Appendix A

Manual Computation

DRIVING POTENTIAL

$(\Delta E)\,driving\ potential\ = E_{cathode} - E_{anode}$

$E_{cathode} = -0.80V$

$E_{anode} = -1.10V$

$\Delta E = \ -0.80 - (-1.10)$

$\Delta E = 0.3V$

Software generated value = 0.30 Volts

ANODE RESISTANCE

Anode resistance for Bracelets anodes;

$$R_{anode} = \frac{0.315\rho}{\sqrt{A}}$$

$\rho(density\ of\ the\ media) = 0.30$; $r(radius\ of\ anode) = 0.15$; $L(length\ of\ anode) =$; 0.5;

$A(\text{Anode surface area}) = 2 \times \pi \times R_{anode} \times l_{anode}$

$A(\text{Anode surface area})_{\text{initial}} = 2 \times \pi \times R_{anode} \times l_{anode}$

$A(\text{Anode surface area})_{\text{initial}} = 2 \times \pi \times 0.15 \times 0.5$

$A(\text{Anode surface area})_{\text{initial}} = 0.471\text{m}^2$

$$R_{anode,initial} = \frac{0.315\rho}{\sqrt{A}}$$

$$R_{anode,initial} = \frac{0.315 \times 0.3}{\sqrt{0.471}}$$

$$R_{anode,initial} = 0.137 \text{ Ohms}$$

The $R_{anode,final}$ will be the anode resistance at 85% utilization factor at this stage the anode length and the radius will change hence the resistance, so the length of the anode at this stage will be;(in this formula it is assume that length reduction corresponding to 10% of the net anode mass/volume reduction occurs when the anode has been consumed to its utilization factor) [13]

$$L_{final} = L_{initial} - (0.1 \times u \times L_{initial})$$

$$L_{final} = 0.5 - (0.1 \times 0.85 \times 0.5)$$

$$L_{final} = 0.458\text{m}$$

$L_{initial}(initial\ lenght\ of\ anode)$ =0.5;

$r_{initial}(initial\ radius\ of\ anode) = 0.15$

$L_{final}\ (final\ length\ of\ anode\ at\ 85\%\ utilization\ factor) = 0.458\text{m}$

$r_{core}(anode\ core\ radius) = 0.01\,\text{m}$; $density(density\ of\ anode) = 2750 kg/\text{m}^3$

$u(utilization\ factor) = 0.85$; $m(anode\ unit\ mass) = 50\text{kg}$

$$r_{final} = \sqrt{\frac{m \times (1-u)}{\pi \times density \times L_{final}} + (r_{core})^2}$$

$$r_{final} = \sqrt{\frac{50 \times (1 - 0.85)}{\pi \times 2750 \times 0.46} + (0.01)^2}$$

r_{final} = 0.04354 m

r_{final} *(final radius of anode at 85% utilization factor)* = 0.04354 m

$A(\text{Anode surface area})_{final} = 2 \times \pi \times 0.04354 \times 0.46$

$A(\text{Anode surface area})_{final} = 0.126\ m^2$

$$R_{anode,final} = \frac{0.315 \times 0.3}{\sqrt{0.126}}$$

$R_{anode,final}$ = 0.267 Ohms

ANODE CURRENT OUTPUT

$$I(anode\ current\ output) = \frac{E_{cathode} - E_{anode}}{R_{anode}}$$

$$I_{(anode\ current\ output)_{initial}} = \frac{\Delta E}{R_{anode(initial)}}$$

$$I_{(anode\ current\ output)_{initial}} = \frac{0.3}{0.137}$$

$I_{(anode\ current\ output)_{initial}} = 2.19A$

ΔE = 0.30Volts; $R_{anode,initial} = 0.137\ ohms$; $R_{anode,final} = 0.267\ A$

$$I_{(anode\ current\ output)final} = \frac{0.3}{0.267}$$

$$I_{(anode\ current\ output)_{final}} = 1.123\ A$$

CURRENT DENSITY DEMAND

$$I_{(current\ density\ demand)} = A_{cathode} \times I_{current\ density} \times f$$

$$I_{(current\ density\ demand)initial} = A_{cathode} \times I_{(current\ density)initial} \times f_{initial}$$

$$A_{cathode} = 2 \times \pi \times R_{cathode} \times L_{cathode}$$

$$A_{cathode} = 2 \times \pi \times 0.15 \times 2000$$

$$A_{cathode} = 1884.96\ m^2$$

$$I_{(current\ density\ demand)initial} = 1884.96 \times 0.06 \times 0.05$$

$I_{(current\ density\ demand)initial}$ =5.655 A

$$I_{(current\ density\ demand)final} = A_{cathode} \times I_{(current\ density)final} \times f_{final}$$

$$I_{(current\ density\ demand)final} = 1884.96 \times 0.04 \times 0.07$$

$$I_{(current\ density\ demand)final} = 5.28\ A$$

$$I_{(current\ density\ demand)average} = \frac{I_{(current\ density\ demand)initial} + I_{(current\ density\ demand)final}}{2}$$

$$I_{(current\ density\ demand)average} = \frac{5.655 + 5.28}{2}$$

$I_{(current\ density\ demand)average}$= 5.468 A

TOTAL ANODE MASS

$$M_{total\ anode\ mass} = \frac{I_{cathode,av} \times T \times 8760}{u \times \varepsilon}$$

$$M_{total\ anode\ mass} = \frac{5.468 \times 20 \times 8760}{0.85 \times 1320}$$

$$M_{total\ anode\ mass} = 853.83\ Kg$$

$$N_{anode\ number} = \frac{M_{total\ anode\ mass}}{M_{anode\ mas}}$$

ANODE NUMBER

$$N_{anode\ number} = \frac{853.83}{50}$$

$$N_{anode\ number} = 17.01$$

ANODE SPACING

$$S_{(anode\ spacing)} = \frac{L_{cathode}}{N_{anode\ number}}$$

$$S_{(anode\ spacing)} = \frac{2000}{17.01}$$

$$S_{(anode\ spacing)} = 117.58\ \mathrm{m}$$

POTENTIAL CHECK

The CP design to be effective the potential must be more negative that -0.80

$$E_{cathode} = E_{anode} - \left(R_{anode,final} \times I_{(anode\ current\ output)_{final}}\right)$$

$$E_{cathode} = -1.1 - (0.267 \times 5.28)$$

$E_{cathode} = -2.51\ \text{V}$

CURRENT TEST

$$I_{(anode\ current\ output)_{final}} \times N_{anode\ number} \geq I_{(current\ density\ demand)final}$$

$$I_{(anode\ current\ output)_{final}} \times N_{anode\ number} \geq I_{(current\ density\ demand)final}$$

$$1.123 \times 17 \geq 5.28$$

$$19.091 \geq 5.28$$

ANODE LIFE TEST

Life test at end of 20 year assuming the pipeline is on a constant initial current of 2.19 A

$$\text{life test} = \frac{M \times u \times \varepsilon}{\frac{\Delta E}{R} \times 8760}$$

$$test\ life = \frac{853.83 \times 1320 \times 0.85}{2.19 \times 8760}$$

$$test\ life = 499\ \text{years}$$

ANODE ADJUSTMENT FACTOR

$$anode\ adjustment\ factor = \frac{N_{anode\ number}}{1 + \left(\frac{2L(\ln 0.656 N_{anode\ number})}{S_{(anode\ spacing)} \left(\ln\left(\frac{8L}{d}\right) - 1 \right)} \right)}$$

$L = anode\ length$; $N = number\ of\ anode$; $d = anode\ diameter$; $r = anode\ radius$

$\rho = resistivity\ of\ the\ medium$; $S = anode\ spacing$

$$anode\ adjustment\ factor = \frac{17}{1+\left(\frac{2\times 0.5(\ln 0.656\times 17)}{117.58\left(\ln\left(\frac{8\times 0.5}{0.03}\right)-1\right)}\right)}$$

$$anode\ adjustment\ factor = 16.91$$

Appendix B

How to install the software

Insert the Installation CD, browse the content. Click on the icon that is label setup(x86). Then follow the installation wizard. If your system does not have the

DOTNET framework installed, clicked on the ***dotnetfx*** icon to install. Then click on the setup(x86) icon on the CD folder to installed. The Software will install with an icon on your desktop.

Using the software

The textboxes and the option boxes all have to get a value, but the lead wire resistance on the environment form, if the lead wire resistance textbox, is highlighted - that is made active, then it must be given a value. The values appropriate depend on the variable. The variable without the unit will most likely take Alpha numeric characters – that is words or a combination of numbers and string. When an incorrect value is entered or picked on leaving the textbox or the list box an error dialog box will appear, giving a short explanation of the kind of error it has encountered, the safe thing to do is to go back to the variable in question and make the necessary changes. After completing the form on clicking on the continue button, the software moves you to the next form, and clicking on the back button moves you are move a step backward to the last form, the cancel button clears the all the inputted values on a form allowing you the chance to start afresh. On getting to the calculate form, which is the last of the forms to fill, clicking on the check value prints a quick result for you allowing you the chance to see the general result in a nutshell, and a chance for a quick change if there is need. When you click on the calculate button the final result is printed on the menu form. The accuracy of the value generated by this software depends to a greater extent on the intelligibility of the inputted values, though extensive error handling algorithm has been incorporated within the software, which was developed after considering all possible errors I could “imagine”.

Note: If on clicking the calculate button/check value button the adjusted current value is equal to infinity please click a gain the calculate button or the check value button.

