumericEditField
        MechanicalEfficiencyEditFieldLabel  matlab.ui.control.Label
        MechanicalEfficiencyEditField  matlab.ui.control.NumericEditField
        ThermodynamicEfficiencyEditFieldLabel  matlab.ui.control.Label
        ThermodynamicEfficiencyEditField  matlab.ui.control.NumericEditField
        Label_5                         matlab.ui.control.Label
    end
    properties (Access = private)
        xdata % Description
        ydata % Description
    end
    methods (Access = private)
        % Value changed function: PowerEditField
        function PowerEditFieldValueChanged(app, event)
            value = app.PowerEditField.Value;
        end
        % Callback function: CalculationButton_4, UIFigure
        function CalculationButton_4Pushed(app, event)
             persistent step;
            if isempty(step)
                step = 0;
            end
            if step == 0
                filename = 'Model2_FMU_internal.mat';
            else
                filename=strcat('Model2_FMU_internal',num2str(step),'.mat');
            end

            struct_input =
struct('pressureBoundary',struct('T',298.15,'X',[1.0],'Tc_port_state',0.1,'steady_s
tate_port_init',boolean(false),'enable_m_flow_iv',[boolean(true)],'enable_p_res',bo
olean(false)),'p0',app.Pressure_InEditField.Value*1000000,'h0',app.Enhalpy_InEditFi
eld.Value*1000);
            struct_turbine =
struct('pstartin',5439000,'pstart',2548000,'hstartin',2777200.0,'hstartout',2658700
.0,'eta_mech',
app.MechanicalEfficiencyEditField.Value,'eta_is_nom',app.ThermodynamicEfficiencyEdi
tField.Value,'Kt',0.003,'m_flow_nom',app.MassFlowRateEditField.Value,'h1_nom',27772
00.0,'p1_nom',5438000,'p2_nom',2548000,'alpha_Bauman',1);
```

```
            struct_output =
struct('pressureBoundary',struct('T',298.15,'X',[1.0],'Tc_port_state',0.1,'steady_s
tate_port_init',boolean(false),'enable_m_flow_iv',[boolean(true)],'enable_p_res',bo
olean(false)),'p0',app.Pressure_OutEditField.Value*1000000,'h0',app.Enhalpy_OutEdit
Field.Value*1000);
            save 'model_prm';
%             parameters = struct(struct_input, struct_turbine, struct_output);
            sim Model2.slx;

            a=dir;
            if length(a)==3
                Iteration=3;
            else
                Iteration = 3;
                for n = 4:length(a)-18
                    x=a(n).datenum;
                    y=a(Iteration).datenum;
                    if x>y
                        Iteration=n;
                    end
                end
            end
            load (a(Iteration).name)
            aa=data_1(108,2);
            app.PowerEditField.Value=aa/10000000;
            app.xdata = [0,1];
            app.ydata = [data_1(108,1),data_1(108,2)];
            plot(app.UIAxes,app.xdata,app.ydata);
            step = step + 1;
        end
    end
    % App initialization and construction
    methods (Access = private)
        % Create UIFigure and components
        function createComponents(app)
            % Create UIFigure
            app.UIFigure = uifigure;
            app.UIFigure.Position = [100 100 696 480];
            app.UIFigure.Name = 'UI Figure';
            app.UIFigure.SizeChangedFcn = createCallbackFcn(app,
@CalculationButton_4Pushed, true);
            % Create UIAxes
            app.UIAxes = uiaxes(app.UIFigure);
            title(app.UIAxes, 'Trend ')
            xlabel(app.UIAxes, 'States')
            ylabel(app.UIAxes, 'Electrical performance')
            app.UIAxes.Box = 'on';
            app.UIAxes.Position = [1 1 320 480];
            % Create CalculationButton_4
            app.CalculationButton_4 = uibutton(app.UIFigure, 'push');
            app.CalculationButton_4.ButtonPushedFcn = createCallbackFcn(app,
@CalculationButton_4Pushed, true);
            app.CalculationButton_4.Position = [541 452 100 22];
            app.CalculationButton_4.Text = 'Calculation';
            % Create PowerEditFieldLabel
            app.PowerEditFieldLabel = uilabel(app.UIFigure);
            app.PowerEditFieldLabel.BackgroundColor = [1 1 1];
            app.PowerEditFieldLabel.HorizontalAlignment = 'right';
            app.PowerEditFieldLabel.Position = [425 169 43 15];
            app.PowerEditFieldLabel.Text = 'Power ';
            % Create PowerEditField
            app.PowerEditField = uieditfield(app.UIFigure, 'numeric');
            app.PowerEditField.Limits = [0 3000];
            app.PowerEditField.ValueChangedFcn = createCallbackFcn(app,
@PowerEditFieldValueChanged, true);
            app.PowerEditField.Editable = 'off';
            app.PowerEditField.Position = [479 165 100 22];
            % Create Pressure_InEditFieldLabel
```

