2007

SOFTWARE ENGINEERING

Bassam Abdullah Alkebsi

Saba University

5/3/2007

هندسة البرامجيات(SOFTWARE ENGINEERING (SE

☺ المحاضرة الاولى ☺

مقدمة عن هندسة البرمجيات :

في الاربعينات والخمسينات ظهرت ما يسمى البرمجيات او Software ومع التدرج بدأت تظهر الانظمة التي تقوم بترتيب وتنظيم اعمال الالة ليلبي احتياجات المستخدم والمقصود بها هي انظمة التشغيل ، حيث كان الانسان - إن صح القول - إذا اراد ان يستخدم أي نظام فكان يجب عليه أن يخاطب كل جزء وليس كما هو اليوم حيث اصبح اليوم المستخدم يتعامل مع البرامج ذات المستوى العالي High Level Language .

ومع تطور الحواسيب بدأت تظهر لغات البرمجة المختلفة وانظمة التشغيل البسيطة واستمر هذا التطور في وتيرة متسارعة حتى وصلنا إلى أن ظهرت مشكلة جديدة وهي ما نسميه ازمة البرامجيات (Software Crisis) وهي انخفاض سعر المعدات Hardware مقابل ارتفاع البرامجيات بل قد يصل الحال إلى أن يصير سعر البرامجيات اكبر من سعر المعدات .

ومن الاسباب التي ادت إلى ارتفاع سعر ألـ Software وبالتالي ضهور هذه الازمة التالي :

1- الاحتفاض بالحقوق الفكرية للبرامج قبل المبرمجين أو الشركات العاملة في هذا المجال .

2- السعر المرتفع الذي يضعه المبرمج لنفسه .

3- الكم الهائل من البرامجيات الموجودة في سوق العمل والتي لا يحكمها قانون .

4- عدم وجود مهندسي البرامجيات (Software Engineer) أو ندرتهم في سوق العمل .

مواصفات ألـ Software Engineer :

1- شهادة متخصصة في مجال الحاسوب(Computer Sciences ,Information Technology) أو أي تخصص حاسوبي .

2- خبرة لا تقل عن سنة في مجال التخصص(انشاء وتصميم الانظمة الحاسوبية).

3- التحديث المستمر لمعلوماتك والمتابعة الدائمة لكل جديد .

الفرق بين ألـ Software وألـ Computer Programs :

لعلنا ذكرنا سابقا البرامجيات و ألـ Software بدون أي تمييز الا أن هناك فرق رئيسي وهو أن ألـ Software هي برامج جاهزة للاستخدام وموثقة والاعتمادية فيها تكون عالية ، أما بالنسبة للبرامجيات فهي برامج تكون معدة من المبرمج ولكنها ليست نهائية أي انها برامج مبدئية ويكون الهدف فيها هو المبرمج (لا يستطيع التعامل معها الا المنتج لها) ، مع العلم ان مرحلة ألـ Computers Programs او البرامجيات تكون مرحلية ومن ثم تتحول الى Software فور الانتهاء منها .

ويمكننا الان ان نعرف الـ Software انها مجموعة من الانظمة البرمجية المستقلة والمترابطة مع بعضها البعض وموثقة .

والمقصود هنا بالمستقلة : أي لكل برنامج كيانه الخاص .

واما المقصود بالمترابطة : أي أن بينها قنوات ربط .

ماهو ألـ Software Engineering (SE) :

يمكننا أن نعرف هندسة البرمجيات بانها : فرع من فروع المعرفة والتي تهتم بانتاج وصناعة وتطوير ألـ Software (البرامجيات) بشرط أن تكون الجودة عالية واقل تكلفة وتسلم في الوقت المناسب .

أي أن هذه المادة يمكننا من خلالها معرفة من أن النظام ذات جودة عالية أو منخفظة وبالتالي يمكن أن يحدد السعر لها وكذلك الالتزام بمواعيد التسليم وكيفية تحديد الفترة الزمنية لذلك وفق معايير معينة .

معلومات عن بحث المادة :

المحاور الاساسية للبحوث :

1- المقارنة بين لغتين برمجيتين أو اكثر من لغة برمجية مثل لغتي JAVA و C# أو C++ ن

أو بين عدة تطبيقات لغوية لقواعد البيانات مثل Oracle و SQL server .

2- ألـ Compilers وماهي مكوناته .

3- تصميم نظام برمجي وفق معايير المادة .

الواجب الاول:

صندوق النص الذكي:

يقوم هذا الصندوق بايجاد ناتج العمليات الحسالبية بمجرد الكتابة عليه وعند الضغط على الزر = في لوحة المفاتيح يضهر الناتج بجانب المعادلة الرياضية ون استخدام أي من الادوات الا صندوق النص هذا

```
    Private Sub TextBox13_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles TextBox13.KeyPress
        Dim s As String = TextBox13.Text
        Dim a, b As Integer
        Dim c As String
        Try
            If e.KeyChar = "=" Then
               For x As Byte = 0 To TextBox13.Text.Length - 1
                    If s(x) = "+" Then
                        a = s.Substring(0, s.IndexOf("+"))
                        b = s.Substring(s.IndexOf("+") + 1, s.Length - s.IndexOf("+") -1)
                        c = a + b
                        TextBox13.Text += "=" + c
                        e.Handled = True
                    ElseIf s(x) = "-" Then
                        a = s.Substring(0, s.IndexOf("-"))
                        b = s.Substring(s.IndexOf("-")+1, s.Length _ - s.IndexOf("-") -1)
                        c = a - b
                        TextBox13.Text += "=" + c
```

```
                    e.Handled = True
                ElseIf s(x) = "*" Then
                    a = s.Substring(0, s.IndexOf("*"))
                    b = s.Substring(s.IndexOf("*")+1, s.Length _ - s.IndexOf("*") -1)
                    c = a * b
                    TextBox13.Text += "=" + c
                    e.Handled = True
                ElseIf s(x) = "/" Then
                    a = s.Substring(0, s.IndexOf("/"))
                    b = s.Substring(s.IndexOf("/")+1, s.Length _ - s.IndexOf("/") -1)
                    c = a / b
                    TextBox13.Text += "=" + c
                    e.Handled = True
                End If
            Next
        End If
    Catch ex As Exception
        MessageBox.Show("يجب عليك التاكد من ان القيم المدخلة هي ارقام", "هناك خطأ",
MessageBoxButtons.OK, MessageBoxIcon.Error)
    End Try
End Sub
```

☺ المحاضرة الثانية ☺

انواع البرامجيات :

هناك نوعان من البرامجيات حسب الاستخدام :

1- Generic Programs (البرامج العامة) وهي البرامجيات التي تكون منتجة لاغراض عامة ويمكن أن يستخدمها أي شخص أو أي شركة من الشركات مثل معالج النصوص Word وكامل البرامج المكتبية المنتجة من قبل شركة مايكروسوفت وغيرها من البرامج.