Appendix C

Cpdex source code

```
Option Explicit On
Option Strict On
```

```
Public Module module2


#Region "cpdex core functions"

    Public Sub find_final_dimension()

        inter.final_anode_length = Final1_length(inter.anode_length,
inter.anode_utilization_factor)

        inter.final_anode_radius = Final1_radius(inter.anode_length,
inter.anode_utilization_factor, inter.final_anode_length, inter.anode_mass,
inter.anode_core_radius, inter.anode_density)

        inter.anode_inner_radius = In_anode_radius(inter.anode_outer_radius,
inter.anode_core_radius)

    End Sub

    Public Sub find_anode_area()

        inter.anode_surface_area = Surface1_area(inter.anode_shape, inter.anode_length,
inter.anode_outer_radius)

        inter.final_anode_surface_area = Surface1_area(inter.anode_shape,
inter.final_anode_length, inter.final_anode_radius)

    End Sub

    Public Sub find_structure_area()

        inter.structure_surface_area = Surface1_area(inter.structure_shape,
inter.structure_length, inter.outer_radius)

    End Sub

    Public Sub find_resistance()

        inter.anode_initial_resistance = Resistance(inter.env_resistivity,
inter.anode_length, inter.anode_outer_radius, inter.anode_inner_radius,
inter.anode_surface_area, inter.anode_type)

        inter.anode_final_resistance = Resistance(inter.env_resistivity,
inter.final_anode_length, inter.final_anode_radius, inter.anode_inner_radius,
inter.final_anode_surface_area, inter.anode_type)

        inter.total_resistance = T_resistance(inter.anode_initial_resistance,
inter.electrolyte_resistance, inter.lead_wire_resistance, inter.anode_stand_off)

        inter.anode_spacing = Anode_sp(inter.number_of_anodes, inter.structure_length)

    End Sub

    Public Sub find_current_demand()

        'initial current demand
```

```
        inter.initial_current_demand =
Initial_current_demand(inter.initial_current_density, inter.initial_coating_factor,
inter.structure_surface_area)

        'final current demand

        inter.final_current_demand = Final_current_demand(inter.final_current_density,
inter.final_coating_factor, inter.structure_surface_area)

        'average current demand

        inter.ave_current_demand = Average_current_demand(inter.initial_current_demand,
inter.final_current_demand)

        'totalcpcurrent

        inter.total_CP_current = Total_CP_current(inter.corrosion_potential,
inter.anode_close_circuit_potential, inter.total_resistance, inter.number_of_anodes)

        If inter.cp_type = "Impress Current System" Then

            inter.rectifier_voltage = Rectifier_volt(inter.total_CP_current,
inter.total_resistance, inter.rectifier_adjustment_factor)

        Else

            inter.rectifier_voltage = 0.0

        End If

        inter.anode_adjustment_factor = Adjustment_factor(inter.number_of_anodes,
inter.anode_length, inter.anode_outer_radius, inter.anode_spacing)

        inter.adjusted_anode_current =
Adjusted_an_current(inter.anode_initial_current_output, inter.anode_adjustment_factor)

    End Sub

    Public Sub find_driving_potential()

        inter.driving_potential = Driving_potential(inter.corrosion_potential,
inter.anode_close_circuit_potential)

    End Sub

    Public Sub find_current_capacity()

        inter.anode_initial_current_output =
Anode_current_output(inter.driving_potential, inter.anode_initial_resistance)

        inter.anode_final_current_output = Anode_current_output(inter.driving_potential,
inter.anode_final_resistance)

        inter.current_capacity = C_capacity(inter.anode_mass,
inter.anode_electro_capacity, inter.anode_utilization_factor)

    End Sub

    Public Sub find_anode_mass()

        'anode mass
```

```
        inter.initial_anode_mass = Initial_anode_mass(inter.designlife,
inter.anode_utilization_factor, inter.anode_electro_capacity, inter.anode_surface_area,
inter.ave_current_demand, inter.anode_initial_resistance)

        'final anode mass

        inter.final_anode_mass = Final_anode_mass(inter.initial_anode_mass,
inter.anode_utilization_factor)

    End Sub

    Public Sub find_number_of_anode()

        inter.number_of_anodes = Number_of_anodes(inter.initial_anode_mass,
inter.anode_mass)

    End Sub

    Public Sub totalresistant()

        inter.coating_resistance = Coating_resistance(inter.structure_surface_area,
inter.coating_resistance_sqrm)

        inter.total_resistance = T_resistance(inter.anode_initial_resistance,
inter.coating_resistance, inter.lead_wire_resistance, inter.anode_stand_off)

        inter.slope_parameter = Slope_param(inter.structure_surface_area,
inter.total_resistance)

    End Sub

    Public Sub cost_project()

        inter.project_cost = P_cost(inter.number_of_anodes, inter.anode_cost,
inter.installation_cost)

    End Sub

    Public Sub conversion_sub()

        inter.conversion_output_value =
conversion_rate_calculation(inter.conversion_input_units, inter.conversion_output_units,
inter.conversion_input_value, inter.conversion_electron, inter.conversion_density, _

        inter.conversion_atomic_mass)

    End Sub

    Public Sub corrosion_rate_sub()

        If inter.coupon_calculation_type = "Corrosion Rate Calculation" Then

            inter.coupon_calcution_result =
corrosion_rate_calculation(inter.coupon_initial_weight, inter.coupon_final_weight,
inter.coupon_area, inter.coupon_exposure_time, inter.coupon_metal_density, _

            inter.coupon_area_module, inter.coupon_shape, inter.coupon_length,
inter.coupon_width)

        ElseIf inter.coupon_calculation_type = "Pitting Rate Calculation" Then
```

```
            inter.coupon_calcution_result =
pitting_rate_calculation(inter.coupon_initial_weight, inter.coupon_exposure_time)

        End If

    End Sub

#End Region

#Region "optimization functions"

    Public Sub pass()

        inter.opt_anode_mass = inter.anode_mass

        inter.opt_anode_length = inter.anode_length

        inter.opt_anode_radius = inter.anode_outer_radius

        inter.opt_anode_spacing = inter.anode_spacing

        inter.opt_anode_number = inter.number_of_anodes

        inter.opt_inputed_values = CDbl(0.0)

        inter.opt_current_capacity = inter.current_capacity

        inter.opt_designlife = inter.designlife

        inter.opt_slope_parameter = inter.slope_parameter

        inter.opt_ave_current_demand = inter.ave_current_demand

        inter.opt_structure_length = inter.structure_length

        inter.opt_anode_total_mass = inter.initial_anode_mass

        inter.opt_env_resistivity = inter.env_resistivity

        inter.opt_anode_type = inter.anode_type

        inter.opt_anode_shape = inter.anode_shape

        inter.opt_anode_resistance = CDbl(0.0)

    End Sub

    Public Sub anode_mass_optimise()

        inter.opt_anode_resistance = Get_resistance(inter.opt_current_capacity,
inter.opt_anode_mass, inter.opt_designlife, inter.opt_slope_parameter,
inter.opt_ave_current_demand)

    End Sub

    Public Sub length_optimise()

        inter.opt_anode_length = Get_radius(inter.opt_env_resistivity,
inter.opt_anode_length, inter.opt_anode_type, inter.opt_anode_resistance,
inter.opt_anode_shape)

    End Sub
```

```
    Public Sub radius_optimise()

        inter.opt_anode_radius = Get_length(inter.opt_env_resistivity,
inter.opt_anode_radius, inter.opt_anode_type, inter.opt_anode_mass,
inter.opt_anode_shape)

    End Sub

    Public Sub mass_optimize()

        inter.opt_anode_mass = Get_mass(inter.opt_current_capacity,
inter.opt_designlife, inter.opt_slope_parameter, inter.opt_ave_current_demand,
inter.opt_anode_resistance)

    End Sub

    Public Sub anode_numbers()

        inter.opt_anode_number = Number_of_anodes(inter.opt_anode_total_mass,
inter.opt_anode_mass)

    End Sub

    Public Sub anode_spacing_optimize()

        inter.opt_anode_spacing = Anode_sp(inter.opt_anode_number,
inter.opt_structure_length)

    End Sub

    Public Sub resistance_optimize()

        inter.opt_anode_mass = Get_mass(inter.opt_current_capacity,
inter.opt_designlife, inter.opt_slope_parameter, inter.opt_ave_current_demand,
inter.opt_anode_resistance)

    End Sub

    Public Sub get_num_by_sp()

        inter.opt_anode_number = Get_number_thro_spacing(inter.opt_structure_length,
inter.anode_spacing)

    End Sub

    Public Sub get_mas_by_number()

        inter.opt_anode_mass = Get_mass_thro_number(inter.opt_anode_total_mass,
inter.opt_anode_number)

    End Sub

#End Region

#Region "error check"

    Public Sub values_check()

        'startform

        string_value_main = startform.Text
```

```
        string_value_sub = "Design Life"

        If inter.designlife = 0.0 Then

            MessageBox.Show("A key variable " & startform.lbldesignlife.Text & " is has
a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            startform.Show()

            Exit Sub

            'structurefrom

            sub_status = "design life value OK!"

            string_value_main = structureform.Text

            string_value_sub = "Structure Length"

        ElseIf inter.structure_length = 0.0 Then

            MessageBox.Show("A key variable " & structureform.lblstrlength.Text & " is
has a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            structureform.Show()

            Exit Sub

            sub_status = "Structure Length Ok!"

            string_value_sub = "Structure Shape"

        ElseIf inter.structure_shape = String.Empty Then

            MessageBox.Show("A key variable " & structureform.lblstructure_shape.Text &
" is has a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True




            structureform.Show()

            Exit Sub

            sub_status = "Structure Shape Ok!"

            string_value_sub = "Structure Outer Radius"

        ElseIf inter.outer_radius = 0.0 Then
```

```
            MessageBox.Show("A key variable " & structureform.lblut_radius.Text & " is
has a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            structureform.Show()

            Exit Sub

            sub_status = "Structure Outer Radius Ok!"

            string_value_sub = "Structure Inner Radius"

        ElseIf inter.anode_core_radius = 0.0 Then

            MessageBox.Show("A key variable " & structureform.lblin_radius.Text & " is
has a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True




            structureform.Show()

            Exit Sub

            sub_status = "Structure Inner Radius Ok!"

            string_value_sub = "Structure Protection Potential "

        ElseIf inter.corrosion_potential = 0.0 Then

            MessageBox.Show("A key variable " &
structureform.lblcorrosion_potential.Text & " is has a null value, please input this
value!", "Error Message", MessageBoxButtons.OK, MessageBoxIcon.Stop,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            structureform.Show()

            Exit Sub

            'environment form

            sub_status = "Structure Protection Potential  Ok!"

            string_value_main = envronform.Text

            string_value_sub = "Environment Resistivity"

        ElseIf inter.env_resistivity = 0.0 Then

            MessageBox.Show("A key variable " & envronform.lblenvr_resistivity.Text & "
is has a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
```

```
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            envronform.Show()

            Exit Sub

            sub_status = "Environment Resistivity Ok!"

            string_value_sub = "Environment Initial Current Density"

        ElseIf inter.initial_current_density = 0.0 Then

            MessageBox.Show("A key variable " &
envronform.lblinitial_current_density.Text & " is has a null value, please input this
value!", "Error Message", MessageBoxButtons.OK, MessageBoxIcon.Stop,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            envronform.Show()

            Exit Sub

            sub_status = "Environment Initial Current Density Ok!"

            string_value_sub = "Environment Final Current Density"

        ElseIf inter.final_current_density = 0.0 Then

            MessageBox.Show("A key variable " & envronform.lblfinal_current_density.Text
& " is has a null value, please input this value!", "Error Message",
MessageBoxButtons.OK, MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            envronform.Show()

            Exit Sub

            sub_status = "Environment Final Current Density Ok!"

            string_value_sub = "Environment Average Current Density"

        ElseIf inter.av_current_density = 0.0 Then

            MessageBox.Show("A key variable " & envronform.lblav_current_density.Text &
" is has a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            envronform.Show()

            Exit Sub

            sub_status = "Environment Average Current Density Ok!"
```

```
            'anode frm

            string_value_main = anodeform.Text

            string_value_sub = "Anode close Circuit Potential"

        ElseIf inter.anode_close_circuit_potential = 0.0 Then

            MessageBox.Show("A key variable " &
anodeform.lblclose_circuit_potential.Text & " is has a null value, please input this
value!", "Error Message", MessageBoxButtons.OK, MessageBoxIcon.Stop,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            anodeform.Show()

            Exit Sub

            sub_status = "Anode Potential OK!"

            string_value_sub = "Anode Outer radius"

        ElseIf inter.anode_outer_radius = 0.0 Then

            MessageBox.Show("A key variable " & anodeform.lbluouter_radius.Text & " is
has a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            anodeform.Show()

            Exit Sub

            sub_status = "Anode Outer radius Ok!"

            string_value_sub = "Anode Core Radius"

        ElseIf inter.anode_core_radius < 0.0 Then

            MessageBox.Show("A key variable " & anodeform.lblucore_radius.Text & " is
has a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            anodeform.Show()

            Exit Sub

            sub_status = "Anode Core Radius Ok!"

            string_value_sub = "Anode Length"

        ElseIf inter.anode_length = 0.0 Then

            MessageBox.Show("A key variable " & anodeform.lblulength.Text & " is has a
null value, please input this value!", "Error Message", MessageBoxButtons.OK,
```

```
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            anodeform.Show()

            Exit Sub

            sub_status = "Anode  Length Ok!"

            string_value_sub = "Anode Shape "

        ElseIf inter.anode_shape = String.Empty Then

            MessageBox.Show("A key variable " & anodeform.lblanode_shape.Text & " is has
a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            anodeform.Show()

            Exit Sub

            sub_status = "Anode  Shape Ok!"

            string_value_sub = "Anode Type "

        ElseIf inter.anode_type = String.Empty Then

            MessageBox.Show("A key variable " & anodeform.lblanode_type.Text & " is has
a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            anodeform.Show()

            Exit Sub

            sub_status = "Anode Type Ok!"

            string_value_sub = "Anode Current Capacity "

        ElseIf inter.anode_electro_capacity = 0.0 Then

            MessageBox.Show("A key variable " & anodeform.lblcurrent_capacity.Text & "
is has a null value, please input this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            tell_stories = True

            anodeform.Show()

            Exit Sub

            sub_status = "Anode Current Capacity Ok!"
```

```
            tell_stories = False

        End If

    End Sub

    Public Sub super_check()

        inter.check_potential = potential_test(inter.corrosion_potential,
inter.anode_close_circuit_potential, inter.anode_initial_resistance,
inter.initial_current_demand)

        inter.check_av_current = av_curr_test(inter.current_capacity,
inter.number_of_anodes, inter.ave_current_demand, inter.designlife)

        inter.check_initial_current = ini_curr_test(inter.number_of_anodes,
inter.initial_current_demand, inter.anode_initial_current_output)

        inter.check_final_current = final_curr_test(inter.number_of_anodes,
inter.final_current_demand, inter.anode_final_current_output)

        If inter.check_potential = False Then

            MessageBox.Show("The potential is not adequate for the protection of the
structure consider revising", "Error Meassage", MessageBoxButtons.OK,
MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

        End If

        If inter.check_av_current = False Then

            MessageBox.Show("The number of anodes will not provide adequate protection
given the design life of " & inter.designlife & "years consider revising the relevant
parameters", "Error Message", MessageBoxButtons.OK, MessageBoxIcon.Warning,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly)

        End If

        If inter.check_initial_current = False Then

            MessageBox.Show("The number of anodes will not provide adequate protection
given the design life of " & inter.designlife & "years consider revising the relevant
parameters", "Error Message", MessageBoxButtons.OK, MessageBoxIcon.Warning,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly)

        End If

        If inter.check_final_current Then

            MessageBox.Show("The number of anodes will not provide adequate protection
given the design life of " & inter.designlife & "years consider revising the relevant
parameters", "Error Message", MessageBoxButtons.OK, MessageBoxIcon.Warning,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly)

        End If

        If inter.anode_spacing >= 200 Then

            MessageBox.Show("The anodes spacing is greater than 200 metres for the
design life of " & inter.designlife & "years, consider attenuation check", "Error
```

```
Message", MessageBoxButtons.OK, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

        End If

        MessageBox.Show("Computatonal error Check succesfull!", "Error Message",
MessageBoxButtons.OK, MessageBoxIcon.None, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

    End Sub

#End Region

#Region "utility functions"

    Public Sub project_name()

        If inter.projectid = "" Then