```
app.Pressure_InEditFieldLabel = uilabel(app.UIFigure);
app.Pressure_InEditFieldLabel.HorizontalAlignment = 'right';
app.Pressure_InEditFieldLabel.Position = [391 396 73 15];
app.Pressure_InEditFieldLabel.Text = 'Pressure_In';
% Create Pressure_InEditField
app.Pressure_InEditField = uieditfield(app.UIFigure, 'numeric');
app.Pressure_InEditField.Limits = [0.0001 600];
app.Pressure_InEditField.Position = [479 392 100 22];
app.Pressure_InEditField.Value = 54.38;
% Create Enhalpy_InEditFieldLabel
app.Enhalpy_InEditFieldLabel = uilabel(app.UIFigure);
app.Enhalpy_InEditFieldLabel.HorizontalAlignment = 'right';
app.Enhalpy_InEditFieldLabel.Position = [396 375 66 15];
app.Enhalpy_InEditFieldLabel.Text = 'Enhalpy_In';
% Create Enhalpy_InEditField
app.Enhalpy_InEditField = uieditfield(app.UIFigure, 'numeric');
app.Enhalpy_InEditField.Limits = [200 4000];
app.Enhalpy_InEditField.Position = [479 371 100 22];
app.Enhalpy_InEditField.Value = 2777.2;
% Create OutputofthemodelLabel
app.OutputofthemodelLabel = uilabel(app.UIFigure);
app.OutputofthemodelLabel.Position = [369 212 115 15];
app.OutputofthemodelLabel.Text = 'Output of the model ';
% Create MWeLabel
app.MWeLabel = uilabel(app.UIFigure);
app.MWeLabel.Position = [588 169 32 15];
app.MWeLabel.Text = 'MWe';
% Create barLabel
app.barLabel = uilabel(app.UIFigure);
app.barLabel.Position = [588 396 25 15];
app.barLabel.Text = 'bar';
% Create kJkgLabel
app.kJkgLabel = uilabel(app.UIFigure);
app.kJkgLabel.Position = [589 375 33 15];
app.kJkgLabel.Text = 'kJ/kg';
% Create barLabel_2
app.barLabel_2 = uilabel(app.UIFigure);
app.barLabel_2.Position = [588 287 25 15];
app.barLabel_2.Text = 'bar';
% Create kJkgLabel_2
app.kJkgLabel_2 = uilabel(app.UIFigure);
app.kJkgLabel_2.Position = [588 266 33 15];
app.kJkgLabel_2.Text = 'kJ/kg';
% Create Pressure_OutEditFieldLabel
app.Pressure_OutEditFieldLabel = uilabel(app.UIFigure);
app.Pressure_OutEditFieldLabel.HorizontalAlignment = 'right';
app.Pressure_OutEditFieldLabel.Position = [382 291 82 15];
app.Pressure_OutEditFieldLabel.Text = 'Pressure_Out';
% Create Pressure_OutEditField
app.Pressure_OutEditField = uieditfield(app.UIFigure, 'numeric');
app.Pressure_OutEditField.Limits = [0.0001 600];
app.Pressure_OutEditField.Position = [479 287 100 22];
app.Pressure_OutEditField.Value = 25.48;
% Create Enhalpy_OutEditFieldLabel
app.Enhalpy_OutEditFieldLabel = uilabel(app.UIFigure);
app.Enhalpy_OutEditFieldLabel.HorizontalAlignment = 'right';
app.Enhalpy_OutEditFieldLabel.Position = [389 270 75 15];
app.Enhalpy_OutEditFieldLabel.Text = 'Enhalpy_Out';
% Create Enhalpy_OutEditField
app.Enhalpy_OutEditField = uieditfield(app.UIFigure, 'numeric');
app.Enhalpy_OutEditField.Limits = [200 4000];
app.Enhalpy_OutEditField.Position = [479 266 100 22];
app.Enhalpy_OutEditField.Value = 2658.7;
% Create InputsofthemodelLabel
app.InputsofthemodelLabel = uilabel(app.UIFigure);
app.InputsofthemodelLabel.Position = [366 436 113 15];
app.InputsofthemodelLabel.Text = 'Inputs of the model ';
% Create kgsLabel
```