2- Bespoke Programs (البرامج الخاصة) وهي البرامج المعدة خصيصا للمستخدم Customized حيث تكون معدة حصب مايريد المبتثمر أو الشركة ألتي طلبت هذا البرنامج وعادة ما تكون ذات حجم صغير مقارنة مع البرامج العامة الاستخدام ومتعبة في نفس الوقت للمبرمج وتكون ايراداتها اقل من البرامج العامة.

من خلال ما سبق يمكننا أن نحدد النقاط ألتي يجب على المستخدم أن يقوم بمراعاتها قبل وبعد واثناء تصميمه لأحد هذا النوعين من البرامجيات .

1- الزمن Time : ونقصد بالزمن من ثلاث نقاط اساسية :

أ- تحديد موعد تسليم النظام .

ب- تحديد الفترة الزمنية للمبرمجين لانتاج هذا النظام .

ت- تحديد سرعة النظام أو ما يسمى استغلال موارد النظام .

2- الجودة Quality : ويمكننا أن ننظر إلى الجودة من ثلاث وجهات نظر :

أ- المالك Customer : بالنسبة إلى المالك فانه يهمه من ناحية الجودة التالي:

- تسليم النظام في الموعد المحدد .
- تحقيق الاعتمادية والامنية والامان في النظام Dependability & Reliability (Security & Safety) ونقصد بالاعتمادية تنفيذ الاعمال من دون اخطاء .
- الكفاءة (تنفيذ اكبر قدر ممكن من العمليات في اقصر وقت) .

- قابلية الصيانة Maintainability (المرونة الكافية للتعديل في العمليات أو اضافتها اوتغيير الصلاحيات والمستخدمين .....

ب- المستخدم User : ينظر المستخدم في جودة من النظام من حيث التالي :

- أن يكون مرن وسهل التعلم جيد التصميم .
- الاعتمادية .
- الكفاءة .

ث- المطور Developer Or Software Engineer :بالنسبة للجودة في نظر مهندس البرامجيات فهو ينظر لها من الاتجهات التالية :

- عامل الامن والامان يكون عاليا جدا .
- جودة التصميم الخارجي Design .
- الاعتمادية .
- الكفاءة .
- قابلية الصيانة .

من خلال ما سبق يمكننا الان أن نجمل كل ماسبق من النقاط في النقاط التالية وتكون تحت السؤال التالي:

<u>كيف يمكننا أن نحكم على النظام بانه نظام جيد أو غير جيد (المعايير ألتى يحدد جودة النظام)</u>

1- قابلية الصيانة .
2- الكفاءة .
3- الاعتمادية .
4- قابلية الاستخدام Usability ونقصد بها مدى امكانية تعلم النظام بسهولة .

وتكون الانظمة متأرجحة ما بين تلك المعايير بين صعود وهبوط وبالتالي على مدى امكانية توفي تلك المعايير تكون الانظمة افضل

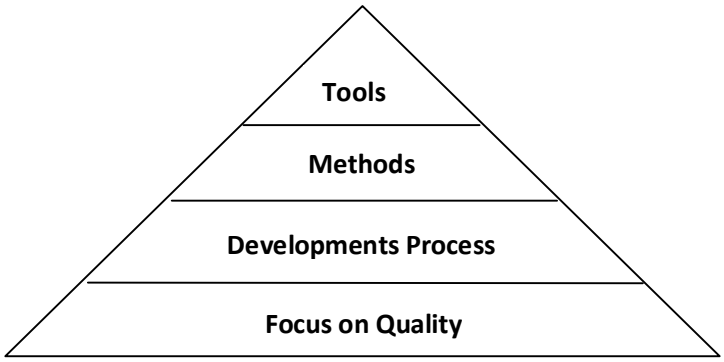
الواجب الثاني : تفحص من قبل النظام للمستخدمين إذا ادخل المستخدم الخاطئ فيقوم باعطاءه ثلاث فرص ومن ثم يقوم النظام باقفال نفسه أو يقوم باقفال نفسه عند مروم ثلاثين ثانية دون أن يدخل المستخدم رقم المستخدم واسم المستخدم .


```
Imports System.Data.oledb            'استيراد فضاء اسماء قواعد البيانات

Public Class Form1
    Dim timer As Byte = 30
    Dim i As Byte = 3
    Dim cnstring As String = "provider=Microsoft.Jet.OLEDB.4.0;data
source=c:\company2.mdb;"       'جملة الاتصال بالقاعدة
    Dim cn As New OleDbConnection(cnstring)      'فئة الاتصال
    Dim cmd As New OleDbCommand                  'فئة الاوامر

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim bool As Boolean = False
        cn.Open()
        cmd.Connection = cn
        cmd.CommandText = " SELECT * FROM USERS"       'من جدول المستخدمين
        Dim read As OleDbDataReader = cmd.ExecuteReader
        Do While read.Read
            If read("id") = TextBox1.Text AndAlso read("name") = TextBox2.Text Then
                main.Show()   'الانتقال إلى النافذة الرئيسية عندتحقق الشرط
                Me.Hide()
                bool = True
                Timer1.Enabled = False
            End If
        Loop
        If bool = False Then
```

```
            i -= 1
            If i = 0 Then
                MessageBox.Show("نفسه باغلاق النظام يقوم سوف محاولة اي لك يعد لم المعذرة",
"تنبيه !!", MessageBoxButtons.OK, MessageBoxIcon.Warning)
                Application.Exit()
            Else
                MessageBox.Show("يبقى لم صحيحة غير ادخلتها التي السر كلمة او المستخدم اسم ان
سوى لك " & i & " البرنامج يغلق سوف والا للدخول مرات ", "تنبيه !!", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                TextBox1.Clear()
                TextBox2.Clear()
            End If
        End If
        read.Close()
        cn.Close()
    End Sub

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Timer1.Tick
        If timer Mod 2 = 0 Then
            PictureBox1.Visible = True
            PictureBox2.Visible = False
        Else
            PictureBox2.Visible = True
            PictureBox1.Visible = False
        End If
        timer -= 1
        Label1.Text = " سوى لك يبقى لم " & timer & "  ثانية ويغلق البرنامج اوتوماتيكيا"
        If timer = 0 Then
            Application.Exit()
        End If
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        Application.Exit()
    End Sub
End Class
```

☺ المحاضرة الثالثة ☺

الطبقات التقنية لهندسة البرامجيات :



1- Tools: ونقصد بها قسمين رئيسيين :

أ- Software

ب- Hardware

اولا Software: يراد بذلك ماهي اللغة البرمجية والتي اسميناها هنا اصطلاحا بالادوات والتي سوف نستخدمها في إنشاء نظامنا بحيث تكون الاعتمادية على النظام أعلى .