            displayform.Text = "New project"

        Else

            displayform.Text = inter.projectid

        End If

    End Sub

    Public Sub status_sub()

        If status = True Then

            main_menufrm.StatusStrip.Text = "Ready"

        End If

    End Sub

    Public Sub calculate()

        find_final_dimension()

        find_anode_area()

        find_structure_area()

        find_current_demand()

        find_driving_potential()

        find_current_capacity()

        find_anode_mass()

        find_number_of_anode()

        find_resistance()

        totalresistant()

        cost_project()
```

```
    End Sub

    Public Sub display()

        displayform.rtbdisplay.Text = (vbTab & "This document is generated by CPDEx
(Academic Version)" & vbNewLine & vbTab & vbNewLine & "
" & _

        "Date :" & Date.Now.ToLongDateString & vbNewLine & vbNewLine & _

        vbTab & "CLIENT :" & vbTab & "                                                  " &
_

        inter.client.ToUpper & vbNewLine & _

        vbTab & "PROJECT :" & vbTab & "                                " & _

        inter.projectid.ToString.ToUpper & vbNewLine & _

        vbTab & "VERSION :" & vbTab & "                                " & _

        inter.projectversion.ToString.ToUpper & vbNewLine & vbNewLine & vbTab &
"**************************************************************************************" &
vbNewLine & vbTab & _

         "DESIGN PARAMETERS ARE AS FOLLOWS:" & vbNewLine & vbTab &
"**************************************************************************************" & _

         vbNewLine & vbNewLine & vbTab & "PROJECT TYPE : " & vbTab & "
" & _

         inter.projectype.ToString & vbNewLine & _

         vbTab & "CONFORMITY:" & vbTab & "                                 " & _

         inter.conformity.ToString & vbNewLine & _

         vbTab & "DESIGN LIFE:" & vbTab & "                                  " & _

         inter.designlife.ToString & "  " & calculateform.lbldesignlife_unts.Text &
vbNewLine & _

         vbTab & "CP TYPE:" & vbTab & "                                " & _

         inter.cp_type.ToString & vbNewLine & _

         vbTab & "RECTIFIER MODIFICATION FACTOR:" & vbTab & "      " & _

         inter.rectifier_adjustment_factor.ToString("f3") & vbNewLine & _

         vbTab & "REFERENCE CELL :" & vbTab & "              " & _

         inter.reference_cell.ToString & vbNewLine & _

         vbTab & "DRIVING POTENTIAL:" & vbTab & "             " & _

         inter.driving_potential.ToString("f3") & "  " &
calculateform.lbldriving_potential_unts.Text & vbNewLine & _

         vbTab & "INITIAL CURRENT DEMAND:" & vbTab & _
```

```
         inter.initial_current_demand.ToString("f3") & "  " &
calculateform.lblinitial_curr_demnd_unts.Text & vbNewLine & _

         vbTab & "AVERAGE CURRENT DEMAND:" & vbTab & _

         inter.ave_current_demand.ToString("f3") & "  " &
calculateform.lblinitial_curr_demnd_unts.Text & vbNewLine & _

         vbTab & " FINAL CURRENT DEMAND :" & vbTab & _

         inter.final_current_demand.ToString("f3") & "  " &
calculateform.lblinitial_curr_demnd_unts.Text & _

         vbNewLine & vbTab & "STRUCTURE SURFACE AREA :" & vbTab & _

         inter.structure_surface_area.ToString("f3") & "  " &
calculateform.lblstructure_area_unts.Text & vbNewLine & _

         vbTab & "ANODE SURFACE AREA :" & vbTab & "                " & _

         inter.anode_surface_area.ToString("f3") & "  " &
calculateform.lblanode_surface_area_unts.Text & vbNewLine & _

         vbTab & "ANODE SPACING :" & vbTab & "                                  " & _

        inter.anode_spacing.ToString("f3") & "  " &
calculateform.lblanode_spacing_unts.Text & vbNewLine & _

        vbTab & " INITIAL ANODE CURRENT OUTPUT :" & vbTab & _

        inter.anode_initial_current_output.ToString("f3") & "  " &
calculateform.lblanode_adjusted_current_unts.Text & _

        vbNewLine & vbTab & "FINAL ANODE CURRENT OUTPUT:" & vbTab & _

        inter.anode_final_current_output.ToString("f3") & "  " &
calculateform.lblanode_adjusted_current_unts.Text & _

        vbNewLine & vbTab & "ANODE RESISTANCE: " & vbTab & "                " & _

        inter.anode_initial_resistance.ToString("f3") & "  " &
calculateform.lblanode_resistance_unts.Text & _

        vbNewLine & vbTab & "NUMBER OF ANODES :" & vbTab & "                " & _

        CInt(inter.number_of_anodes).ToString & _

        vbNewLine & vbTab & "TOTAL ANODE WEIGHT :" & vbTab & "                 " & _

        inter.initial_anode_mass.ToString("f3") & "  " &
calculateform.lbltotal_anode_weight_unts.Text & _

        vbNewLine & vbTab & "ANODE MASS : " & vbTab & "                                  "
& _

        inter.anode_mass.ToString("f3") & "  " &
calculateform.lbltotal_anode_weight_unts.Text & vbNewLine & _

        vbTab & "PROJECT COST :" & vbTab & "                                " & _

        inter.project_cost.ToString("f3") & "  " &
calculateform.lblproject_cost_unts.Text & _
```

```
        vbNewLine & vbTab & "RECTIFIER VOLTAGE :" & vbTab & "                  " & _
        inter.rectifier_voltage.ToString("f3") & "  " &
calculateform.lblrectifier_voltage_unts.Text & _
        vbNewLine)
    End Sub
    Public Sub display_resize()
        displayfrm.Size = menu_form_size
        displayfrm.rtbdisplay.Size = menu_form_size
    End Sub
    Public Function cp_type_check(ByVal cp_type As String) As Boolean
        If cp_type = "Galvanic Anode System" Then
            Return (True)
        End If
    End Function
    Public Sub set_to_empty()
        startform.set_2_null()
        calculateform.set_2_null()
        envronform.set_2_null()
        anodeform.set_2_null()
        optfrm.set_2_null()
        structureform.set_2_null()
        displayform.Hide()
        set_list4anodefrm()
    End Sub
    Public Sub set_list4anodefrm()
        If anodeform.lstanodeshape.SelectedValue Is String.Empty Then
            anodeform.txtouter_radius.Enabled = False
        End If
    End Sub
#End Region
End Module
Public Module module1
```

```
#Region "declaration of core cpdex functions"

    Public err As New errorcls

    Public inter As New wrappercls

    Public resistant As New anode_resistance

    Public area As New surface_area

    Public main_current As New current

    Public main_resistance As New total_resistance

    Public project_cost As New cost

    Public opt As New optimization

    Public t_resistant As New total_resistance

    Public corr_rate_calc As New corrosion_rate

    Public conversion_calc As New conversion

    Public rtbdisplay As New RichTextBox

#End Region

#Region "delegate for cpdx core functions"

    Delegate Function d_inner_radius(ByVal anode_outer_radius As Double, ByVal
anode_core_radius As Double) As Double

    Delegate Function d_area(ByVal shape As String, ByVal length As Double, ByVal radius
As Double) As Double

    Delegate Function d_resistance(ByVal resistivity As Double, ByVal length As Double,
ByVal outer_radius As Double, ByVal inner_radius As Double, ByVal surface_area As
Double, ByVal anodechoice As String) As Double

    Delegate Function d_final_radius(ByVal length As Double, ByVal utfactor As Double,
ByVal length_final As Double, ByVal mass As Double, ByVal radius_core As Double, ByVal
density As Double) As Double

    Delegate Function d_final_length(ByVal length As Double, ByVal utfactor As Double)
As Double

    Delegate Function d_initial_current_demand(ByVal structure_current_density As
Double, ByVal incbdfactor As Double, ByVal area As Double) As Double

    Delegate Function d_final_current_demand(ByVal current_density As Double, ByVal
fncbdfactor As Double, ByVal area As Double) As Double

    Delegate Function d_total_cp_current(ByVal structure_potential As Double, ByVal
anode_potential As Double, ByVal total_resistance As Double, ByVal anode_number As
Double) As Double

    Delegate Function d_average_current_demand(ByVal incurrent_demand As Double, ByVal
fncurrent_demand As Double) As Double
```

```
    Delegate Function d_initial_anode_mass(ByVal design_life As Double, ByVal utfactor
As Double, ByVal cur_capacity As Double, ByVal carea As Double, ByVal avcurrent_demand
As Double, ByVal anode_resistance As Double) As Double

    Delegate Function d_final_anode_mass(ByVal mass As Double, ByVal utfactor As Double)
As Double

    Delegate Function d_number_of_anode(ByVal total_Anode_weight As Double, ByVal
mass_per_anode As Double) As Double

    Delegate Function d_driving_potential(ByVal structure_potential As Double, ByVal
anode_potential As Double) As Double

    Delegate Function d_project_cost(ByVal anodenumber As Double, ByVal cost_per_anode
As Double, ByVal installation_cost_per_anode As Double) As Double

    Delegate Function d_current_capacity(ByVal mass_per_anode As Double, ByVal
anode_electro_capacity As Double, ByVal utilzation_fctr As Double) As Double

    Delegate Function d_anode_current_output(ByVal driving_potential As Double, ByVal
anode_resistance As Double) As Double

    Delegate Function d_rectifier_voltage(ByVal total_protection_current As Double,
ByVal total_resistance As Double, ByVal rectifier_adjustment_factor As Double) As Double

    Delegate Function d_anode_adjustment_factor(ByVal anode_number As Double, ByVal
anode_length As Double, ByVal anode_outer_radius As Double, ByVal anode_spacing As
Double) As Double

    Delegate Function d_anode_spacing(ByVal anode_number As Double, ByVal
structure_length As Double) As Double

    Delegate Function d_anode_adjusted_current(ByVal anode_initial_current_output As
Double, ByVal adjustment_factor As Double) As Double

    Delegate Function d_opt_g_radius(ByVal resistivity As Double, ByVal anode_length As
Double, ByVal anodechoice As String, ByVal anode_resistance As Double, ByVal anode_shape
As String) As Double

    Delegate Function d_opt_g_length(ByVal resistivity As Double, ByVal anode_radius As
Double, ByVal anodechoice As String, ByVal anode_resistance As Double, ByVal anode_shape
As String) As Double

    Delegate Function d_opt_g_resistance(ByVal anode_current_capacity As Double, ByVal
anode_mass As Double, ByVal design_life As Double, ByVal slope_parameter As Double,
ByVal average_current_demand As Double) As Double

    Delegate Function d_opt_g_anode_mass(ByVal anode_current_capacity As Double, ByVal
design_life As Double, ByVal slope_parameter As Double, ByVal average_current_demand As
Double, ByVal anode_resistance As Double) As Double

    Delegate Function d_opt_g_anode_number(ByVal anode_total_weight As Double, ByVal
anode_mass As Double) As Double

    Delegate Function d_slope_parameter(ByVal total_surface_area As Double, ByVal
total_resistance As Double) As Double

    Delegate Function d_coating_resistance(ByVal structure_surface_area As Double, ByVal
coating_resistance_area As Double) As Double
```

```
    Delegate Function d_total_resistance(ByVal anode_resistance As Double, ByVal
coating_resistance As Double, ByVal leadwires_resistance As Double, ByVal stand_off As
Double) As Double

    Delegate Function d_opt_g_mass_by_number(ByVal total_anode_mass As Double, ByVal
anode_number As Double) As Double

    Delegate Function d_opt_g_number_by_spacing(ByVal structure_length As Double, ByVal
spacing As Double) As Double

    Delegate Function d_potential_check(ByVal structure_potential As Double, ByVal
anode_potential As Double, ByVal total_resistance As Double, ByVal current_demand As
Double) As Boolean

    Delegate Function d_av_current_check(ByVal cur_capacity As Double, ByVal
anode_number As Double, ByVal av_current_demand As Double, ByVal designyears As Double)
As Boolean

    Delegate Function d_ini_current_check(ByVal anode_number As Double, ByVal
initial_cur_demand As Double, ByVal initial_anode_current As Double) As Boolean

    Delegate Function d_final_current_check(ByVal anode_number As Double, ByVal
final_cur_demand As Double, ByVal final_anode_current As Double) As Boolean

    Delegate Function d_thickness_error(ByVal qw As Control, ByVal outer_value As
Double, ByVal inner_value As Double) As Boolean

    Delegate Function d_corr_calc(ByVal initial_weight As Double, ByVal final_weight As
Double, ByVal coupon_area As Double, ByVal exposure_time As Double, ByVal coupon_density
As Double, ByVal area_module As Boolean, ByVal coupon_shape As String, _

    ByVal coupon_length As Double, ByVal coupon_width As Double) As Double

    Delegate Function d_conversion_calc(ByVal input_units As String, ByVal output_units
As String, ByVal input_value As Double, ByVal conversion_electron As Integer, ByVal
conversion_density As Double, _

    ByVal conversion_atomic_mass As Double) As Double

    Delegate Function d_pitting_rate(ByVal pit_depth As Double, ByVal exposure_time As
Double) As Double

#End Region

#Region "delegate for error class"

    Delegate Function ch_setting() As DialogResult

    Delegate Function impro(ByVal et As Control) As DialogResult

    Delegate Function empt_t(ByVal et As Control) As DialogResult

    Delegate Function empt_l(ByVal et As Control) As DialogResult

    Delegate Function causion(ByVal et As Control) As DialogResult

    Delegate Function t2_large(ByVal et As Control, ByVal limit_value As String) As
DialogResult

    Delegate Function module_absent() As DialogResult

    Delegate Function oversite(ByVal et As Control) As DialogResult
```

```
    Delegate Function not_developed(ByVal et As Control) As DialogResult
#End Region
#Region "forms declaration"
    Public copyrightform As New copyrightfrm
    Public welcomeform As New welcomefrm
    Public startform As New startfrm
    Public structureform As New structurefrm
    Public envronform As New envronfrm
    Public anodeform As New anodefrm
    Public calculateform As New calculatefrm
    Public displayform As New displayfrm
    Public menu_form_size As System.Drawing.Size
    Public string_value_main As String
    Public string_value_sub As String
    Public sub_status As String
    Public tell_stories As Boolean
    Public conversionform As New conversionfrm
    Public rate_calculationform As New c_ratefrm
    Public status As Boolean
#End Region
#Region "implementation of delegates for error class"
    Public error_test As New errorcheck
    Public errt As ch_setting = AddressOf err.about_2_changesetting
    Public emp_l As empt_l = AddressOf err.l_empty
    Public emp_t As empt_t = AddressOf err.t_empty
    Public absent As module_absent = AddressOf err.moduleabsent
    Public imprp As impro = AddressOf err.improperval
    Public caus As causion = AddressOf err.caution
    Public two_large As t2_large = AddressOf err.toolarge
    Public not_develop As not_developed = AddressOf err.module_not_developed
    Public ovsight As oversite = AddressOf err.oversight
```

```
#End Region

#Region "implementation of core cpdex functions"

    Public In_anode_radius As d_inner_radius = AddressOf resistant.anode_inner_radius

    Public Resistance As d_resistance = AddressOf resistant.anode_resistance

    Public Surface1_area As d_area = AddressOf area.surface_area

    Public Final1_length As d_final_length = AddressOf resistant.anode_length_final

    Public Final1_radius As d_final_radius = AddressOf resistant.anode_radius_final

    Public Number_of_anodes As d_number_of_anode = AddressOf resistant.number_anodes

    Public Initial_current_demand As d_initial_current_demand = AddressOf
main_current.initial_current_demand

    Public Final_current_demand As d_final_current_demand = AddressOf
main_current.final_current_demand

    Public Total_CP_current As d_total_cp_current = AddressOf
main_current.total_cpcurrent

    Public Initial_anode_mass As d_initial_anode_mass = AddressOf
resistant.total_anode_mass

    Public Average_current_demand As d_average_current_demand = AddressOf
main_current.av_current_demand

    Public Final_anode_mass As d_final_anode_mass = AddressOf resistant.mass_final

    Public Driving_potential As d_driving_potential = AddressOf
main_current.Driving_potential

    Public P_cost As d_project_cost = AddressOf project_cost.total_cost

    Public C_capacity As d_current_capacity = AddressOf main_current.current_capacity

    Public Anode_current_output As d_anode_current_output = AddressOf
main_current.anode_current_output

    Public Rectifier_volt As d_rectifier_voltage = AddressOf
main_current.rectifier_voltage

    Public Adjustment_factor As d_anode_adjustment_factor = AddressOf
main_current.anode_adjustment_factor

    Public Anode_sp As d_anode_spacing = AddressOf resistant.anode_spacing

    Public Adjusted_an_current As d_anode_adjusted_current = AddressOf
main_current.anode_adjusted_current

    Public Get_radius As d_opt_g_radius = AddressOf opt.get_radius

    Public Get_length As d_opt_g_length = AddressOf opt.get_length

    Public Get_anode_number As d_opt_g_anode_number = AddressOf opt.get_anode_number

    Public Get_resistance As d_opt_g_resistance = AddressOf opt.get_anode_resistance
```

```
    Public Get_mass As d_opt_g_anode_mass = AddressOf opt.get_anode_weight

    Public Get_mass_thro_number As d_opt_g_mass_by_number = AddressOf
opt.get_mass_bs_on_number

    Public Get_number_thro_spacing As d_opt_g_number_by_spacing = AddressOf
opt.get_number_bs_on_spacing

    Public Slope_param As d_slope_parameter = AddressOf t_resistant.slope_parameter

    Public Coating_resistance As d_coating_resistance = AddressOf
t_resistant.coating_resistance

    Public T_resistance As d_total_resistance = AddressOf
main_resistance.total_resistance

    Public potential_test As d_potential_check = AddressOf error_test.potential_check

    Public av_curr_test As d_av_current_check = AddressOf
error_test.current_check_average

    Public ini_curr_test As d_ini_current_check = AddressOf
error_test.current_check_initial

    Public final_curr_test As d_final_current_check = AddressOf
error_test.current_check_final

    Public corrosion_rate_calculation As d_corr_calc = AddressOf
corr_rate_calc.corrosion_rate_calc

    Public conversion_rate_calculation As d_conversion_calc = AddressOf
conversion_calc.corrosion_conversion

    Public pitting_rate_calculation As d_pitting_rate = AddressOf
corr_rate_calc.pitting_rate

#End Region

End Module
```