```
            app.kgsLabel = uilabel(app.UIFigure);
            app.kgsLabel.Position = [588 354 28 15];
            app.kgsLabel.Text = 'kg/s';
            % Create Label
            app.Label = uilabel(app.UIFigure);
            app.Label.Position = [589 333 25 15];
            app.Label.Text = '[-]';
            % Create MassFlowRateEditFieldLabel
            app.MassFlowRateEditFieldLabel = uilabel(app.UIFigure);
            app.MassFlowRateEditFieldLabel.HorizontalAlignment = 'right';
            app.MassFlowRateEditFieldLabel.Position = [377 354 87 15];
            app.MassFlowRateEditFieldLabel.Text = 'MassFlowRate';
            % Create MassFlowRateEditField
            app.MassFlowRateEditField = uieditfield(app.UIFigure, 'numeric');
            app.MassFlowRateEditField.Limits = [0 2000];
            app.MassFlowRateEditField.Position = [479 350 100 22];
            app.MassFlowRateEditField.Value = 1400;
            % Create MechanicalEfficiencyEditFieldLabel
            app.MechanicalEfficiencyEditFieldLabel = uilabel(app.UIFigure);
            app.MechanicalEfficiencyEditFieldLabel.HorizontalAlignment = 'right';
            app.MechanicalEfficiencyEditFieldLabel.Position = [346 333 118 15];
            app.MechanicalEfficiencyEditFieldLabel.Text = 'MechanicalEfficiency';
            % Create MechanicalEfficiencyEditField
            app.MechanicalEfficiencyEditField = uieditfield(app.UIFigure,
'numeric');
            app.MechanicalEfficiencyEditField.Limits = [0 1];
            app.MechanicalEfficiencyEditField.Position = [479 329 100 22];
            app.MechanicalEfficiencyEditField.Value = 0.92;
            % Create ThermodynamicEfficiencyEditFieldLabel
            app.ThermodynamicEfficiencyEditFieldLabel = uilabel(app.UIFigure);
            app.ThermodynamicEfficiencyEditFieldLabel.HorizontalAlignment =
'right';
            app.ThermodynamicEfficiencyEditFieldLabel.Position = [319 312 145 15];
            app.ThermodynamicEfficiencyEditFieldLabel.Text =
'ThermodynamicEfficiency';
            % Create ThermodynamicEfficiencyEditField
            app.ThermodynamicEfficiencyEditField = uieditfield(app.UIFigure,
'numeric');
            app.ThermodynamicEfficiencyEditField.Limits = [0 1];
            app.ThermodynamicEfficiencyEditField.Position = [479 308 100 22];
            app.ThermodynamicEfficiencyEditField.Value = 0.98;
            % Create Label_5
            app.Label_5 = uilabel(app.UIFigure);
            app.Label_5.Position = [589 312 25 15];
            app.Label_5.Text = '[-]';
        end
    end
    methods (Access = public)
        % Construct app
        function app = GUI
            % Create and configure components
            createComponents(app)
            % Register the app with App Designer
            registerApp(app, app.UIFigure)
            if nargout == 0
                clear app
            end
        end
        % Code that executes before app deletion
        function delete(app)
            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```