و الجدول التالي يوضح مراحل تطور البرمجة واهم المشاكل ألتي كانت تواجه مهندس البرامجيات :

Process of Data	Example	Note	Display
By Line	Basic ,C Pascal	GoTo	المشاكل ألتي كانت تظهر لدى مهندسي البرامجيات هو عدم امكانية تتبع الاخطاء بسبب كثرة القفزات بألـ GoTo .
Procedure or Function	T C++,T B ,T Pascal	GoTo	وفي هذه المرحلة كانت لغات ألـ Turbo هي الوحيدة ألتي تحول البرنامج المصدري إلى برنامج تنفيذي أي من source إلى exe .
OOP(Object Oriented Programming)	V C++ ,V Basic ,.Net	GoTo	وفي هذه المرحلة اصبحت البرمجة برمجة الكائنات والتي تتميز بالوراثة وتعدد الواجهات وغيرها من المميزات ألتي اضيفت إلى تلك اللغة .
By Logic	Prolog	____	وهي احدث انواع البرمجة إلى الان وتتميز بان الشيفرة هي مكتوبة على اساس المخاطبة بينها وبين الانسان .

ثانيا ألـ Hardware : ونقصد بها هو الادوات ألتي سوف تستخدم للنظام من قطع ولوازم اخرى .

2- Methods:

وهي الطرق أو الخوارزميات المختلفة لانشاء وتصميم النظام أو البرنامج وكيفية تدفق البيانات من وجهة نطر المهندس .

3- Development Process :

وهي الكود البرمجي و اسميناها هنا بالتطوير لان المهندس يقوم بتطير النظام الحالي سواء اكان يدوي ام الي .

4- Focus on Quality :

حيث يجب على مهندس البرامجيات التركيز على الجودة في كل مما سبق بحيث أن كل طبقة من الطبقات السابقة مرتبطة و معتمدة على الاخرى .

المسؤلية الاخلاقية والاحترافية :

وهنا يرى انه يجب إنشاء هيئة للمعايير والجودة شبيهة بالهيئات المتخصصة في جودة الكهربائيات على سبيل المثال كالمنظمة الامريكية IEEE ، ومن الجدير بالذكر أن هناك بعض الجامعات في العالم وضعت قسم لمهندسي البرامجيات كقسم الاطباء عند بدءهم ممارستهم مهنتهم .

وبذلك يجب على مهندس البرامجيات الاهتمام بالنقاط التالية :

1- الخصوصية Confidentiality : خصوصية العملاء الذين نتعامل معهم حيث يجب علينا نحن المهندسين الحفاظ على اسرارهم ، فعند تصميم النظام يجب علينا مراعاة ذلك .
2- التخصصية Competence : لا تضع نفسك في مكان ليس مكانك فيجب عليك تحديد للعميل تخصصك .
3- الحفاظ على حقوق الملكية .
4- استخدام الحاسوب Computer Misuse :لا يستخدم الحاسوب الا في الشيئ الذي اعد من اجله .

سؤال : ما هي الاشياء ألتي تعمل على رفع الاعتمادية ؟ :

اولا الاعتمادية تحدثنا فيها مسبقا وهو مدى ثقة المنظمة بالنظام اما الاشياء ألتي تعمل على رفع الاعتمادية فهي :

1- Hardware Reliability: فلو كان لدينا نظام ذات جودة عالية لكن المعدات كانت رديئة فذلك يؤدي إلى ضعف الاعتمادية.

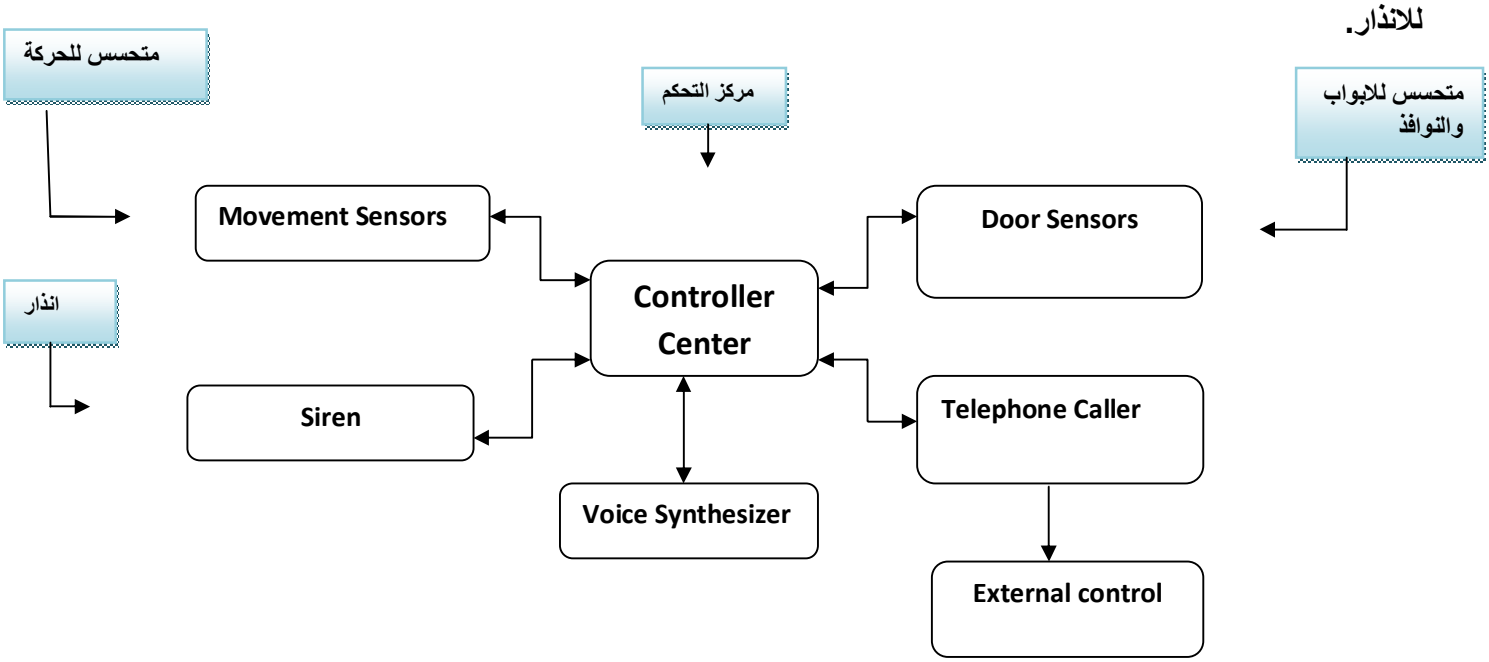
2- Software Reliability: وبالمثل لو أن المعدات عالية المستوى ومن ثم يكون النظام رديء أو أن تكون قاعدة البيانات الخاصة بالنظام ضعيفة الحماية فذلك يؤدي إلى ضعف الاعتمادية في النظام.