Corecpdx class

```
Option Explicit On

Option Strict On

Imports System.Math

Imports System.Windows.Forms

Public Class surface_area

    Public Function surface_area(ByVal shape As String, ByVal length As Double, ByVal
radius As Double) As Double

        'Try

        If shape = "Cylindrical" Then

            Return (2 * PI * radius * length)
```

```
        ElseIf shape = "Cuboid" Then

            Return (6 * (length * radius))

        ElseIf shape = "Cubic" Then

            Return (6 * length * radius)

        End If

    End Function

    Public Function structure_thickness(ByVal outer_radius As Double, ByVal inner_radius
As Double) As Double

        Return (outer_radius - inner_radius)

    End Function

    Public Function check_thickness_error(ByVal qw As Control, ByVal outer_value As
Double, ByVal inner_value As Double) As Boolean

        If outer_value <= inner_value Then

            MessageBox.Show("The value in " & qw.Text & " implies a zero or negative
thickness, adjust this value!", "Error Message", MessageBoxButtons.OK,
MessageBoxIcon.Error, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

            Return (True)

        End If

    End Function

End Class

Public Class anode_resistance

    Public Sub New()

    End Sub

    Public Function anode_inner_radius(ByVal anode_outer_radius As Double, ByVal
anode_core_radius As Double) As Double

        Return ((anode_outer_radius) - (anode_core_radius))

    End Function

    Public Function anode_length_final(ByVal length As Double, ByVal utfactor As Double)
As Double

        Return (length - (0.1 * utfactor * length))

    End Function

    Public Function anode_radius_final(ByVal length As Double, ByVal utfactor As Double,
ByVal length_final As Double, ByVal mass As Double, ByVal radius_core As Double, ByVal
density As Double) As Double
```

```
        Return (Sqrt((mass * (1 - utfactor) / (PI * density * length_final)) +
(radius_core) ^ 2))

    End Function

    Public Function mass_final(ByVal mass As Double, ByVal utfactor As Double) As Double

        Return (mass * (1 - utfactor))

    End Function

    Public Function anode_resistance(ByVal resistivity As Double, ByVal length As
Double, ByVal outer_radius As Double, ByVal inner_radius As Double, ByVal surface_area
As Double, ByVal anodechoice As String) As Double

        Dim length_test As Double

        Dim thickness As Double

        Dim thickness_test As Double

        Dim width#

        Dim width_test#

        length_test = 4 * outer_radius

        thickness = ((outer_radius) - (inner_radius))

        width = (2 * ((outer_radius) * (PI)))

        thickness_test = ((4 * thickness))

        width_test = (4 * width)

        If (anodechoice = "Long slender stand-off") AndAlso ((length >= length_test))
Then

            Return (((resistivity / (2 * length * PI) * (Log(4 * length /
(outer_radius)) - 1))))

        ElseIf (anodechoice = "Short slender stand-off") AndAlso (length < length_test)
Then

            Return (((resistivity / (2 * PI * length)) * (Log((2 * length /
outer_radius) * (1 + (Sqrt(1 + (outer_radius / 2 * length) ^ 2))) + (outer_radius / (2 *
length)) - (Sqrt(1 + (outer_radius / 2 * length) ^ 2))))))

        ElseIf (anodechoice = "Long flush mounted") AndAlso (length > thickness_test)
AndAlso (length > width_test) Then

            Return (((resistivity) / (2 * ((length + width) / 2))))

        Else

            Return (((0.315 * resistivity) / (Sqrt(surface_area))))

        End If

    End Function
```

```
    Public Function total_anode_mass(ByVal design_life As Double, ByVal utfactor As
Double, ByVal cur_capacity As Double, ByVal carea As Double, ByVal avcurrent_demand As
Double, ByVal anode_resistance As Double) As Double

        Return (((avcurrent_demand) * (design_life) * 8760) / ((cur_capacity) *
(utfactor)))

    End Function

    Public Function number_anodes(ByVal total_Anode_weight As Double, ByVal
mass_per_anode As Double) As Double

        Return (total_Anode_weight / mass_per_anode)

    End Function

    'Public Function number_anode_thro_weight(ByVal weight_per_anode As Double, ByVal
anode_total_weight As Double) As Double

    '    Return (((anode_total_weight) / (weight_per_anode)))

    'End Function

    Public Function anodetest(ByVal anodenumber As Double, ByVal anode_mass As Double,
ByVal cur_capacity As Double) As Double

        Return (anodenumber * anode_mass)

    End Function

    Public Function anode_spacing(ByVal anode_number As Double, ByVal structure_length
As Double) As Double

        Return (structure_length / anode_number)

    End Function

End Class

Public Class total_resistance

    Public Function coating_resistance(ByVal structure_surface_area As Double, ByVal
coating_resistance_area As Double) As Double

        Return (coating_resistance_area / structure_surface_area)

    End Function

    Public Function total_resistance(ByVal anode_resistance As Double, ByVal
coating_resistance As Double, ByVal leadwires_resistance As Double, ByVal stand_off As
Double) As Double

        If stand_off >= 0.15 AndAlso stand_off < 0.3 Then

            Return ((anode_resistance * 1.3) + coating_resistance +
leadwires_resistance)

        Else

            Return (anode_resistance + coating_resistance + leadwires_resistance)

        End If
```

```
    End Function

    Public Function structure_resistance(ByVal structure_surface_area As Double, ByVal
structure_resistivity As Double) As Double

        Return (structure_resistivity / structure_surface_area)

    End Function

    Public Function slope_parameter(ByVal total_surface_area As Double, ByVal
total_resistance As Double) As Double

        Return ((total_surface_area * total_resistance))

    End Function

End Class

Public Class current

    Public Function current_capacity(ByVal mass_per_anode As Double, ByVal
anode_electro_capacity As Double, ByVal utilzation_fctr As Double) As Double

        Return ((mass_per_anode * anode_electro_capacity * utilzation_fctr))

    End Function

    Public Function initial_current_demand(ByVal structure_current_density As Double,
ByVal incbdfactor As Double, ByVal area As Double) As Double

        Return (structure_current_density * area * incbdfactor)

    End Function

    Public Function final_current_demand(ByVal current_density As Double, ByVal
fncbdfactor As Double, ByVal area As Double) As Double

        Return (current_density * area * fncbdfactor)

    End Function

    Public Function av_current_demand(ByVal incurrent_demand As Double, ByVal
fncurrent_demand As Double) As Double

        Return (((incurrent_demand + fncurrent_demand) * 0.5))

    End Function

    Public Function total_cpcurrent(ByVal structure_potential As Double, ByVal
anode_potential As Double, ByVal total_resistance As Double, ByVal anode_number As
Double) As Double

        Return (((anode_number) * ((structure_potential) - (anode_potential))) /
(total_resistance))

    End Function

    Public Function Driving_potential(ByVal structure_potential As Double, ByVal
anode_potential As Double) As Double

        Return (structure_potential - anode_potential)
```

```
    End Function

    Public Function anode_current_output(ByVal driving_potential As Double, ByVal
anode_resistance As Double) As Double

        Return (driving_potential / anode_resistance)

    End Function

    Public Function anode_adjusted_current(ByVal anode_initial_current_output As Double,
ByVal adjustment_factor As Double) As Double

        Return (anode_initial_current_output * adjustment_factor)

    End Function

    Public Function anode_adjustment_factor(ByVal anode_number As Double, ByVal
anode_length As Double, ByVal anode_outer_radius As Double, ByVal anode_spacing As
Double) As Double

        Return ((anode_number / (1 + ((2 * anode_length * (Log(0.656 * anode_number))) /
(anode_spacing * (Log((16 * anode_length) / anode_outer_radius) - 1))))))

    End Function

    Public Function rectifier_voltage(ByVal total_protection_current As Double, ByVal
total_resistance As Double, ByVal rectifier_adjustment_factor As Double) As Double

        Return ((total_protection_current * total_resistance *
rectifier_adjustment_factor))

    End Function

End Class

Public Class errorcheck

    Sub New()

    End Sub

    Public Function total_current_capacity(ByVal anodenumber As Double, ByVal
cur_capacity As Double) As Double

        Return (anodenumber * cur_capacity)

    End Function

    Public Function total_current(ByVal average_current_demand As Double, ByVal
design_life As Double) As Double

        Return (average_current_demand * design_life * 87690)

    End Function

    Public Function initial_current_output(ByVal anodenumber As Double, ByVal
anode_initial_current_output As Double) As Double

        Return (anodenumber * anode_initial_current_output)

    End Function
```

```
    Public Function final_current_output(ByVal anodenumber As Double, ByVal
anode_final_current_output As Double) As Double

        Return (anodenumber * anode_final_current_output)

    End Function

    Public Function protection_potential(ByVal anode_potential As Double, ByVal
total_resistance As Double, ByVal current_demand As Double) As Double

        Return ((anode_potential) - (total_resistance * current_demand))

    End Function

    Public Function potential_check(ByVal structure_potential As Double, ByVal
anode_potential As Double, ByVal total_resistance As Double, ByVal current_demand As
Double) As Boolean

        Dim contrl As Double

        contrl = ((anode_potential) - (total_resistance * current_demand))

        If structure_potential >= contrl Then

            Return (True)

        Else

            Return (False)

        End If

    End Function

    Public Function current_check_average(ByVal cur_capacity As Double, ByVal
anode_number As Double, ByVal av_current_demand As Double, ByVal designyears As Double)
As Boolean

        Dim contrl As Double

        Dim contrl2 As Double

        contrl = ((av_current_demand * designyears * 8760))

        contrl2 = (anode_number * cur_capacity)

        If contrl2 >= contrl Then

            Return (True)

        Else

            Return (False)

        End If

    End Function

    Public Function current_check_initial(ByVal anode_number As Double, ByVal
initial_cur_demand As Double, ByVal initial_anode_current As Double) As Boolean

        Dim contrl As Double
```

```
        contrl = (anode_number * initial_anode_current)

        If contrl >= initial_cur_demand Then

            Return (True)

        Else

            Return (False)

        End If

    End Function

    Public Function current_check_final(ByVal anode_number As Double, ByVal
final_cur_demand As Double, ByVal final_anode_current As Double) As Boolean

        Dim contrl As Double

        contrl = (anode_number * final_anode_current)

        If contrl = final_cur_demand Then

            Return (True)

        Else

            Return (False)

        End If

    End Function

End Class

Public Class optimization

    Public Function get_radius(ByVal resistivity As Double, ByVal anode_length As
Double, ByVal anodechoice As String, ByVal anode_resistance As Double, ByVal anode_shape
As String) As Double

        Dim area As Double

        If anodechoice = "Long slender stand-off" Then

            Return (((4 * anode_length) / (10 ^ ((2 * PI * anode_length *
anode_resistance) / resistivity) + 1)))

        ElseIf anodechoice = "Bracelets" Then

            Return ((resistivity - (anode_resistance * anode_length)) /
(anode_resistance))

        ElseIf anodechoice = "Long flush mounted" Then

            area = (((0.315 * resistivity) / (anode_resistance)) ^ 2)

            If anode_shape = " Cylindrical" Then

                Return ((area) / (2 * PI * anode_length))

            ElseIf anode_shape = "Cuboid" Then
```

```
                Return ((area) / (6 * anode_length))

            End If

        End If


    End Function

    Public Function get_length(ByVal resistivity As Double, ByVal anode_radius As
Double, ByVal anodechoice As String, ByVal anode_resistance As Double, ByVal anode_shape
As String) As Double

        Dim area As Double

        If anodechoice = "Long flush mounted" Then

            Return ((resistivity - (anode_resistance * anode_radius)) /
(anode_resistance))

        ElseIf anodechoice = "Bracelets " Then

            area = (((0.315 * resistivity) / (anode_resistance)) ^ 2)

            If anode_shape = "Cylindrical" Then

                Return ((area) / (2 * PI * anode_radius))

            ElseIf anode_shape = "Cuboid" Then

                Return ((area) / (6 * anode_radius))

            End If

        End If

    End Function

    Public Function get_anode_number(ByVal anode_total_weight As Double, ByVal
anode_mass As Double) As Double

        Return ((anode_total_weight / anode_mass))

    End Function

    Public Function get_anode_weight(ByVal anode_current_capacity As Double, ByVal
design_life As Double, ByVal slope_parameter As Double, ByVal average_current_demand As
Double, ByVal anode_resistance As Double) As Double

        Return ((((anode_current_capacity ^ -1) * design_life * slope_parameter *
average_current_demand) / (anode_resistance)))

    End Function

    Public Function get_anode_resistance(ByVal anode_current_capacity As Double, ByVal
anode_mass As Double, ByVal design_life As Double, _

ByVal slope_parameter As Double, ByVal average_current_demand As Double) As Double

        Return ((((anode_current_capacity ^ -1) * design_life * slope_parameter *
average_current_demand) / (anode_mass)))
```

```
    End Function

    Public Function get_number_bs_on_spacing(ByVal structure_length As Double, ByVal
spacing As Double) As Double

        Return ((structure_length / spacing))

    End Function

    Public Function get_mass_bs_on_number(ByVal total_anode_mass As Double, ByVal
anode_number As Double) As Double

        Return ((total_anode_mass / anode_number))

    End Function

End Class

Public Class cost

    Public Function total_cost(ByVal anodenumber As Double, ByVal cost_per_anode As
Double, ByVal installation_cost_per_anode As Double) As Double

        Return ((anodenumber * cost_per_anode) + (anodenumber *
installation_cost_per_anode))

    End Function

End Class

Public Class attenuation

End Class

Public Class wrappercls

#Region "declarations"

#Region "declaration for startup "

    'from the start up form

    Private m_client, m_projectid, m_projectversion As String

    Private m_conformity, m_projectype, m_cp_type, m_input_values, m_output_values As
String

    Private m_designlife As Integer

    Private m_rectifier_adjustment_factor As Double

#End Region

#Region "declaration for structures"

    'from structure form

    'physical properties

    Private m_structure_shape As String
```

```
    Private m_structure_length, m_structure_outer_radius, m_structure_inner_radius,
m_density As Double

    'metallurgical properties

    Private m_material_type, m_material, m_material_specification, m_coating_type As
String

    'electrochemical property

    Private m_corrosion_potential, m_resistivity As Double

    Private m_reference_cell As String

    'coating break down factor

    Private m_coating_resistance_sqrm, m_initial_coating_factor, m_final_coating_factor
As Double

#End Region

#Region "declaration for environment form"

    ' from environment class

    Private m_climatic_region, m_environment_type, m_environment_sub_type,
m_environment_ph As String

    Private m_env_resistivity, m_depth, m_temperature, m_electrolyte_resistance,
m_lead_wire_resistance, m_initial_current_density, m_av_current_density,
m_final_current_density As Double

#End Region

#Region "from anode form"

    'from anode form

    'physical property

    Private m_anode_shape, m_anode_type As String

    Private m_anode_length, m_anode_outer_radius, m_anode_inner_radius,
m_anode_core_radius, m_anode_stand_off, m_anode_density As Double

    'anode spacing

    Private m_anode_spacing As Double

    Private m_anode_spacing_type As Boolean

    'electrochemical properties

    Private m_anode_alloy_type As String

    Private m_anode_close_circuit_potential, m_anode_utilization_factor, m_anode_mass,
m_anode_electro_capacity As Double

    'cost

    Private m_anode_cost, m_installation_cost As Double
```

```
#End Region

#Region "function return values"

    'function return values

    Private m_final_anode_length, m_final_anode_radius, m_initial_resistance,
m_final_resistance, m_anode_surface_area, _

 m_final_anode_surface_area, m_structure_surface_area, m_structure_thickness As Double

    Private m_current_demand, m_final_current_demand, m_ave_current_demand,
m_initial_anode_mass, m_final_anode_mass, _

m_driving_potential, m_total_CP_current, m_number_of_anodes, _

 m_total_resistance, m_project_cost, m_current_capacity As Double

    Private m_anode_final_current_output, m_anode_initial_current_output,
m_adjusted_anode_current, m_rectifier_voltage As Double

    Private m_anode_adjustment_factor, m_structure_resistance, m_slope_parameter As
Double

    Private m_check_potential, m_check_av_current, m_check_initial_current,
m_check_final_current As Boolean

#End Region

#Region "declaration optimization variables"

    'optimization variables

    Private m_opt_anode_mass, m_opt_anode_resistance, m_opt_anode_length,
m_opt_anode_radius, m_opt_anode_spacing, m_opt_anode_number As Double

    Private m_opt_inputed_values As Double

    Private m_opt_current_capacity, m_opt_designlife, m_opt_slope_parameter,
m_opt_ave_current_demand As Double

    Private m_opt_structure_length, m_opt_anode_total_mass, m_opt_env_resistivity,
m_coating_resistance As Double

    Private m_opt_anode_type, m_opt_anode_shape As String

#End Region

#End Region

#Region "declaration from conversion and corrosion rate"

    Private m_coupon_calculation_type, m_coupon_metal_type, m_coupon_type,
m_coupon_shape As String

    Private m_coupon_initial_weight, m_coupon_final_weight, m_coupon_metal_density,
m_coupon_exposure_time As Double

    Private m_coupon_length, m_coupon_width, m_coupon_area, m_coupon_calcution_result As
Double

    Private m_coupon_area_module As Boolean
```

```
    Private m_conversion_input_value, m_conversion_output_value As Double
    Private m_conversion_output_units, m_conversion_input_units As String
    Private m_conversion_electron As Integer
    Private m_conversion_atomic_mass As Double
    Private m_conversion_density As Double
#End Region
#Region " property values  "
#Region "values from start form"
    'values from  start form
    Public Property client() As String
        Get
            client = m_client
        End Get
        Set(ByVal value As String)
            If value = String.Empty Then
                value = "empty"
            Else
                m_client = value
            End If
        End Set
    End Property
    Public Property projectid() As String
        Get
            projectid = m_projectid
        End Get
        Set(ByVal value As String)
            m_projectid = value
        End Set
    End Property
    Public Property projectversion() As String
        Get
```

```
            projectversion = m_projectversion
        End Get
        Set(ByVal value As String)
            m_projectversion = value
        End Set
    End Property
    Public Property conformity() As String
        Get
            conformity = m_conformity
        End Get
        Set(ByVal value As String)
            m_conformity = value
        End Set
    End Property
    Public Property projectype() As String
        Get
            projectype = m_projectype
        End Get
        Set(ByVal value As String)
            m_projectype = value
        End Set
    End Property
    Public Property cp_type() As String
        Get
            cp_type = m_cp_type
        End Get
        Set(ByVal value As String)
            m_cp_type = value
        End Set
    End Property
    Public Property designlife() As Integer
```

```
    Get
        designlife = m_designlife
    End Get
    Set(ByVal value As Integer)
        m_designlife = value
    End Set
End Property
Public Property input_values() As String
    Get
        input_values = m_input_values
    End Get
    Set(ByVal value As String)
        m_input_values = value
    End Set
End Property
Public Property output_values() As String
    Get
        output_values = m_output_values
    End Get
    Set(ByVal value As String)
        m_output_values = value
    End Set
End Property
Public Property rectifier_adjustment_factor() As Double
    Get
        rectifier_adjustment_factor = m_rectifier_adjustment_factor
    End Get
    Set(ByVal value As Double)
        m_rectifier_adjustment_factor = (value / 100)
    End Set
End Property
```

```
#End Region
#Region "values from structure form "
    'values from structure form
    Public Property structure_shape() As String
        Get
            structure_shape = m_structure_shape
        End Get
        Set(ByVal value As String)
            m_structure_shape = value
        End Set
    End Property
    Public Property structure_length() As Double
        Get
            structure_length = m_structure_length
        End Get
        Set(ByVal value As Double)
            m_structure_length = value
        End Set
    End Property
    Public Property outer_radius() As Double
        Get
            outer_radius = m_structure_outer_radius
        End Get
        Set(ByVal value As Double)
            m_structure_outer_radius = value
        End Set
    End Property
    Public Property structure_inner_radius() As Double
        Get
            structure_inner_radius = m_structure_inner_radius
        End Get
```

```
    Set(ByVal value As Double)
        m_structure_inner_radius = value
    End Set
End Property
Public Property density() As Double
    Get
        density = m_density
    End Get
    Set(ByVal value As Double)
        m_density = value
    End Set
End Property

Public Property material_type() As String
    Get
        material_type = m_material_type
    End Get
    Set(ByVal value As String)
        m_material_type = value
    End Set
End Property
Public Property material() As String
    Get
        material = m_material
    End Get
    Set(ByVal value As String)
        m_material = value
    End Set
End Property
Public Property material_specification() As String
    Get
```

```
            material_specification = m_material_specification
        End Get
        Set(ByVal value As String)
            m_material_specification = value
        End Set
    End Property
    Public Property corrosion_potential() As Double
        Get
            corrosion_potential = m_corrosion_potential
        End Get
        Set(ByVal value As Double)
            m_corrosion_potential = value
        End Set
    End Property
    Public Property reference_cell() As String
        Get
            reference_cell = m_reference_cell
        End Get
        Set(ByVal value As String)
            m_reference_cell = value
        End Set
    End Property
    Public Property structure_resistivity() As Double
        Get
            structure_resistivity = m_resistivity
        End Get
        Set(ByVal value As Double)
            m_resistivity = value
        End Set
    End Property
    Public Property coating_type() As String
```

```
    Get
        coating_type = m_coating_type
    End Get
    Set(ByVal value As String)
        m_coating_type = value
    End Set
End Property
Public Property initial_coating_factor() As Double
    Get
        initial_coating_factor = m_initial_coating_factor
    End Get
    Set(ByVal value As Double)
        m_initial_coating_factor = (value / 100)
    End Set
End Property
Public Property final_coating_factor() As Double
    Get
        final_coating_factor = m_final_coating_factor
    End Get
    Set(ByVal value As Double)
        m_final_coating_factor = (value / 100)
    End Set
End Property
Public Property coating_resistance_sqrm() As Double
    Get
        coating_resistance_sqrm = m_coating_resistance_sqrm
    End Get
    Set(ByVal value As Double)
        m_coating_resistance_sqrm = value
    End Set
End Property
```

```
#End Region
#Region "value from environment form"
    'value from environment form
    Public Property climatic_region() As String
        Get
            climatic_region = m_climatic_region
        End Get
        Set(ByVal value As String)
            m_climatic_region = value
        End Set
    End Property
    Public Property environment_type() As String
        Get
            environment_type = m_environment_type
        End Get
        Set(ByVal value As String)
            m_environment_type = value
        End Set
    End Property
    Public Property environment_sub_type() As String
        Get
            environment_sub_type = m_environment_sub_type
        End Get
        Set(ByVal value As String)
            m_environment_sub_type = value
        End Set
    End Property
    Public Property environment_ph() As String
        Get
            environment_ph = m_environment_ph
        End Get
```

```
    Set(ByVal value As String)
        m_environment_ph = value
    End Set
End Property
Public Property env_resistivity() As Double
    Get
        env_resistivity = m_env_resistivity
    End Get
    Set(ByVal value As Double)
        m_env_resistivity = value
    End Set
End Property

Public Property depth() As Double
    Get
        depth = m_depth
    End Get
    Set(ByVal value As Double)
        m_depth = value
    End Set
End Property
Public Property temperature() As Double
    Get
        temperature = m_temperature
    End Get
    Set(ByVal value As Double)
        m_temperature = value
    End Set
End Property
Public Property lead_wire_resistance() As Double
    Get
```

```
            lead_wire_resistance = m_lead_wire_resistance
        End Get
        Set(ByVal value As Double)
            m_lead_wire_resistance = value
        End Set
    End Property
    Public Property electrolyte_resistance() As Double
        Get
            electrolyte_resistance = m_electrolyte_resistance
        End Get
        Set(ByVal value As Double)
            m_electrolyte_resistance = value
        End Set
    End Property
    Public Property initial_current_density() As Double
        Get
            initial_current_density = m_initial_current_density
        End Get
        Set(ByVal value As Double)
            m_initial_current_density = value
        End Set
    End Property
    Public Property av_current_density() As Double
        Get
            av_current_density = m_av_current_density
        End Get
        Set(ByVal value As Double)
            m_av_current_density = value
        End Set
    End Property
    Public Property final_current_density() As Double
```

```
        Get
            final_current_density = m_final_current_density
        End Get
        Set(ByVal value As Double)
            m_final_current_density = value
        End Set
    End Property
#End Region
#Region "from anode form"
    'value from anode form
    'physical property
    Public Property anode_shape() As String
        Get
            anode_shape = m_anode_shape
        End Get
        Set(ByVal value As String)
            m_anode_shape = value
        End Set
    End Property
    Public Property anode_type() As String
        Get
            anode_type = m_anode_type
        End Get
        Set(ByVal value As String)
            m_anode_type = value
        End Set
    End Property
    Public Property anode_length() As Double
        Get
            anode_length = m_anode_length
        End Get
```

```
    Set(ByVal value As Double)
        m_anode_length = value
    End Set
End Property
Public Property anode_outer_radius() As Double
    Get
        anode_outer_radius = m_anode_outer_radius
    End Get
    Set(ByVal value As Double)
        m_anode_outer_radius = value
        If m_anode_shape = "Cylindrical" Then
            m_anode_outer_radius = value
        Else
            m_anode_outer_radius = ((value) / (2 * PI))
        End If
    End Set
End Property
Public Property anode_core_radius() As Double
    Get
        anode_core_radius = m_anode_core_radius
    End Get
    Set(ByVal value As Double)
        m_anode_core_radius = value
    End Set
End Property
Public Property anode_stand_off() As Double
    Get
        anode_stand_off = m_anode_stand_off
    End Get
    Set(ByVal value As Double)
        m_anode_stand_off = value
```

```
    End Set
End Property
Public Property anode_density() As Double
    Get
        anode_density = m_anode_density
    End Get
    Set(ByVal value As Double)
        m_anode_density = value
    End Set
End Property
'electrochemical properties
Public Property anode_close_circuit_potential() As Double
    Get
        anode_close_circuit_potential = m_anode_close_circuit_potential
    End Get
    Set(ByVal value As Double)
        m_anode_close_circuit_potential = value
    End Set
End Property
Public Property anode_utilization_factor() As Double
    Get
        anode_utilization_factor = m_anode_utilization_factor
    End Get
    Set(ByVal value As Double)
        m_anode_utilization_factor = (value / 100)
    End Set
End Property
Public Property anode_electro_capacity() As Double
    Get
        anode_electro_capacity = m_anode_electro_capacity
    End Get
```

```
    Set(ByVal value As Double)
        m_anode_electro_capacity = value
    End Set
End Property
Public Property anode_mass() As Double
    Get
        anode_mass = m_anode_mass
    End Get
    Set(ByVal value As Double)
        m_anode_mass = value
    End Set
End Property
Public Property anode_alloy_type() As String
    Get
        anode_alloy_type = m_anode_alloy_type
    End Get
    Set(ByVal value As String)
        m_anode_alloy_type = value
    End Set
End Property
'cost
Public Property anode_cost() As Double
    Get
        anode_cost = m_anode_cost
    End Get
    Set(ByVal value As Double)
        m_anode_cost = value
    End Set
End Property
Public Property installation_cost() As Double
    Get
```

```
            installation_cost = m_installation_cost
        End Get
        Set(ByVal value As Double)
            m_installation_cost = value
        End Set
    End Property
#End Region
#Region "from core cpdexfunction "
#Region "main function"
    'return values from function
    Public Property anode_initial_resistance() As Double
        Get
            anode_initial_resistance = m_initial_resistance
        End Get
        Set(ByVal value As Double)
            m_initial_resistance = value
        End Set
    End Property
    Public Property anode_final_resistance() As Double
        Get
            anode_final_resistance = m_final_resistance
        End Get
        Set(ByVal value As Double)
            m_final_resistance = value
        End Set
    End Property
    Public Property anode_surface_area() As Double
        Get
            anode_surface_area = m_anode_surface_area
        End Get
        Set(ByVal value As Double)
```

```
        m_anode_surface_area = value
    End Set
End Property
Public Property structure_surface_area() As Double
    Get
        structure_surface_area = m_structure_surface_area
    End Get
    Set(ByVal value As Double)
        m_structure_surface_area = value
    End Set
End Property
Public Property final_anode_length() As Double
    Get
        final_anode_length = m_final_anode_length
    End Get
    Set(ByVal value As Double)
        m_final_anode_length = value
    End Set
End Property
Public Property final_anode_radius() As Double
    Get
        final_anode_radius = m_final_anode_radius
    End Get
    Set(ByVal value As Double)
        m_final_anode_radius = value
    End Set
End Property
Public Property final_anode_surface_area() As Double
    Get
        final_anode_surface_area = m_final_anode_surface_area
    End Get
```

```
    Set(ByVal value As Double)
        m_final_anode_surface_area = value
    End Set
End Property
Public Property anode_inner_radius() As Double
    Get
        anode_inner_radius = m_anode_inner_radius
    End Get
    Set(ByVal value As Double)
        m_anode_inner_radius = value
    End Set
End Property
Public Property structure_thickness() As Double
    Get
        structure_thickness = m_structure_thickness
    End Get
    Set(ByVal value As Double)
        m_structure_thickness = value
    End Set
End Property



Public Property initial_current_demand() As Double
    Get
        initial_current_demand = m_current_demand
    End Get
    Set(ByVal value As Double)
        m_current_demand = value
    End Set
End Property
Public Property final_current_demand() As Double
```

```
    Get
        final_current_demand = m_final_current_demand
    End Get
    Set(ByVal value As Double)
        m_final_current_demand = value
    End Set
End Property
Public Property ave_current_demand() As Double
    Get
        ave_current_demand = m_ave_current_demand
    End Get
    Set(ByVal value As Double)
        m_ave_current_demand = value
    End Set
End Property
Public Property initial_anode_mass() As Double
    Get
        initial_anode_mass = m_initial_anode_mass
    End Get
    Set(ByVal value As Double)
        m_initial_anode_mass = value
    End Set
End Property
Public Property final_anode_mass() As Double
    Get
        final_anode_mass = m_final_anode_mass
    End Get
    Set(ByVal value As Double)
        m_final_anode_mass = value
    End Set
End Property
```

```
Public Property driving_potential() As Double
    Get
        driving_potential = m_driving_potential
    End Get
    Set(ByVal value As Double)
        m_driving_potential = value
    End Set
End Property
Public Property total_CP_current() As Double
    Get
        total_CP_current = m_total_CP_current
    End Get
    Set(ByVal value As Double)
        m_total_CP_current = value
    End Set
End Property
Public Property number_of_anodes() As Double
    Get
        number_of_anodes = m_number_of_anodes
    End Get
    Set(ByVal value As Double)
        m_number_of_anodes = value
    End Set
End Property
Public Property total_resistance() As Double
    Get
        total_resistance = m_total_resistance
    End Get
    Set(ByVal value As Double)
        m_total_resistance = value
    End Set
```

```
End Property
Public Property project_cost() As Double
    Get
        project_cost = m_project_cost
    End Get
    Set(ByVal value As Double)
        m_project_cost = value
    End Set
End Property
Public Property current_capacity() As Double
    Get
        current_capacity = m_current_capacity
    End Get
    Set(ByVal value As Double)
        m_current_capacity = value
    End Set
End Property
Public Property anode_initial_current_output() As Double
    Get
        anode_initial_current_output = m_anode_initial_current_output
    End Get
    Set(ByVal value As Double)
        m_anode_initial_current_output = value
    End Set
End Property
Public Property anode_final_current_output() As Double
    Get
        anode_final_current_output = m_anode_final_current_output
    End Get
    Set(ByVal value As Double)
        m_anode_final_current_output = value
```

```
    End Set
End Property
Public Property anode_spacing() As Double
    Get
        anode_spacing = m_anode_spacing
    End Get
    Set(ByVal value As Double)
        m_anode_spacing = value
    End Set
End Property
Public Property anode_adjustment_factor() As Double
    Get
        anode_adjustment_factor = m_anode_adjustment_factor
    End Get
    Set(ByVal value As Double)
        m_anode_adjustment_factor = value
    End Set
End Property
Public Property adjusted_anode_current() As Double
    Get
        adjusted_anode_current = m_adjusted_anode_current
    End Get
    Set(ByVal value As Double)
        m_adjusted_anode_current = value
    End Set
End Property
Public Property rectifier_voltage() As Double
    Get
        rectifier_voltage = m_rectifier_voltage
    End Get
    Set(ByVal value As Double)
```

```
        m_rectifier_voltage = value
    End Set
End Property
Public Property structure_resistance() As Double
    Get
        structure_resistance = m_structure_resistance
    End Get
    Set(ByVal value As Double)
        m_structure_resistance = value
    End Set
End Property
Public Property slope_parameter() As Double
    Get
        slope_parameter = m_slope_parameter
    End Get
    Set(ByVal value As Double)
        m_slope_parameter = value
    End Set
End Property
Public Property coating_resistance() As Double
    Get
        coating_resistance = m_coating_resistance
    End Get
    Set(ByVal value As Double)
        m_coating_resistance = value
    End Set
End Property
Public Property check_potential() As Boolean
    Get
        check_potential = m_check_potential
    End Get
```

```
        Set(ByVal value As Boolean)
            m_check_potential = value
        End Set
    End Property
    Public Property check_av_current() As Boolean
        Get
            check_av_current = m_check_av_current
        End Get
        Set(ByVal value As Boolean)
            m_check_av_current = value
        End Set
    End Property
    Public Property check_initial_current() As Boolean
        Get
            check_initial_current = m_check_initial_current
        End Get
        Set(ByVal value As Boolean)
            m_check_initial_current = value
        End Set
    End Property
    Public Property check_final_current() As Boolean
        Get
            check_final_current = m_check_final_current
        End Get
        Set(ByVal value As Boolean)
            m_check_final_current = value
        End Set
    End Property
#End Region
#Region "opimization propeties"
    'opimization propeties
```

```
Public Property opt_anode_mass() As Double
    Get
        opt_anode_mass = m_opt_anode_mass
    End Get
    Set(ByVal value As Double)
        m_opt_anode_mass = value
    End Set
End Property
Public Property opt_anode_resistance() As Double
    Get
        opt_anode_resistance = m_opt_anode_resistance
    End Get
    Set(ByVal value As Double)
        m_opt_anode_resistance = value
    End Set
End Property
Public Property opt_anode_length() As Double
    Get
        opt_anode_length = m_opt_anode_length
    End Get
    Set(ByVal value As Double)
        m_opt_anode_length = value
    End Set
End Property
Public Property opt_anode_radius() As Double
    Get
        opt_anode_radius = m_opt_anode_radius
    End Get
    Set(ByVal value As Double)
        m_opt_anode_radius = value
    End Set
```

```
End Property
Public Property opt_anode_spacing() As Double
    Get
        opt_anode_spacing = m_opt_anode_spacing
    End Get
    Set(ByVal value As Double)
        m_opt_anode_spacing = value
    End Set
End Property
Public Property opt_anode_number() As Double
    Get
        opt_anode_number = m_opt_anode_number
    End Get
    Set(ByVal value As Double)
        m_opt_anode_number = value
    End Set
End Property
Public Property opt_inputed_values() As Double
    Get
        opt_inputed_values = m_opt_inputed_values
    End Get
    Set(ByVal value As Double)
        m_opt_inputed_values = value
    End Set
End Property
Public Property opt_current_capacity() As Double
    Get
        opt_current_capacity = m_opt_current_capacity
    End Get
    Set(ByVal value As Double)
        m_opt_current_capacity = value
```

```
    End Set
End Property
Public Property opt_designlife() As Double
    Get
        opt_designlife = m_opt_designlife
    End Get
    Set(ByVal value As Double)
        m_opt_designlife = value
    End Set
End Property
Public Property opt_slope_parameter() As Double
    Get
        opt_slope_parameter = m_opt_slope_parameter
    End Get
    Set(ByVal value As Double)
        m_opt_slope_parameter = value
    End Set
End Property
Public Property opt_ave_current_demand() As Double
    Get
        opt_ave_current_demand = m_opt_ave_current_demand
    End Get
    Set(ByVal value As Double)
        m_opt_ave_current_demand = value
    End Set
End Property
Public Property opt_structure_length() As Double
    Get
        opt_structure_length = m_opt_structure_length
    End Get
    Set(ByVal value As Double)
```

```
        m_opt_structure_length = value
    End Set
End Property
Public Property opt_anode_total_mass() As Double
    Get
        opt_anode_total_mass = m_opt_anode_total_mass
    End Get
    Set(ByVal value As Double)
        m_opt_anode_total_mass = value
    End Set
End Property
Public Property opt_env_resistivity() As Double
    Get
        opt_env_resistivity = m_opt_env_resistivity
    End Get
    Set(ByVal value As Double)
        m_opt_env_resistivity = value
    End Set
End Property
Public Property opt_anode_type() As String
    Get
        opt_anode_type = m_opt_anode_type
    End Get
    Set(ByVal value As String)
        m_opt_anode_type = value
    End Set
End Property
Public Property opt_anode_shape() As String
    Get
        opt_anode_shape = m_opt_anode_shape
    End Get
```

```
        Set(ByVal value As String)
            m_opt_anode_shape = value
        End Set
    End Property
#End Region
#End Region
#Region "value from conversion and corrosion rate"
    Public Property coupon_calculation_type() As String
        Get
            coupon_calculation_type = m_coupon_calculation_type
        End Get
        Set(ByVal value As String)
            m_coupon_calculation_type = value
        End Set
    End Property
    Public Property coupon_metal_type() As String
        Get
            coupon_metal_type = m_coupon_metal_type
        End Get
        Set(ByVal value As String)
            m_coupon_metal_type = value
        End Set
    End Property
    Public Property coupon_type() As String
        Get
            coupon_type = m_coupon_type
        End Get
        Set(ByVal value As String)
            m_coupon_type = value
        End Set
    End Property
```

```
Public Property coupon_initial_weight() As Double
    Get
        coupon_initial_weight = m_coupon_initial_weight
    End Get
    Set(ByVal value As Double)
        m_coupon_initial_weight = value
    End Set
End Property
Public Property coupon_final_weight() As Double
    Get
        coupon_final_weight = m_coupon_final_weight
    End Get
    Set(ByVal value As Double)
        m_coupon_final_weight = value
    End Set
End Property
Public Property coupon_metal_density() As Double
    Get
        coupon_metal_density = m_coupon_metal_density
    End Get
    Set(ByVal value As Double)
        m_coupon_metal_density = value
    End Set
End Property
Public Property coupon_exposure_time() As Double
    Get
        coupon_exposure_time = m_coupon_exposure_time
    End Get
    Set(ByVal value As Double)
        m_coupon_exposure_time = value
    End Set
```

```
End Property
Public Property coupon_length() As Double
    Get
        coupon_length = m_coupon_length
    End Get
    Set(ByVal value As Double)
        m_coupon_length = value
    End Set
End Property
Public Property coupon_width() As Double
    Get
        coupon_width = m_coupon_width
    End Get
    Set(ByVal value As Double)
        m_coupon_width = value
    End Set
End Property
Public Property coupon_area() As Double
    Get
        coupon_area = m_coupon_area
    End Get
    Set(ByVal value As Double)
        m_coupon_area = value
    End Set
End Property
Public Property coupon_calcution_result() As Double
    Get
        coupon_calcution_result = m_coupon_calcution_result
    End Get
    Set(ByVal value As Double)
        m_coupon_calcution_result = value
```

```
    End Set
End Property
Public Property coupon_area_module() As Boolean
    Get
        coupon_area_module = m_coupon_area_module
    End Get
    Set(ByVal value As Boolean)
        m_coupon_area_module = value
    End Set
End Property
Public Property coupon_shape() As String
    Get
        coupon_shape = m_coupon_shape
    End Get
    Set(ByVal value As String)
        m_coupon_shape = value
    End Set
End Property
Public Property conversion_input_value() As Double
    Get
        conversion_input_value = m_conversion_input_value
    End Get
    Set(ByVal value As Double)
        m_conversion_input_value = value
    End Set
End Property
Public Property conversion_output_value() As Double
    Get
        conversion_output_value = m_conversion_output_value
    End Get
    Set(ByVal value As Double)
```

```
            m_conversion_output_value = value
        End Set
    End Property
    Public Property conversion_output_units() As String
        Get
            conversion_output_units = m_conversion_output_units
        End Get
        Set(ByVal value As String)
            m_conversion_output_units = value
        End Set
    End Property
    Public Property conversion_input_units() As String
        Get
            conversion_input_units = m_conversion_input_units
        End Get
        Set(ByVal value As String)
            m_conversion_input_units = value

        End Set
    End Property
    Public Property conversion_electron() As Integer
        Get
            conversion_electron = m_conversion_electron
        End Get
        Set(ByVal value As Integer)
            m_conversion_electron = value
        End Set
    End Property
    Public Property conversion_atomic_mass() As Double
        Get
            conversion_atomic_mass = m_conversion_atomic_mass
```

```
        End Get
        Set(ByVal value As Double)
            m_conversion_atomic_mass = value
        End Set
    End Property
    Public Property conversion_density() As Double
        Get
            conversion_density = m_conversion_density
        End Get
        Set(ByVal value As Double)
            m_conversion_density = value
        End Set
    End Property
#End Region
#End Region
End Class
Public Class errorcls
    Public Sub New()


    End Sub
    Public Function improperval(ByVal et As Control) As DialogResult
        Return (MessageBox.Show(" The inputed value type in " & et.Text & " control is
not accepted", "Error Meassage", MessageBoxButtons.OK, MessageBoxIcon.Error,
MessageBoxDefaultButton.Button1, MessageBoxOptions.ServiceNotification))
    End Function
    Public Function t_empty(ByVal et As Control) As DialogResult
        Return (MessageBox.Show("No value was inputed into " & et.Text & " the textbox
control!", "Error Meassage", MessageBoxButtons.OK, MessageBoxIcon.Error,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly))
    End Function
    Public Function l_empty(ByVal et As Control) As DialogResult
        Return (MessageBox.Show("No value was selected from the " & et.Text & " list
control!", "Error Meassage", MessageBoxButtons.OK, MessageBoxIcon.Error,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly))
```

```
    End Function

    Public Function toolarge(ByVal et As Control, ByVal limit_value As String) As
DialogResult

        Return (MessageBox.Show("Inputed value in " & et.Text & " control is too high!
Value cannot exceed" & limit_value.ToString & "", "Error Meassage",
MessageBoxButtons.OK, MessageBoxIcon.Error, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly))

    End Function

    Public Function caution(ByVal et As Control) As DialogResult

        Return (MessageBox.Show("The Value in " & et.Text & " control may lead to
unexpected result, and this value does not make much sense in this context, do yo wish
to proceed anywhere!", "Error Meassage", MessageBoxButtons.OKCancel,
MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly))

    End Function

    Public Function moduleabsent() As DialogResult

        Dim result As DialogResult

        result = MessageBox.Show("The module your intending to use is not avaliable in
this version of CPDEx, for professional edtion, ask your software vendor", "Error
Meassage", MessageBoxButtons.OK, MessageBoxIcon.Error, MessageBoxDefaultButton.Button1,
MessageBoxOptions.DefaultDesktopOnly)

        Return (result)

    End Function

    Public Function about_2_changesetting() As DialogResult

        Return (MessageBox.Show("With this action, all the inputed values on this form
will be lost", "Error Meassage", MessageBoxButtons.YesNoCancel, MessageBoxIcon.Warning,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly))

    End Function

    Public Function oversight(ByVal et As Control) As DialogResult

        Return (MessageBox.Show("The " & et.Text & " control is empty, is it
idelibrate?", "Error Meassage", MessageBoxButtons.YesNo, MessageBoxIcon.Warning,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly))

    End Function

    Public Function module_not_developed(ByVal et As Control) As DialogResult

        Return (MessageBox.Show("The module " & et.Text & " has not yet been developed,
if your interested in developing modules for CPDEx please refer to 'ABOUT CPDEx' !",
"Error Meassage", MessageBoxButtons.OK, MessageBoxIcon.Error,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly))

    End Function

End Class
```

```
Public Class conversion

    Public Function corrosion_conversion(ByVal input_units As String, ByVal output_units
As String, ByVal input_value As Double, ByVal conversion_electron As Integer, ByVal
conversion_density As Double, _

    ByVal conversion_atomic_mass As Double) As Double

        Dim c_input_value As Double

        If input_units = "mA/cm^2" Then

            c_input_value = input_value

        ElseIf input_units = "mm/yr^1" Then

            c_input_value = (input_value * 0.306 * ((conversion_density *
conversion_electron) / (conversion_atomic_mass)))

        ElseIf input_units = "mpy" Then

            c_input_value = (input_value * 0.00777 * ((conversion_density *
conversion_electron) / (conversion_atomic_mass)))

        ElseIf input_units = "g/m^2 Day" Then

            c_input_value = (input_value * 0.112 * ((conversion_electron) /
(conversion_atomic_mass)))

        End If

        If output_units = "mA/cm^2" Then

            Return (c_input_value)

        ElseIf output_units = "mm/yr^1" Then

            Return ((3.28 * ((conversion_atomic_mass) / (conversion_density *
conversion_electron)) * c_input_value))

        ElseIf output_units = "mpy" Then

            Return ((129 * ((conversion_atomic_mass) / (conversion_density *
conversion_electron)) * c_input_value))

        ElseIf output_units = "g/m^2 Day" Then

            Return ((8.95 * ((conversion_atomic_mass) / (conversion_electron)) *
c_input_value))

        End If

    End Function

End Class

Public Class corrosion_rate

    Public Function corrosion_rate_calc(ByVal initial_weight As Double, ByVal
final_weight As Double, ByVal coupon_area As Double, ByVal exposure_time As Double,
ByVal coupon_density As Double, ByVal area_module As Boolean, ByVal coupon_shape As
String, _
```

```
    ByVal coupon_length As Double, ByVal coupon_width As Double) As Double
        Dim c_area As Double
        If area_module = True Then
            c_area = coupon_area
        Else
            If coupon_shape = "Rectangular" Then
                c_area = 2 * coupon_length * coupon_width
            ElseIf coupon_shape = "Disc" Then
                c_area = 2 * PI * coupon_width ^ 2
            ElseIf coupon_shape = "Rod" Then
                c_area = 2 * PI * coupon_width * coupon_length
            End If
        End If
        Return (((initial_weight - final_weight) / c_area) * ((3.65 * 10 ^ 6) /
(exposure_time * coupon_density)))
    End Function
    Public Function pitting_rate(ByVal pit_depth As Double, ByVal exposure_time As
Double) As Double
        Return (((pit_depth * 365) / (exposure_time)))
    End Function
End Class
```