3- Operator Reliability: فلو كان المستخدم لا يجيد استخدام النظام فقد يقوم بعمل اشياء تضر بالمنظمة ككل.

الانظمة و بيئة العمل الخاصة بها :

يقصد بالبيئة بيئة العمل المحيطة بالنظام و كذلك البيئة الطبيعية فعندما تكون البيئة الخارجية أو الطبيعية تتاثر و توثر بالنطام كان يتاثر النظام لدرجة الحرارة العالية أو أن يتفاعل مع الضغط المتزايد في الغرفة مثلا فيقوم باجراء معين وهكذا .

الشكل التالي يوضح النظام الالي وكيف يتفاعل مع البيئة الخارجية ويتحسس لها كان يكون نظام بنكي يتحسس لاي فتح لاي باب أو نافذة خارج الدوام الرسمي وبالتالي يقوم باجراء معين كان يقوم بالاتصال بالشرطة أو بمدير البنك أو أن يقوم باطلاق جرس للانذار.



الواجب الثالث مقدار تفاعل النظام مع البيئة الخارجية بحيث لو ارتفعت مثلا درجة الحرارة عن خمسين يعطي اشارة او صوت بان درجة الحرارة ارتفعت عن الحد الطبيعي وسنمثل درجة الحرارة بارقام عشوائية :


```
Imports System.Media
Public Class Form1
    Dim snd As New SoundPlayer("c:\bassam ring1.wav")      'لتشغيل ملفات الصوت
    Event Hot()                                            'تعريف حدث
    Private Sub danger() Handles Me.Hot          'التقاط الحدث عند انطلاقه
        Timer1.Enabled = False
        snd.Load()                      'تحميل ملف الصوت
        snd.Play()                      'تشغيل جرس الانذار
        Panel1.Visible = True
        Label1.Visible = True           'ظهور رسالة التنبيه
        Label1.BackColor = Color.Red
        Me.Enabled = False
    End Sub
    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Timer1.Tick
        Dim rnd As New Random()
        Dim par As Byte = rnd.Next(0, 70)
        ProgressBar1.Value = par
        Me.Text = par
        If par > 50 Then
            RaiseEvent Hot()              ' اطلاق الحدث

        End If
    End Sub

End Class
```

☺ المحاضرة الرابعة ☺

سؤال : ما هي الاسئلة التي يجب على مهندس البرامجيات ان يسأل نفسه عند تصميمه للنظام ؟

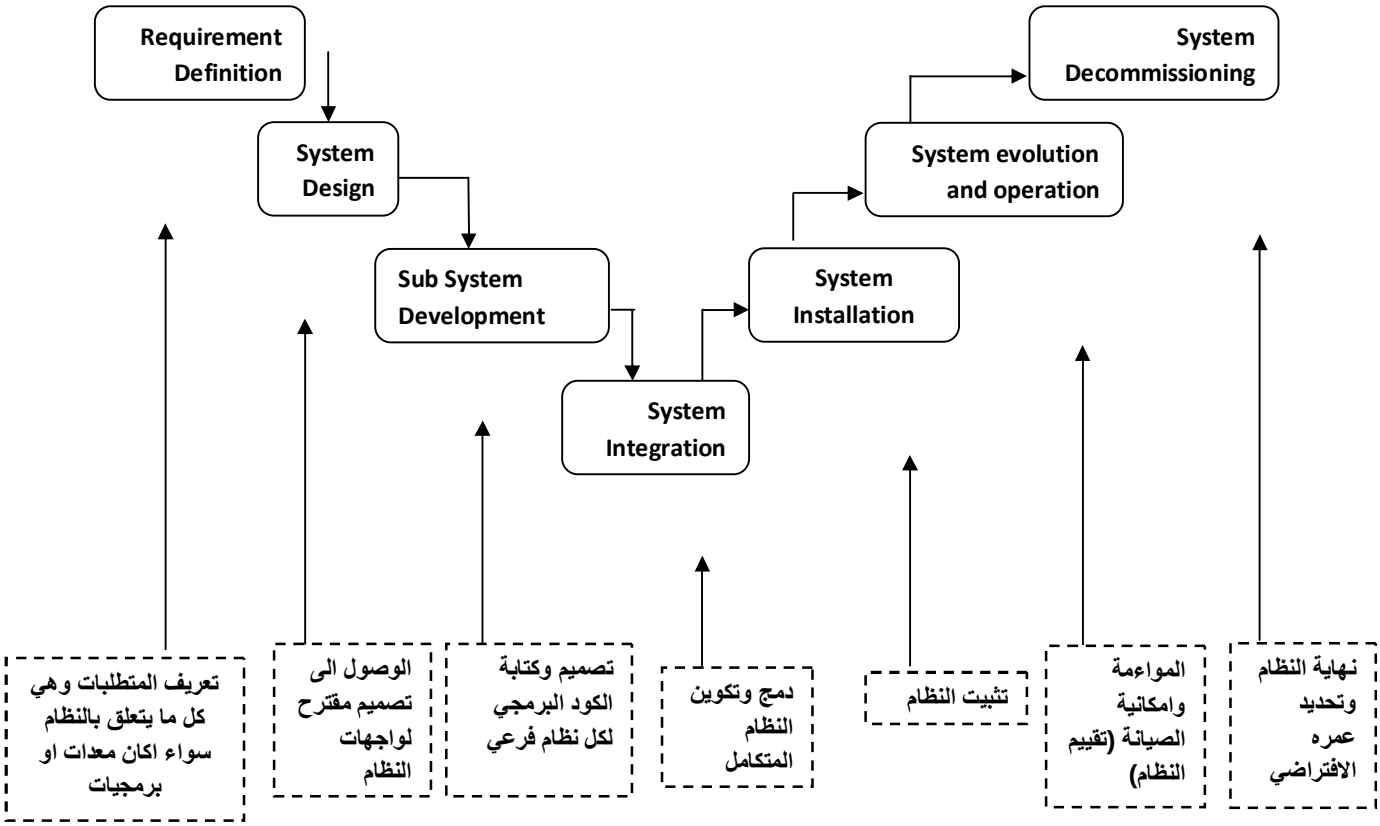
1- Process Change: أي هل النظام المقترح سيؤدي الى تغير في شكل معالجة العمليات فمثلا العمليات الحسابية في احد الشركات فهل نظامنا سوف يؤدي الى تحول تلك العمليات الى عمليات الية داخل الحاسوب ام لا ؟