Structure

```
Option Explicit On
Option Strict On
Public Class structurefrm
#Region "declaration"
    'physical properties
    Dim shape As String
    Dim length, outer_radius, inner_radius, density As Double
    'metallurgical properties
    Dim material_type, material, material_specification, coating_type As String
```

```
    'electrochemical property
    Dim corrosion_potential, current_density, resistivity As Double
    Dim reference_cell As String
    'coating break down factor
    Dim initial_coating_factor, final_coating_factor, coating_resistance As Double
    'error message delegate
    Dim result As DialogResult
    Dim errt As ch_setting = AddressOf err.about_2_changesetting
    Dim emp_l As empt_l = AddressOf err.l_empty
    Dim emp_t As empt_t = AddressOf err.t_empty
    Dim absent As module_absent = AddressOf err.moduleabsent
    Dim imprp As impro = AddressOf err.improperval
    Dim caus As causion = AddressOf err.caution
    Dim ovsight As oversite = AddressOf err.oversight
    Dim check_thickness_error As d_thickness_error = AddressOf
area.check_thickness_error
    Dim itm1, itm2, itm3, itm4 As String
#End Region
#Region "key control events"
#Region " lststructure_shape"
    'Private Sub lststructure_shape_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lststructure_shape.LostFocus


    'End Sub
    'Private Sub lststructure_shape_GotFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lststructure_shape.GotFocus


    'End Sub
    'Private Sub lststructure_shape_SelectedIndexChanged(ByVal sender As Object, ByVal e
As System.EventArgs) Handles lststructure_shape.SelectedIndexChanged
    '    'itm1 = Me.lststructure_shape.SelectedItem.ToString()
    '    If itm1 = "Cubic" Then
    '        resetbck(Me.lststructure_shape)
```

```
    '        shape = lststructure_shape.SelectedItem.ToString

    '    ElseIf itm1 = "Cylindrical" Then

    '        resetbck(Me.lststructure_shape)

    '        shape = lststructure_shape.SelectedItem.ToString

    '    ElseIf itm1 = "Spherical" Then

    '        result = absent()

    '        result = Windows.Forms.DialogResult.OK

    '        resetbck(Me.lststructure_shape)

    '        Me.lststructure_shape.Focus()

    '    End If

    'End Sub

    'Private Sub lststructure_shape_SelectedValueChanged(ByVal sender As Object, ByVal e
As System.EventArgs) Handles lststructure_shape.SelectedValueChanged

    '    'If itm1 = String.Empty Then

    '    '    result = emp_l(Me.lblstructure_shape)

    '    '    result = Windows.Forms.DialogResult.OK

    '    'Else

    '    '    shape = itm1

    '    'End If

    'End Sub

    Private Sub lststructure_shape_Enter(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lststructure_shape.Enter

        get_focus_effect(Me.lststructure_shape)

    End Sub

    Private Sub lststructure_shape_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lststructure_shape.Leave

        resetbck(Me.lststructure_shape)

    End Sub

    Private Sub lststructure_shape_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles lststructure_shape.MouseClick

        resetbck(Me.lststructure_shape)

        Try

            itm1 = Me.lststructure_shape.SelectedIndex.ToString
```

```
            If itm1 = "0" Then
                resetbck(Me.lststructure_shape)
                shape = lststructure_shape.SelectedItem.ToString
            ElseIf itm1 = "1" Then
                resetbck(Me.lststructure_shape)
                shape = lststructure_shape.SelectedItem.ToString
            ElseIf itm1 = "2" Then
                result = absent()
                result = Windows.Forms.DialogResult.OK
                resetbck(Me.lststructure_shape)
                Me.lststructure_shape.Focus()
                Me.lststructure_shape.ClearSelected()
                shape = String.Empty
            End If
        Catch ex As System.NullReferenceException When
Me.lststructure_shape.SelectedIndex < 0
            result = emp_l(Me.lblstructure_shape)
            result = Windows.Forms.DialogResult.OK
            Me.lststructure_shape.Focus()
            shape = String.Empty
        Catch ex As System.IndexOutOfRangeException When
Me.lststructure_shape.SelectedIndex > Me.lststructure_shape.Items.Count
            result = emp_l(Me.lblstructure_shape)
            result = Windows.Forms.DialogResult.OK
            Me.lststructure_shape.Focus()
            shape = String.Empty
        End Try
    End Sub
    Private Sub lststructure_shape_MouseDoubleClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles lststructure_shape.MouseDoubleClick
        itm1 = String.Empty
        resetbck(Me.lststructure_shape)
        Me.lststructure_shape.ClearSelected()
```

```
        shape = String.Empty

    End Sub

    Private Sub lststructure_shape_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lststructure_shape.Validated

        inter.structure_shape = shape

    End Sub

    Private Sub lststructure_shape_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles lststructure_shape.Validating

        If shape = String.Empty Then

            e.Cancel = True

            result = emp_l(Me.lblstructure_shape)

            result = Windows.Forms.DialogResult.OK

            Me.lststructure_shape.Focus()

        End If

    End Sub


#End Region

#Region "txtlength"

    Private Sub txtlength_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles txtlength.Enter

        get_focus_effect(Me.txtlength)

    End Sub

    Private Sub txtlength_Leave(ByVal sender As Object, ByVal e As System.EventArgs)
Handles txtlength.Leave

        resetbck(Me.txtlength)

    End Sub

    Private Sub txtlength_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles txtlength.MouseClick

        resetbck(Me.txtlength)

    End Sub

    Private Sub txtlength_Validated(ByVal sender As Object, ByVal e As System.EventArgs)
Handles txtlength.Validated

        inter.structure_length = length

    End Sub
```

```
    Private Sub txtlength_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles txtlength.Validating

        Dim ddi As Double

        Try

            ddi = CDbl(Me.txtlength.Text)

            If Not IsNumeric(ddi) OrElse ddi <= 0 Then

                result = imprp(Me.lblstrlength)

                result = Windows.Forms.DialogResult.OK

                Me.txtlength.Focus()

                e.Cancel = True

            ElseIf ddi <= Double.MaxValue Then

                length = ddi

            End If

        Catch ex As System.Exception

            result = imprp(Me.lblstrlength)

            result = Windows.Forms.DialogResult.OK

            e.Cancel = True

        End Try

    End Sub

#End Region

#Region "txtut_radius"


    Private Sub txtut_radius_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles txtut_radius.Enter

        get_focus_effect(Me.txtut_radius)

    End Sub

    Private Sub txtut_radius_Leave(ByVal sender As Object, ByVal e As System.EventArgs)
Handles txtut_radius.Leave

        resetbck(Me.txtut_radius)

    End Sub

    Private Sub txtut_radius_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles txtut_radius.MouseClick

        resetbck(Me.txtut_radius)
```

```
    End Sub

    Private Sub txtut_radius_TextChanged(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtut_radius.TextChanged

        If Me.txtut_radius.Text = String.Empty Then

            Me.txtin_radius.Enabled = False

        Else

            Me.txtin_radius.Enabled = True

        End If

    End Sub

    Private Sub txtut_radius_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtut_radius.Validated

        inter.outer_radius = outer_radius

    End Sub

    Private Sub txtut_radius_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles txtut_radius.Validating

        Dim ddi As Double

        Try

            ddi = CDbl(Me.txtut_radius.Text)

            If Not IsNumeric(ddi) OrElse ddi <= 0 Then

                result = imprp(Me.lblut_radius)

                result = Windows.Forms.DialogResult.OK

                Me.txtut_radius.Focus()

                e.Cancel = True

            ElseIf ddi <= Double.MaxValue Then

                outer_radius = ddi

                Me.txtin_radius.Enabled = True

            End If

        Catch ex As System.Exception

            result = imprp(Me.lblut_radius)

            result = Windows.Forms.DialogResult.OK

            e.Cancel = True

        End Try

    End Sub
```

```
#End Region

#Region "txtin_radius "

    Private Sub txtin_radius_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles txtin_radius.Enter

        get_focus_effect(Me.txtin_radius)

    End Sub

    Private Sub txtin_radius_Leave(ByVal sender As Object, ByVal e As System.EventArgs)
Handles txtin_radius.Leave

        resetbck(Me.txtin_radius)



    End Sub

    Private Sub txtin_radius_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles txtin_radius.MouseClick

        resetbck(Me.txtin_radius)

    End Sub

    Private Sub txtin_radius_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtin_radius.Validated

        inter.structure_inner_radius = inner_radius

    End Sub

    Private Sub txtin_radius_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles txtin_radius.Validating

        Dim ddi As Double

        Try

            ddi = CDbl(Me.txtin_radius.Text)

            If Not IsNumeric(ddi) OrElse ddi < 0 Then

                result = imprp(Me.lblin_radius)

                result = Windows.Forms.DialogResult.OK

                e.Cancel = True

                Me.txtin_radius.Focus()

            ElseIf check_thickness_error(Me.lblin_radius, CDbl(Me.txtut_radius.Text),
CDbl(Me.txtin_radius.Text)) = True Then

                e.Cancel = True

                Me.txtin_radius.Focus()
```

```
            ElseIf ddi <= Double.MaxValue Then
                inner_radius = ddi
            End If
        Catch ex As System.Exception
            result = imprp(Me.lblin_radius)
            result = Windows.Forms.DialogResult.OK
            e.Cancel = True
        End Try
    End Sub
#End Region
#Region "lstmaterial_type "
    'Private Sub lstmaterial_type_GotFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstmaterial_type.GotFocus


    'End Sub
    'Private Sub lstmaterial_type_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstmaterial_type.LostFocus


    'End Sub
    'Private Sub lstmaterial_type_SelectedIndexChanged(ByVal sender As Object, ByVal e
As System.EventArgs) Handles lstmaterial_type.SelectedIndexChanged
    '    'itm2 = Me.lstmaterial_type.SelectedIndex.ToString()
    '    'If itm2 = "Ferrous" Then
    '    '    resetbck(Me.lstmaterial_type)
    '    '    shape = lstmaterial_type.SelectedItem.ToString
    '    '    resetbck(Me.lststructure_shape)
    '    'ElseIf itm2 = "Non-Ferrous" Then
    '    '    resetbck(Me.lstmaterial_type)
    '    '    shape = lstmaterial_type.SelectedItem.ToString
    '    '    resetbck(Me.lststructure_shape)
    '    'End If
    'End Sub
```

```
    'Private Sub lstmaterial_type_SelectedValueChanged(ByVal sender As Object, ByVal e
As System.EventArgs) Handles lstmaterial_type.SelectedValueChanged

    '    'If itm2 = String.Empty Then

    '    '    result = emp_l(Me.lblmaterial_type)

    '    '    result = Windows.Forms.DialogResult.OK

    '    '    material_type = itm2

    '    'Else

    '    '    material_type = itm2

    '    'End If

    'End Sub

    Private Sub lstmaterial_type_Enter(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstmaterial_type.Enter

        get_focus_effect(Me.lstmaterial_type)

    End Sub

    Private Sub lstmaterial_type_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstmaterial_type.Leave

        resetbck(Me.lstmaterial_type)

    End Sub

    Private Sub lstmaterial_type_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles lstmaterial_type.MouseClick

        resetbck(Me.lstmaterial_type)

        Try

            itm2 = Me.lstmaterial_type.SelectedIndex.ToString

            If itm2 = "0" Then

                resetbck(Me.lstmaterial_type)

                material_type = lstmaterial_type.SelectedItem.ToString

                resetbck(Me.lststructure_shape)

            ElseIf itm2 = "1" Then

                resetbck(Me.lstmaterial_type)

                material_type = lstmaterial_type.SelectedItem.ToString

                resetbck(Me.lststructure_shape)

            End If

        Catch ex As System.NullReferenceException When Me.lstmaterial_type.SelectedIndex
< 0
```

```
            result = emp_l(Me.lblmaterial_type)

            result = Windows.Forms.DialogResult.OK

            Me.lstmaterial_type.Focus()

            material_type = String.Empty

        Catch ex As System.IndexOutOfRangeException When
Me.lstmaterial_type.SelectedIndex > Me.lstmaterial_type.Items.Count

            result = emp_l(Me.lblmaterial_type)

            result = Windows.Forms.DialogResult.OK

            Me.lstmaterial_type.Focus()

            material_type = String.Empty

        End Try

    End Sub

    Private Sub lstmaterial_type_MouseDoubleClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles lstmaterial_type.MouseDoubleClick

        itm2 = String.Empty

        Me.lstmaterial_type.ClearSelected()

        material_type = String.Empty

    End Sub

    Private Sub lstmaterial_type_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstmaterial_type.Validated

        inter.material_type = material_type

    End Sub

    Private Sub lstmaterial_type_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles lstmaterial_type.Validating

        If material_type = String.Empty Then

            e.Cancel = True

            result = emp_l(Me.lblmaterial_type)

            result = Windows.Forms.DialogResult.OK

            Me.lstmaterial_type.Focus()

        End If

    End Sub

#End Region

#Region "txtmaterial "
```

```
    Private Sub txtmaterial_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles txtmaterial.Enter

        get_focus_effect(Me.txtmaterial)

    End Sub

    Private Sub txtmaterial_Leave(ByVal sender As Object, ByVal e As System.EventArgs)
Handles txtmaterial.Leave

        resetbck(Me.txtmaterial)

    End Sub

    Private Sub txtmaterial_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles txtmaterial.MouseClick

        resetbck(Me.txtmaterial)

    End Sub

    Private Sub txtmaterial_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtmaterial.Validated

        inter.material = material

    End Sub

    Private Sub txtmaterial_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles txtmaterial.Validating

        Dim ddi As String

        Try

            ddi = Me.txtmaterial.Text

            If ddi = String.Empty Then

                result = emp_t(Me.lblmaterial)

                result = Windows.Forms.DialogResult.OK

                Me.txtmaterial.Focus()

                e.Cancel = True

            ElseIf IsNumeric(ddi) Then

                result = imprp(Me.lblmaterial)

                result = Windows.Forms.DialogResult.OK

                Me.txtmaterial.Focus()

                e.Cancel = True

            Else

                material = ddi

            End If
```

```
        Catch ex As System.Exception

            result = imprp(Me.lblmaterial)

            result = Windows.Forms.DialogResult.OK

            e.Cancel = True

        End Try

    End Sub

#End Region

#Region "txtmaterial_specification "


    Private Sub txtmaterial_specification_Enter(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtmaterial_specification.Enter

        get_focus_effect(Me.txtmaterial_specification)

    End Sub

    Private Sub txtmaterial_specification_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtmaterial_specification.Leave

        resetbck(Me.txtmaterial_specification)

    End Sub

    Private Sub txtmaterial_specification_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles txtmaterial_specification.MouseClick

        resetbck(Me.txtmaterial_specification)

    End Sub

    Private Sub txtmaterial_specification_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtmaterial_specification.Validated

        inter.material_specification = material_specification

    End Sub

    Private Sub txtmaterial_specification_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles txtmaterial_specification.Validating

        Dim ddi As String

        Try

            ddi = Me.txtmaterial_specification.Text

            If ddi = String.Empty Then

                result = emp_t(Me.lblmaterial_specification)

                result = Windows.Forms.DialogResult.OK

                Me.txtmaterial.Focus()
```

```
                e.Cancel = True
            ElseIf IsNumeric(ddi) Then
                result = imprp(Me.lblmaterial_specification)
                result = Windows.Forms.DialogResult.OK
                Me.txtmaterial.Focus()
                e.Cancel = True
            Else
                material_specification = ddi
            End If
        Catch ex As System.Exception
            result = imprp(Me.lblmaterial_specification)
            result = Windows.Forms.DialogResult.OK
            e.Cancel = True
        End Try
    End Sub
#End Region
#Region "lstreference_cell "
    Private Sub lstreference_cell_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstreference_cell.Leave
        resetbck(Me.lstreference_cell)
    End Sub
    Private Sub lstreference_cell_Enter(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstreference_cell.Enter
        get_focus_effect(Me.lstreference_cell)
    End Sub
    Private Sub lstreference_cell_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles lstreference_cell.MouseClick
        resetbck(Me.lstreference_cell)
        Try
            itm4 = Me.lstreference_cell.SelectedIndex.ToString
            If itm4 = "0" Then
                resetbck(Me.lstreference_cell)
                reference_cell = Me.lstreference_cell.SelectedItem.ToString
```

```
            ElseIf itm4 = "1" Then

                resetbck(Me.lstreference_cell)

                reference_cell = Me.lstreference_cell.SelectedItem.ToString

            End If

        Catch ex As System.NullReferenceException When
Me.lstreference_cell.SelectedIndex < 0

            result = emp_l(Me.lblrefence_cell)

            result = Windows.Forms.DialogResult.OK

            Me.lstreference_cell.Focus()

            reference_cell = String.Empty

            Me.lstreference_cell.ClearSelected()

        Catch ex As System.IndexOutOfRangeException When
Me.lstreference_cell.SelectedIndex > Me.lstreference_cell.Items.Count

            result = emp_l(Me.lblrefence_cell)

            result = Windows.Forms.DialogResult.OK

            Me.lstreference_cell.Focus()

            reference_cell = String.Empty

            Me.lstreference_cell.ClearSelected()

        End Try

    End Sub

    Private Sub lstreference_cell_MouseDoubleClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles lstreference_cell.MouseDoubleClick

        itm4 = String.Empty

        Me.lstreference_cell.ClearSelected()

        reference_cell = String.Empty

    End Sub

    Private Sub lstreference_cell_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstreference_cell.Validated

        inter.reference_cell = reference_cell

    End Sub

    Private Sub lstreference_cell_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles lstreference_cell.Validating

        If reference_cell = String.Empty Then

            e.Cancel = True
```

```
            result = emp_l(Me.lblrefence_cell)

            result = Windows.Forms.DialogResult.OK

            Me.lstreference_cell.Focus()

            Me.lstreference_cell.ClearSelected()

        End If


    End Sub


#End Region

#Region "txtcorrosion_potential "


    Private Sub txtcorrosion_potential_Enter(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtcorrosion_potential.Enter

        get_focus_effect(Me.txtcorrosion_potential)

    End Sub

    Private Sub txtcorrosion_potential_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtcorrosion_potential.Leave

        resetbck(Me.txtcorrosion_potential)

    End Sub

    Private Sub txtcorrosion_potential_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles txtcorrosion_potential.MouseClick

        resetbck(Me.txtcorrosion_potential)

    End Sub

    Private Sub txtcorrosion_potential_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtcorrosion_potential.Validated

        inter.corrosion_potential = corrosion_potential

    End Sub

    Private Sub txtcorrosion_potential_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles txtcorrosion_potential.Validating

        Dim ddi As Double

        Try

            ddi = CDbl(Me.txtcorrosion_potential.Text)

            If Not IsNumeric(ddi) OrElse ddi = 0 Then

                result = imprp(Me.txtcorrosion_potential)
```

```
                result = Windows.Forms.DialogResult.OK
                Me.txtcorrosion_potential.Focus()
                e.Cancel = True
            ElseIf ddi <= -1000 OrElse ddi >= 1000 Then
                result = caus(Me.txtcorrosion_potential)
                If result = Windows.Forms.DialogResult.OK Then
                    corrosion_potential = (ddi)
                Else
                    e.Cancel = True
                    Me.txtcorrosion_potential.Focus()
                End If
            Else
                corrosion_potential = (ddi)
            End If
        Catch ex As System.Exception
            result = imprp(Me.lblcorrosion_potential)
            result = Windows.Forms.DialogResult.OK
            e.Cancel = True
        End Try
    End Sub
#End Region
#Region "txtresistivity "


    Private Sub txtresistivity_Enter(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtresistivity.Enter
        get_focus_effect(Me.txtresistivity)
    End Sub
    Private Sub txtresistivity_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtresistivity.Leave
        resetbck(Me.txtresistivity)
    End Sub
    Private Sub txtresistivity_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles txtresistivity.MouseClick
```

```
        resetbck(Me.txtresistivity)

    End Sub

    Private Sub txtresistivity_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtresistivity.Validated

        inter.structure_resistivity = resistivity

    End Sub

    Private Sub txtresistivity_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles txtresistivity.Validating

        Dim ddi As Double

        Try

            ddi = CDbl(Me.txtresistivity.Text)

            If Not IsNumeric(ddi) OrElse ddi <= 0 Then

                result = imprp(Me.txtresistivity)

                result = Windows.Forms.DialogResult.OK

                Me.txtresistivity.Focus()

                e.Cancel = True

            Else

                resistivity = (ddi)

            End If

        Catch ex As System.Exception

            result = imprp(Me.txtresistivity)

            result = Windows.Forms.DialogResult.OK

            e.Cancel = True

        End Try

    End Sub

#End Region

#Region " lstcoating_type "

    'Private Sub lstcoating_type_GotFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstcoating_type.GotFocus



    'End Sub

    'Private Sub lstcoating_type_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstcoating_type.LostFocus
```

```
    'End Sub

    'Private Sub lstcoating_type_SelectedIndexChanged(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstcoating_type.SelectedIndexChanged

    '    'itm3 = Me.lstcoating_type.SelectedItem.ToString

    '    'If itm3 = "Blasted White" Then

    '    '    resetbck(Me.lstcoating_type)

    '    '    coating_type = Me.lstcoating_type.SelectedItem.ToString

    '    'ElseIf itm3 = "Coal Tar Enamel" Then

    '    '    resetbck(Me.lstcoating_type)

    '    '    coating_type = Me.lstcoating_type.SelectedItem.ToString

    '    'ElseIf itm3 = "Extruded Polyolefin / Polyethelyene / Polypropylene - Adhesive"
Then

    '    '    resetbck(Me.lstcoating_type)

    '    '    coating_type = Me.lstcoating_type.SelectedItem.ToString

    '    'ElseIf itm3 = "Extruded Polyolefin / Polyethelyene / Polypropylene - Epoxy,
Adhesive" Then

    '    '    resetbck(Me.lstcoating_type)

    '    '    coating_type = Me.lstcoating_type.SelectedItem.ToString

    '    'ElseIf itm3 = "Fusion Bonded Epoxy" Then

    '    '    resetbck(Me.lstcoating_type)

    '    '    coating_type = Me.lstcoating_type.SelectedItem.ToString

    '    'ElseIf itm3 = "Polymer Tape Wraps" Then

    '    '    resetbck(Me.lstcoating_type)

    '    '    coating_type = Me.lstcoating_type.SelectedItem.ToString

    '    'End If

    'End Sub

    'Private Sub lstcoating_type_SelectedValueChanged(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstcoating_type.SelectedValueChanged

    '    'If itm3 = String.Empty Then

    '    '    result = emp_l(Me.lblcoating_type)

    '    '    result = Windows.Forms.DialogResult.OK

    '    'Else
```

```
    '    '    coating_type = itm3

    '    'End If

    'End Sub

    Private Sub lstcoating_type_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstcoating_type.Leave

        resetbck(Me.lstcoating_type)

    End Sub

    Private Sub lstcoating_type_Enter(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstcoating_type.Enter

        get_focus_effect(Me.lstcoating_type)

    End Sub

    Private Sub lstcoating_type_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles lstcoating_type.MouseClick

        resetbck(Me.lstcoating_type)

        Try

            itm3 = Me.lstcoating_type.SelectedIndex.ToString

            If itm3 = "0" Then

                resetbck(Me.lstcoating_type)

                coating_type = Me.lstcoating_type.SelectedItem.ToString

            ElseIf itm3 = "1" Then

                resetbck(Me.lstcoating_type)

                coating_type = Me.lstcoating_type.SelectedItem.ToString

            ElseIf itm3 = "2" Then

                resetbck(Me.lstcoating_type)

                coating_type = Me.lstcoating_type.SelectedItem.ToString

            ElseIf itm3 = "3" Then

                resetbck(Me.lstcoating_type)

                coating_type = Me.lstcoating_type.SelectedItem.ToString

            ElseIf itm3 = "4" Then

                resetbck(Me.lstcoating_type)

                coating_type = Me.lstcoating_type.SelectedItem.ToString

            ElseIf itm3 = "5" Then

                resetbck(Me.lstcoating_type)
```

```
                coating_type = Me.lstcoating_type.SelectedItem.ToString

            End If

        Catch ex As System.NullReferenceException When Me.lstcoating_type.SelectedIndex
< 0

            result = emp_l(Me.lblcoating_type)

            result = Windows.Forms.DialogResult.OK

            Me.lstcoating_type.Focus()

            coating_type = String.Empty

        Catch ex As System.IndexOutOfRangeException When
Me.lstcoating_type.SelectedIndex > Me.lstcoating_type.Items.Count

            result = emp_l(Me.lblcoating_type)

            result = Windows.Forms.DialogResult.OK

            Me.lstcoating_type.Focus()

            coating_type = String.Empty

        End Try

    End Sub

    Private Sub lstcoating_type_MouseDoubleClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles lstcoating_type.MouseDoubleClick

        itm3 = String.Empty

        Me.lstcoating_type.ClearSelected()

        coating_type = String.Empty

    End Sub

    Private Sub lstcoating_type_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles lstcoating_type.Validated

        inter.coating_type = coating_type

    End Sub

    Private Sub lstcoating_type_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles lstcoating_type.Validating

        If coating_type = String.Empty Then

            e.Cancel = True

            result = emp_l(Me.lblcoating_type)

            result = Windows.Forms.DialogResult.OK

            Me.lstcoating_type.Focus()

        End If
```

```
    End Sub

#End Region

#Region "txtcoating_resistance"

    Private Sub txtcoating_resistance_Enter(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtcoating_resistance.Enter

        get_focus_effect(Me.txtcoating_resistance)

    End Sub

    Private Sub txtcoating_resistance_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtcoating_resistance.Leave

        resetbck(Me.txtcoating_resistance)

    End Sub

    Private Sub txtcoating_resistance_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles txtcoating_resistance.MouseClick

        resetbck(Me.txtcoating_resistance)

    End Sub

    Private Sub txtcoating_resistance_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtcoating_resistance.Validated

        inter.coating_resistance_sqrm = coating_resistance

    End Sub

    Private Sub txtcoating_resistance_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles txtcoating_resistance.Validating

        Dim ddi As Double

        Try

            ddi = CType(Me.txtcoating_resistance.Text, Double)

            If Not IsNumeric(ddi) OrElse ddi <= 0 Then

                result = imprp(Me.lblcoating_resistance)

                result = Windows.Forms.DialogResult.OK

                Me.txtcoating_resistance.Focus()

                e.Cancel = True

            ElseIf ddi >= 10000 Then

                result = caus(Me.lblcoating_resistance)

                If result = Windows.Forms.DialogResult.OK Then

                    coating_resistance = ddi

                Else
```

```
                    e.Cancel = True
                    Me.txtcoating_resistance.Focus()
                End If
            Else
                coating_resistance = ddi
            End If
        Catch ex As System.Exception
            e.Cancel = True
            result = imprp(Me.lblcoating_resistance)
            result = Windows.Forms.DialogResult.OK
        End Try
    End Sub
#End Region
#Region " txtinitial_cbf "

    Private Sub txtinitial_cbf_GotFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtinitial_cbf.GotFocus
        get_focus_effect(Me.txtinitial_cbf)
    End Sub
    Private Sub txtinitial_cbf_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtinitial_cbf.LostFocus
        resetbck(Me.txtinitial_cbf)
    End Sub
    Private Sub txtinitial_cbf_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles txtinitial_cbf.MouseClick
        resetbck(Me.txtinitial_cbf)
    End Sub
    Private Sub txtinitial_cbf_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtinitial_cbf.Validated
        inter.initial_coating_factor = initial_coating_factor
    End Sub
    Private Sub txtinitial_cbf_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles txtinitial_cbf.Validating
        Dim ddi As Double
```

```
        Try
            ddi = CDbl(Me.txtinitial_cbf.Text)
            If Not IsNumeric(ddi) OrElse ddi < 0 Then
                e.Cancel = True
                result = imprp(Me.lblinitial_cbf)
                result = Windows.Forms.DialogResult.OK
                Me.txtinitial_cbf.Focus()
            ElseIf ddi > 100 Then
                e.Cancel = True
                result = imprp(Me.lblinitial_cbf)
                result = Windows.Forms.DialogResult.OK
                Me.txtinitial_cbf.Focus()
            Else
                initial_coating_factor = (ddi)
            End If
        Catch ex As System.Exception
            result = imprp(Me.lblinitial_cbf)
            result = Windows.Forms.DialogResult.OK
            e.Cancel = True
        End Try
    End Sub
#End Region
#Region "txtfinal_cbf "
    Private Sub txtfinal_cbf_GotFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtfinal_cbf.GotFocus
        get_focus_effect(Me.txtfinal_cbf)
    End Sub
    Private Sub txtfinal_cbf_Leave(ByVal sender As Object, ByVal e As System.EventArgs)
Handles txtfinal_cbf.Leave
    End Sub
    Private Sub txtfinal_cbf_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtfinal_cbf.LostFocus
        resetbck(Me.txtfinal_cbf)
```

```
    End Sub

    Private Sub txtfinal_cbf_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles txtfinal_cbf.MouseClick

        resetbck(Me.txtfinal_cbf)

    End Sub

    Private Sub txtfinal_cbf_Validated(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtfinal_cbf.Validated

        inter.final_coating_factor = final_coating_factor

    End Sub

    Private Sub txtfinal_cbf_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles txtfinal_cbf.Validating

        Dim ddi As Double

        Try

            ddi = CDbl(Me.txtfinal_cbf.Text)

            If Not IsNumeric(ddi) OrElse ddi < 0 Then

                result = imprp(Me.lblfinal_cbf)

                result = Windows.Forms.DialogResult.OK

                Me.txtfinal_cbf.Focus()

                e.Cancel = True

            ElseIf ddi > 100 Then

                result = imprp(Me.lblfinal_cbf)

                result = Windows.Forms.DialogResult.OK

                Me.txtfinal_cbf.Focus()

                e.Cancel = True

            Else

                final_coating_factor = (ddi)

            End If

        Catch ex As System.Exception

            result = imprp(Me.lblfinal_cbf)

            result = Windows.Forms.DialogResult.OK

            e.Cancel = True

        End Try

    End Sub
```

```
#End Region

#End Region

#Region "buttons control events"

    Private Sub bttnback_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles bttnback.MouseClick

        startform.Show()

        Me.Hide()

    End Sub

    Private Sub bttncancel_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles bttncancel.MouseClick

        result = errt()

        If Not (result = Windows.Forms.DialogResult.Yes) Then

            Exit Sub

        Else

            set_2_null()

            Me.Refresh()

            Me.lststructure_shape.Focus()

            get_focus_effect(lststructure_shape)

        End If


    End Sub

    Private Sub bttncontinue_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles bttncontinue.MouseClick

        If super() = True Then

            Me.Hide()

            envronform.Show()

        End If

    End Sub

#End Region

#Region "linitialization and closing  events"

    Protected Overrides Sub OnClosing(ByVal e As System.ComponentModel.CancelEventArgs)

        MyBase.OnClosing(e)

        e.Cancel = True
```

```
        Me.Hide()

        main_menufrm.MenuStrip.Enabled = True

        main_menufrm.ToolStrip.Enabled = True

    End Sub

    Private Sub structurefrm_FormClosing(ByVal sender As Object, ByVal e As
System.Windows.Forms.FormClosingEventArgs) Handles Me.FormClosing




    End Sub

    Private Sub structurefrm_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

        module1.structureform = Me

        Me.txtin_radius.Enabled = False

    End Sub

#End Region

#Region "utility"

    Private Sub resetbck(ByVal e As Control)

        e.ResetBackColor()

    End Sub

    Private Function super() As Boolean

        If shape = String.Empty Then

            result = emp_t(Me.lblstructure_shape)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.lststructure_shape.Focus()

        ElseIf length = 0 OrElse Not IsNumeric(length) Then

            result = emp_t(Me.lblstrlength)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.txtlength.Focus()

        ElseIf outer_radius = 0 OrElse Not IsNumeric(outer_radius) Then

            result = emp_t(Me.lblut_radius)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)
```

```
            Me.txtut_radius.Focus()

        ElseIf inner_radius = 0 Then

            result = ovsight(Me.lblin_radius)

            result = CType(Windows.Forms.DialogResult.No = 7,
Windows.Forms.DialogResult)

            Me.txtin_radius.Focus()

        ElseIf inner_radius < 0 OrElse Not IsNumeric(inner_radius) Then

            result = imprp(Me.lblin_radius)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.txtin_radius.Focus()

        ElseIf resistivity = 0 OrElse Not IsNumeric(resistivity) Then

            result = imprp(Me.lblresistivity)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.txtresistivity.Focus()

        ElseIf coating_resistance <= 0 OrElse Not IsNumeric(coating_resistance) Then

            result = imprp(Me.lblcoating_resistance)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.txtcoating_resistance.Focus()

        ElseIf material_type = String.Empty Then

            result = emp_l(Me.lblmaterial_type)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.lstmaterial_type.Focus()

        ElseIf material = String.Empty Then

            result = emp_t(Me.lblmaterial)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.txtmaterial.Focus()

        ElseIf material_specification = String.Empty Then

            result = emp_l(Me.lblmaterial_specification)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)
```

```
            Me.txtmaterial_specification.Focus()

        ElseIf coating_type = String.Empty Then

            result = emp_l(Me.lblcoating_type)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.lstcoating_type.Focus()

        ElseIf corrosion_potential = 0 OrElse Not IsNumeric(corrosion_potential) Then

            result = emp_l(Me.lblcorrosion_potential)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.txtcorrosion_potential.Focus()

        ElseIf reference_cell = String.Empty Then

            result = emp_t(Me.lblrefence_cell)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.lstreference_cell.Focus()

            Me.lstcoating_type.Focus()

        ElseIf initial_coating_factor = 0 OrElse Not IsNumeric(initial_coating_factor)
Then

            result = emp_t(Me.lblinitial_cbf)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.txtinitial_cbf.Focus()

        ElseIf final_coating_factor = 0 OrElse Not IsNumeric(density) Then

            result = emp_t(Me.lblfinal_cbf)

            result = CType(Windows.Forms.DialogResult.OK = 1,
Windows.Forms.DialogResult)

            Me.txtfinal_cbf.Focus()

        Else

            Return True

        End If

    End Function

    Public Sub set_2_null()

        lststructure_shape.ClearSelected()
```

```
resetbck(Me.lststructure_shape)
shape = String.Empty
txtlength.Text = ""
resetbck(Me.txtlength)
length = 0.0
txtut_radius.Text = ""
resetbck(Me.txtut_radius)
outer_radius = 0.0
txtin_radius.Text = ""
resetbck(Me.txtin_radius)
inner_radius = 0.0
lstmaterial_type.ClearSelected()
resetbck(Me.lstmaterial_type)
material_type = String.Empty
txtmaterial.Text = ""
resetbck(Me.txtmaterial)
material = String.Empty
txtmaterial_specification.Text = ""
resetbck(Me.txtmaterial_specification)
material_specification = String.Empty
lstcoating_type.ClearSelected()
resetbck(Me.lstcoating_type)
coating_type = String.Empty
txtcorrosion_potential.Text = ""
resetbck(Me.txtcorrosion_potential)
corrosion_potential = 0.0
txtin_radius.Text = ""
resetbck(Me.txtin_radius)
inner_radius = 0.0
lstreference_cell.ClearSelected()
resetbck(Me.lstreference_cell)
```

```
        reference_cell = String.Empty
        txtinitial_cbf.Text = ""
        resetbck(Me.txtinitial_cbf)
        initial_coating_factor = 0.0
        txtresistivity.Text = ""
        resetbck(Me.txtresistivity)
        resistivity = 0.0
        Me.txtcoating_resistance.Text = ""
        resetbck(Me.txtcoating_resistance)
        coating_resistance = 0.0
        txtfinal_cbf.Text = ""
        resetbck(Me.txtfinal_cbf)
        final_coating_factor = 0.0
    End Sub
    Private Sub get_focus_effect(ByVal e As Windows.Forms.Control)
        If e.Focus Then
            e.BackColor = Color.MediumSpringGreen
        End If
    End Sub
#End Region
End Class
```