2- Job Change : أي هل طبيعة العمل العمل للموظفين ستتغير بمعنى اخر هل سيبقى الموظفون في وظائفهم بحيث سيحتاجون الى اعادة تاهيل ام لا.

3- Organization Change: هل هذا النظام سيؤدي الى مايسمى بالتغيرات المنظمية أي هل ستتغير هيكلية المنظمة ام لا فعلى سبيل المثال ادارة الارشفة فهي مستقبلا ايلة الى الاختفاء مع ظهور انظمة قواعد البيانات ذات الامنية والكفائة العالية .

وبعد ان يسأل مهندس البرامجيات نفسه الاسئلة السابقة ينتقل الى "خطوات هندسة البرامجيات".

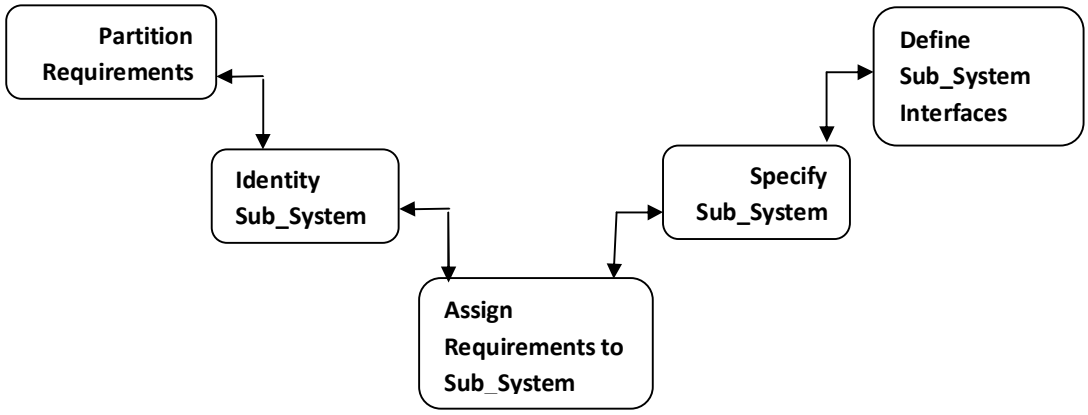
خطوات هندسة البرامجيات System Engineering Process: ويمكن توضيح تلك الخطوات في الشكل التالي



تصميم النظام System Design :

وتصميم النظام لا يعني تصميم الواجهات فقط وايضا هو تصميم للكود البرمجي والتقارير ...

ويمكن توضيح مراحل التصميم بالشكل التالي :



1- Partition Requirements: تقسيم المتطلبات وكذلك تحديد متطلبات كل قسم من اقسام النظام سواءا اكانت تلك المتطلبات هي معدات كالطابعات او كاميرات المراقبة او كانت برمجيات معينة فمثلا لو كان النظام كبير الحجم فان الشركة المصنعة له سوف تقسم النظام وتعطي كل فريق عمل من المبرمجين او المحللين جزء من النظام الكلي.

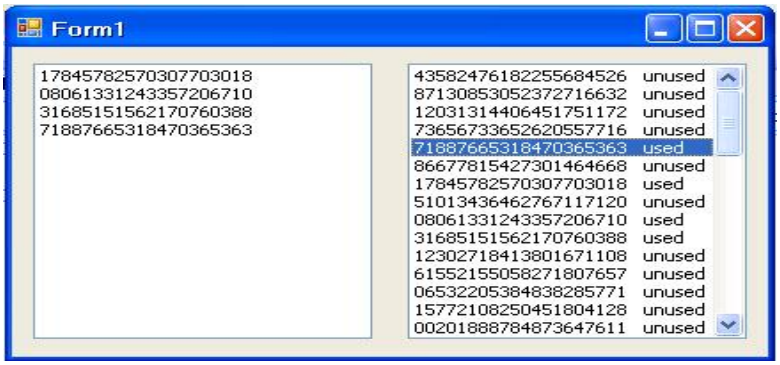
2- Identity Sub_System: تحديد او تعريف الانظمة الفرعية وذلك حسب تقسيمنا للمتطلبات.

3- Assign Requirements to Sub_System: تاثير تلك المتطلبات المحددة على الانظمة الفرعية بمعنى هل تلك المتطلبات تواءم الانظمة الفرعية اذا كان نعم فننتقل الى المرحلة التالية واذا كان لا فنعود لنحدد ونقسم المتطلبات والانظمة الفرعية ـ لاحظ اننا وضعنا سهم ذات جهتين حيث بامكاننا ان نعود ان اردنا ذلك ـ.

4- Specify Sub_System: نحدد ونعطي مواصفات لكل نظام فرعي ان اننا سوف نحدد وظيفة كل جزء من اجزاء النظام الفرعي فلا يكون هناك نظام فرعي دون عمل مثلا.

5- Define Sub_System Interfaces: تحديد الشكل والواجهات الخاصة بالنظام.

الواجب الرابع : توليد خمسين رقما عشوائيا في قائمة بحيث ان كل رقم يحتوي على عشرين خانة فاذا ما تم اختيار احد الارقام يتم تحويله الى حالة "مستخدم" ونقله الى قائمة اخرى كالتالي :




```
Public Class Form1
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        Dim rnd As New Random        'تعريف كائن خاص بالقيم العشوائية
        Dim s As String = ""
        For x1 As Byte = 1 To 50
            For x2 As Byte = 1 To 20
                s &= rnd.Next(0, 9)
            Next
            ListBox1.Items.Add(s & "   unused")
            s = ""
        Next
    End Sub

    Private Sub ListBox1_DoubleClick(ByVal sender As Object, ByVal e As System.EventArgs)
Handles ListBox1.DoubleClick
        If ListBox2.FindStringExact(ListBox1.SelectedItem.ToString.Substring(0, 20)) = -1
Then
            ListBox2.Items.Add(ListBox1.SelectedItem.ToString.Substring(0, 20))
            ListBox1.Items.Insert(ListBox1.SelectedIndex,
ListBox1.SelectedItem.ToString.Substring(0, 20) + "   used")
            ListBox1.Items.RemoveAt(ListBox1.SelectedIndex)
        End If
    End Sub

End Class
```

☺ المحاضرة الخامسة ☺

عمليات البرامجيات Software Process:

هناك عدة طرق من اجل معالجة البرامجيات سوف نذكرها خلال بحثنا القادم ولكن قبل ان نبدأ يجب ان نعرف الفرق بين هندسة النظم وهندسة البرامجيات :

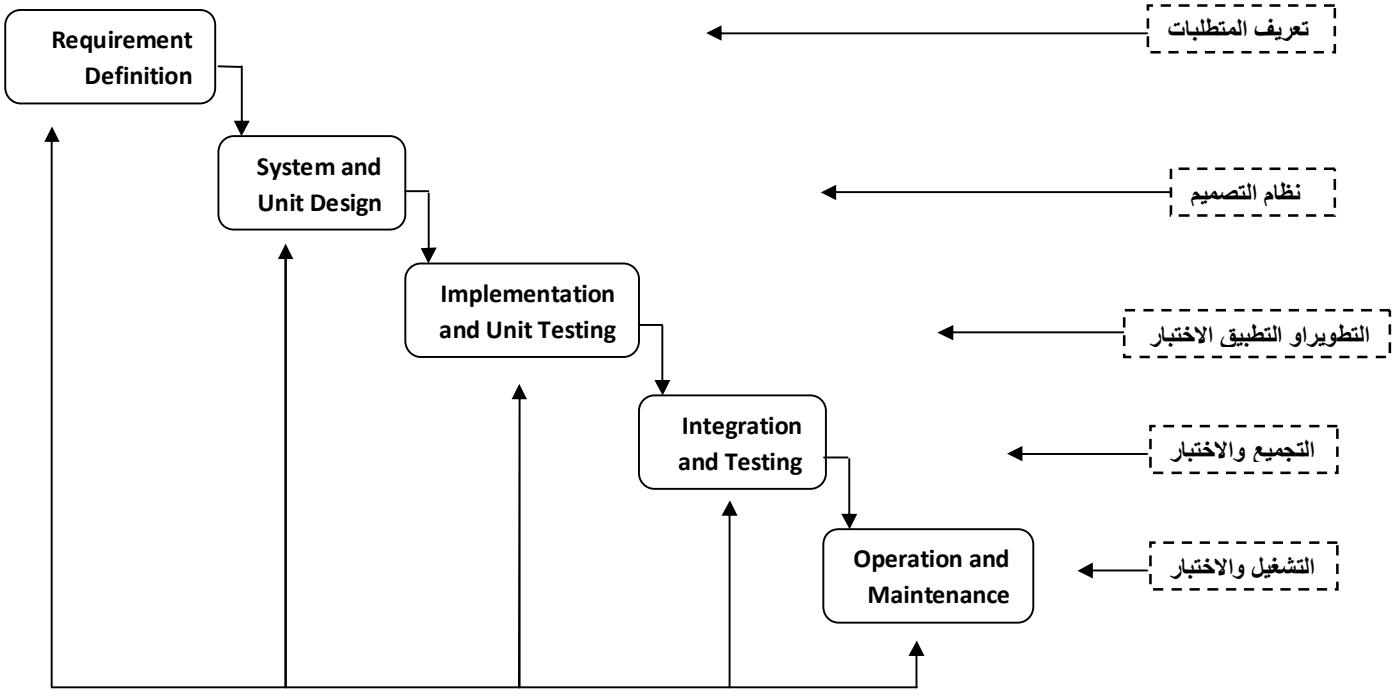
فهندسة البرامجيات : مجموعة من الانشطة المترابطة بهدف انتاج برامجيات تهتم بالمعدات HW والبرامجيات SW.

بينما هندسة النظم : مجموعة من الانشطة المترابطة بهدف انتاج برامجيات تهتم بانتاج الانظمة والبرامجيات وليس لها علاقة بالمعدات HW.:

اولا: الطريقة الانحدارية او طريقة الشلال Waterfall Model

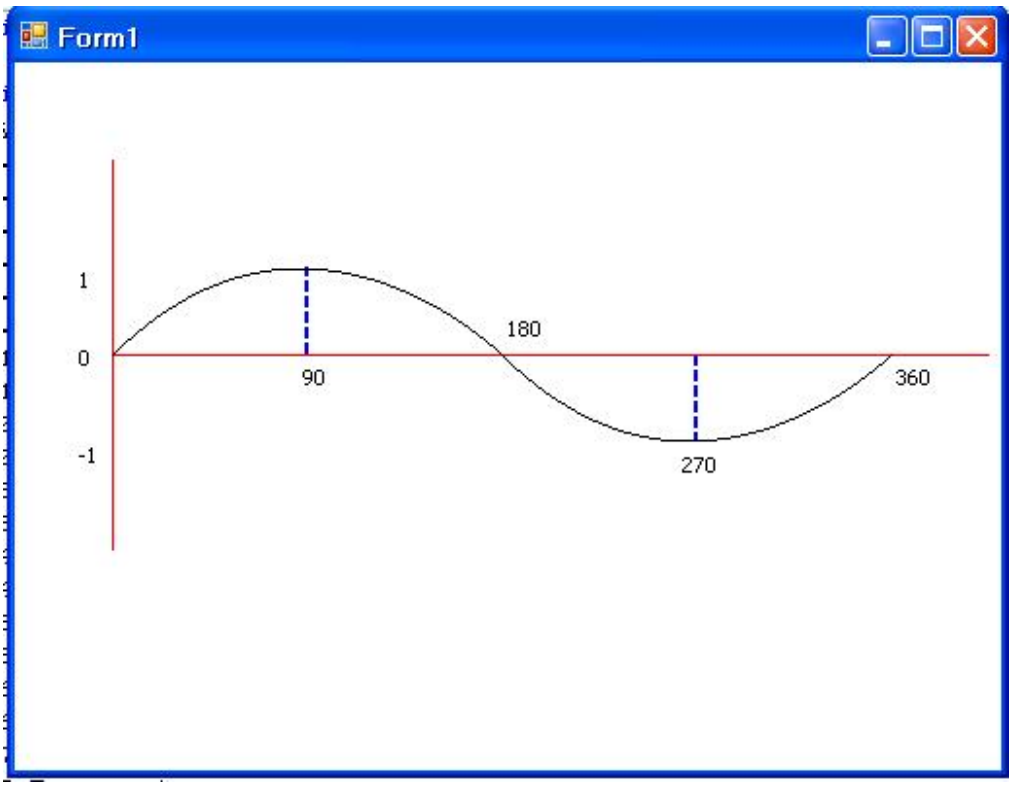
وتستخدم هذه الطريقة اذا كانت متطلبات النظام واضحة المعالم ومحددة وذلك لان هذه الطريقة اذا انتقلنا من احد المراحل فلا عودة بعدها ـ كما هو موضح بالرسم ـ وذلك لانها ستؤدي الى خسائر فعلى سبيل المثال لو اننا حددنا المعدات المفترض تواجدها وعلى هذا الاساس قام المبرمج بابرام العقد مع الشركة الراغبة في النظام وعلى اساس المعدات والتكاليف التي اعدها لهم المهندس او المحلل ومن ثم لاحظ ان المعدات ناقصة فان تكاليف المعدات الناقصة سوف يتحملها المهندس .