REFERENCES

1. **Pierre R. Roberge, "***Handbook of Corrosion Engineering***"** New York: McGraw-Hill, (1999). 30 – 70.

2. **OVRI, J. E., "***Vandalization of Oil Pipelines and the Corrosion Factor***".** *Journal Corrosion Science & Technology* Special edition 1.1 (2004) 26 - 27.

3. **Talbot D., *et al,* "***Corrosion Science and Technology***".** CRC, Florida (1998) 20 – 33.

4. **Ijomah, M. N. C., "***Elements of Corrosion and Corrosion Theory***".** Auto-century Publishing Company Ltd, Enugu (1991) 15 – 26.

5. **Material Section, Exxon Mobil Upstream Research Company, "***Materials and Corrosion Overview***".** Exxon Mobil Upstream Research Company, Houston (1998).

6. **Nyborg, R., "***Corrosion Control in Oil and Gas Pipeline***"** The Oil and Gas Review Vol. 2 (2003) 7 - 18.

7. **Fairhurst, Dr. D., "***Offshore cathodic protection. What we have learnt?* **".** JSCE Vol. 8, Manchester (2005) 30 - 33.

8. **U.S. Army Corps of Engineers, "***Electrical Design, Cathodic Protection***".** United States Army Corp, Washington DC (1985).

9. **Norsork Standard, "***Common requirement in Cathodic Protection***".** NORSORK (1997)

10. **Marshall E. Parker, et al, "***Pipeline Corrosion and Cathodic Protection***".** Gulf Publishing Company, Houston (1999) 20 - 25.

11. **Bushman, James B., "***Galvanic Anode Cathodic Protection System Design.* Medina**".** Bushman and Associates, Inc., Ohio (2000) 10 - 15.

12. **U.S. Army Corps of Engineers, "***Cathodic Protection Anode Selection***".** Public Works Technical Bulletin, DEPARTMENT OF THE ARMY, U.S. Army Corps of Engineers, Washington, DC, (2001) 37 - 49.

13. **Meillier, A., "***A Review of Galvanic Anode Cathodic Protection Design***".** JSCE Vol. 8, Manchester (2004) 4 - 10.

14. **E Santana Diaz, R Adey, "***A Computational Environment for the Optimisation of CP system Performance and Signatures***".** Computational Mechanics BEASY, Southampton (2003). 12 – 20.

15. **Lysogorski, D. K. et al, "***A Comprehensive Review of The Slope Parameter Based Approach To Cathodic Protection Design And Analysis***".** JSCE Vol. 8, Manchester (2003) 10 - 23.

16. **Townley, D.W. "***Unified Design Equation for Offshore Cathodic Protection***".** NACE, HOUSTON (1997) 32 - 45.

17. **Grant Gibson, et al "***Novel cathodic Protection of Subsea Flowline and Risers***".** JSCE Vol. 4., Manchester (2003) 23 – 45.

18. **Recommended Practice DNV - RP-B40***,* **"***Cathodic Protection Design***".** Det Norske Veritas, Hovik (2005).

19. **Det Norske Veritas,** *Recommended Practice DNV-RP- B401:Cathodic Protection Design.* Hovik : Det Norske Veritas, (2005).

20. **Googan, C.G., "***Cathodic misconceptions***".** Anticorrosion Engineering Limited, Shrewsbury (2003). 23 – 56.

21. **G. K. Glass, A. M. Hassanein,** *"Suprisingly Effective Cathodic Protection.* Manchester" JSCE Vol. 4., Manchester (2003) 23 – 35.

22. . **Robinson, C. B et al , "***Optimum Cathodic Protection Potentials for High Strength Steels in Sea Water***".** JSCE, Manchester (2005).

23. **T. Sydberger, et al, "***Use of Convservatism in Cathodic Protection Design***".** NACE International PAPER 550, Houston (2001).

24. **DeGiorgi, V. G, "***Corrosion Basics and Computer Modelling***"**. JSCE Vol. 8., Manchester (2005) 8 – 12.

25. **Paul, S., "***Modelling and Computer Simulation of Cathodic Protection of Steel Structure***"**, JSCE Vol. 7., Manchester (2005) 10 – 29.

26. **Santa-Diaz E. et al, "***Predictive Modelling of Corrosion and Cathodic Protection System***".** Computational Mechanics BEASY, Southampton (2005). 23 -40.

27. **Fagan, B., et al, "***Mathematical Studies on Galvanic Corrosion***".**, Journal of the Electrochemical Society, New York (1956) 90 -103.

28. **P. Doig & P. Flewitt, "***A Finite Difference Numerical Analysis of Galvanic Corrosion for Seim-Infinite Linear Coplanar Electrodes***".** Journal of Electrochemical Society Vol. 126., New York (1979) 120 -122.

29. **R. A. Adey, C. A.Bebbia & S.M.Niku, "***Application of Boundary Elemenets in Corrosion Engineering***".** JCSE Vol. 8., Manchester (2003), 20 – 34.

30. **K. Amaya, J. Togashi, & S. Aoki, "***Inverse Analysis of Galvanic Corrosion - Using Fuzzy a-Priori Information***".** JSME Vol. 38 International Journal Series A-Mechanics and Material Engineering, New York 1995, 541–546.

31. . **J. Telles, L. Wrobel, W. Mansur, & J. Azevedo, "***Boundary Elements for Cathodic Protection Problems -Boundary Elements VII***",** Springer-Verlag, New York (1985) 73–83.

32. **K. J. Kennelley, L. Bone, and M. E. Orazem, "***Current and Potential Distribution on a Coated Pipeline with Holidays: Model and Experimental Verification***".**, Corrosion, Houston (1993) 211–219.

33. **D. Riemer, & M. Orazem, "***Cathodic Protection of Multiple Pipelines with Coating Holidays in Procedings of the NACE '99 Topical Research Symposium:Cathodic Protection***". *NACE,*** Houston (1999) 123 -150.

34. . **F. Brichau, J. Deconinck, & T. Driesens. "***Modeling of Underground Cathodic***"**, Corrosion Vol. 52, Houston (1996) 480–488.

35. **Townley, D. W.**, **"***Boundary Elemenet Modeling of Galvanic Anode Cathodic Protection Using The Design Slope Method***".** JSCE Vol. 9., Manchester (2003) *20* – 23.

36. **Force Technology, "***Seacorr - Cathodic Protection Modelling and Design***".** Force Technology Norway AS, Harald Osvoll (2008).

37. **Townley, D. W., "***Boundary Element of Modelling of Galvanic Anode Cathodic Protection Using The Slope Method***".** JSCE Vol. 7., Manchester (2003), 19 – 22.

38. . **S. Aoki, K. Amaya, A. Nakayama, and A. Nishikawa, "***Elimination of Error from Nonuniform Current Distribution in Polarization Measurement by Boundary Element Inverse AnalysisBoundary Element Inverse Analysis*". Corrosion Vol. 54., Houston (1998) 259–264.

39. **Qiu, Chenchen, "***Model for Interpretation of Pipeline Survey Data***".** Ph.D Dissertation, University of Florida, Gainesville (2003) 23 -78.

40. **Dominguez J., & Bribbia C., "***Boundary Element An Introductory Course***".** McGraw-Hill Book Company, New York (1988) 23 -50.

41. **Ernest W. Klenchla, Jr.,P.E., "**Coating for Corrosion Protection**".** [Online] Colorado School of Mines,[Cited: April 14, 2004.] ww.colorado.edu. (2002).

42. **Crundel, Bob,** "*Future of Sacrificial Anodes".* JSCE Vol. 8., Manchester (2001), 7 – 13.

43. **Lysogorski, W.H. Hartt and D.K, "***A FIrst - Principle Base Approach To Potential Attenuation Projection For Marine Pipelines And Risers***".** JSCE, Manchester (2004), 12 – 25.

44. **Britton, Jim, "**External Corrosion And Inspection Of Deep Water Pipelines**"**. [Online] Deepwater Corrosion Services Inc,[Cited: May 16, 2008.] www.stoprust.com. (2005).

45. **William Hart and Chu Wei, "***Design of Cathodic Protection System for Deep Water Compliant Petroleum Production Riser***".** United States Department of the Interior, Mineral Management Service, Herndon, Virginia (2005) 20 - 50.

46. **M. E. Orazem, J. M. Esteban, K. J. Kennelley, and R. M. Degerstedt, "***Mathematical Models for Cathodic Protection of an Underground Pipeline with Coating Holidays: Case Studies of Parallel Anode CP Systems***".** Corrosion Vol. 53., Houston (1997) 427– 436.

47. **V. G. Degiorgi, A. Kee, E. D Thomas, "***Characterization Accuracy in Modelling of Corrosion System***"***s.*: Mechanics of Material Branch, Naval Reasearch Laboratory, Washington DC, USA (2004) 10 - 20.

48. **Per Olav Gartland, Frode Bjoernaas & Harald Osvoll, "***Computer Modelling of Offshore CP System For 15 Years: What Have We Learned?***",** Trondheim, Norway : Material Performance, Houston (1995) 2 – 25.

49. **Turban, E., "***Decision Support and Expert Systems: Management Support Systems***".** Macmillan, New York (1990) 10 – 20.

50. **Nobar, P. M., Crilly, A. J., and Lynkaran, K., "***The Increasing Influence of Computers in Engineering Education: Teaching Vibration via Multimedia Programs***".** Engineering Education, New York (1996) 12 – 20.

51. **Jadot, A., and Lanclus, L., "***ESCORT: Expert Software for Corrosion Technology***".** Leuven University, Leuven (1985) 12 - 20.

52. **Durkin, J., "***Expert Systems: Design and Development***".** Macmillan New York , (1994) 23 -34.

53. **Materials Technology Institute, "***Report of Task Group 1 of the Working Party on Expert Systems in Materials Engineering***".** Materials Technology Institute, St. Louis (1990) 19 – 22.