ويمكن توضيحها بالشكل التالي



الواجب الخامس : رسم دالة الـ COS :

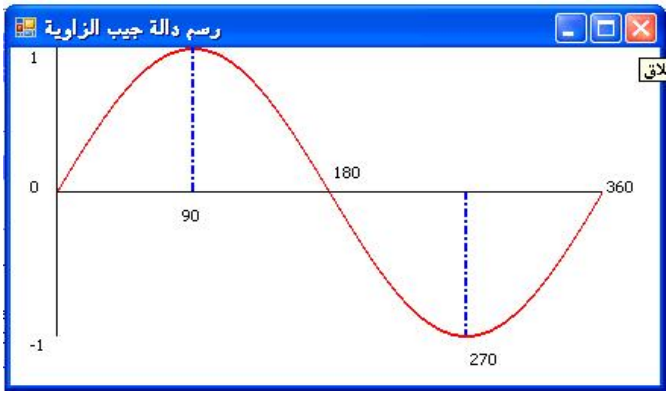
1- رسم ثابت او ستاتيكي :




```
Imports System.Drawing
Imports System.Drawing.Drawing2D


Public Class Form1
      Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim r As Graphics = Me.CreateGraphics
        Dim mypen As New Pen(Color.Blue, 2)
        mypen.DashStyle = DashStyle.Dash
        r.DrawLine(mypen, 150, 150, 150, 105)
        r.DrawLine(mypen, 350, 150, 350, 195)
        r.DrawLine(Pens.Red, 50, 250, 50, 50)
        r.DrawLine(Pens.Red, 50, 150, 500, 150)
```

```
        r.DrawBezier(Pens.Black, 50, 150, 150, 50, 250, 150, 250, 150)
        r.DrawBezier(Pens.Black, 250, 150, 350, 250, 450, 150, 450, 150)
        l1.Location() = New Point(30, 145)
        l1.Text = 0
        l2.Location = New Point(145, 155)
        l2.Text = 90
        l3.Location = New Point(250, 130)
        l3.Text = 180
        l4.Location = New Point(340, 200)
        l4.Text = 270
        l5.Location = New Point(450, 155)
        l5.Text = 360
        l6.Location = New Point(30, 105)
        l6.Text = 1
        l7.Location = New Point(30, 195)
        l7.Text = -1
        r.DrawLine(Pens.Green, 20, 300, 20, 300)
        Dim p As New Point(New Size(50, 50))
           End Sub
End Class
```

2- رسم المنحنى باستخدام الدالة sin وديناميكية (متحركة):

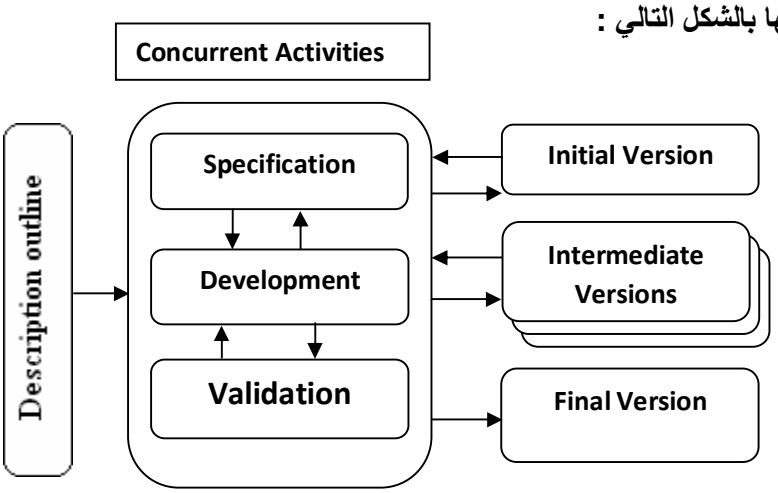


```
Imports System.Drawing
Imports System.Drawing.Drawing2D

Public Class Form1
    Dim i As Single = 0

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Timer1.Tick
        Dim gr As Graphics = Me.CreateGraphics
        Dim mypen As New Pen(Color.Blue, 2)
        mypen.DashStyle = DashStyle.DashDot
        Dim toDegree As Double = (22 / 7) / 180
        gr.DrawLine(mypen, 300, 100, 300, 200)
        gr.DrawLine(mypen, 120, 100, 120, 0)
        gr.DrawLine(Pens.Black, 30, 100, 390, 100)
        gr.DrawLine(Pens.Black, 30, 0, 30, 200)
        gr.DrawEllipse(Pens.Red, i + 30, -CInt(Math.Sin(toDegree * i) * 100) + 100, 1, 1)
        l1.Text = 1
        l1.Location = New Point(10, 0)
        l2.Text = -1
        l2.Location = New Point(10, 200)
        l3.Text = 0
        l3.Location = New Point(10, 90)
        l4.Text = 90
        l4.Location = New Point(110, 110)
        l5.Text = 180
        l5.Location = New Point(210, 80)
        l6.Text = 270
        l6.Location = New Point(300, 210)
        l7.Text = 360
        l7.Location = New Point(390, 90)
        i += 1
        If i = 360 Then
            Timer1.Enabled = False
        End If
    End Sub
End Class
```

☺ المحاضرة السادسة ☺

ثانيا : طريقة التطوير الارتقائي :

ويمكن توضيحها بالشكل التالي :

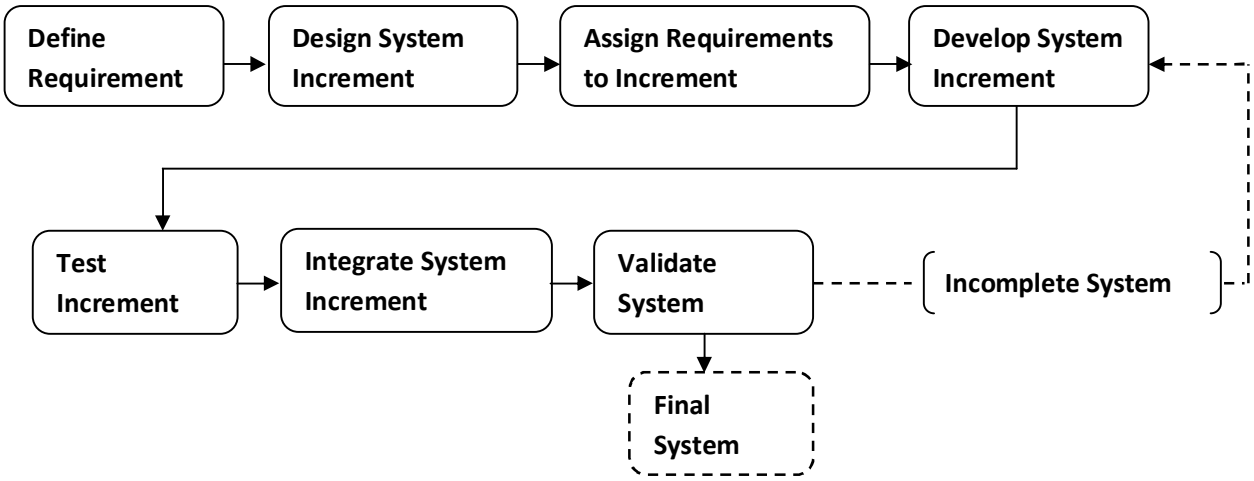


عملية التطوير في هذه الطريقة تبدا من المستثمر الذي يقوم بوضع الخطوط العريضة للنظام ومن ثم تقوم الشركة بعملية التحليل للنظام الحالي سواء اكان يدوي ام الي حتى تصل الى مواصفات النظام الجديد وتحدد المتطلبات والقيود النظام وبالتالي نحصل على نسخة اولية للنظام وقد يكون النظام القديم هو النسخة الاولية للنظام الجديد في هذه الطريقةوبعد ذلك يمكننا ان ننتقل الى التطوير Development فتصبح لدينا نسخة نسميها بالوسيطة ومن ثم الاختبار للنظام والاختبار هو اختبار للشروط الموضوعة في النظام اي هل يؤدي الوظائف والاغراض المحددة له مسبقا فان كان نعم فننتقل الى النسخة النهائية والا فنعود الى مرحلة التطوير وبالتالي تصبح لدينا نسخة وسيطة اخرى فلذلك سميناها نسخ وسيطة اي قد تكون اكثر من نسخة ومن ثم نعود الى مرحلة الاختبار مرة اخرى وهكذا حتى يتم وضع النسخة النهائية للنظام .

والمسؤل عن عملية الانتقال من مرحلة الاختبار الى الوسيطة او النهائية هو رئيس الفريق في التظوير .

ومن مزايا هذه الطريقة الانتقال من مرحلة الى اخرى دون تخوف او اي خسارة لفريق العمل لان النسخة النهائية هي التي سوف تسلم للمستثمر او طالب النظام

ثالثا : الطريقة التزايدية Incremental Development :



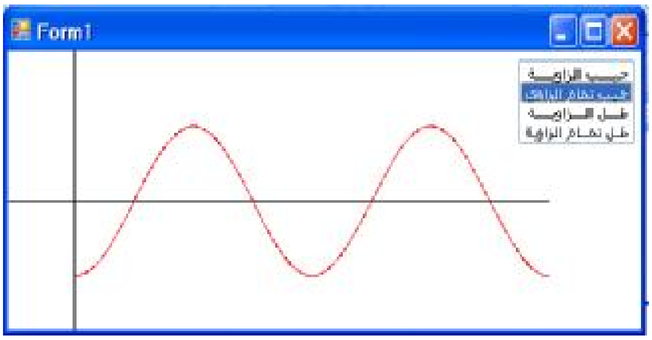
1. Define Requirements: تحديد متطلبات جديدة على نظام قديم قائم.
2. Design System Increment: تصميم النظام الجديد (وفق خطوات هندسة البرمجيات).
3. Assign Requirements to Increment: تخصيص تلك المتطلبات على النظام الجديد (على الانظمة الفرعية التي تم اضافتها في مرحلة التصميم).
4. Develop System Increment: تطوير ما تم اضافته في النظام القديم (كتابة الكود البرمجي).
5. Test Increment: اختبار تلك الزيادة في النظام.
6. Integrate System Increment: تجميع النظام لوضع نظام كلي متكامل.
7. Validate System: اختبار النظام الكلي
8. Final System: انتاج النظام النهائي.

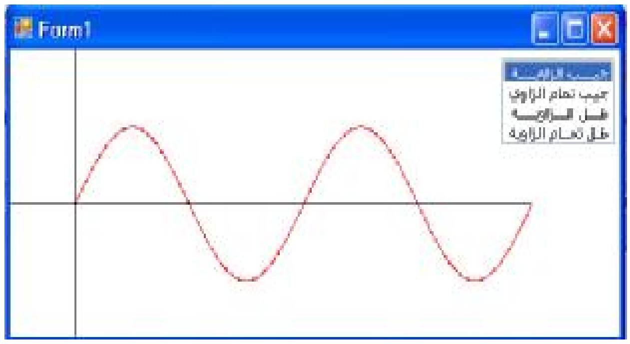
نلاحظ في الطريقة السابقة انها تتميز باننا نقوم بتطوير نظام واضافة وضائف او متطلبات عليه وبالتالي فأن كل مرحلة يتم فيها اضافة شيء جديد وعند الوصول الى المرحة السابعة وهي اختبار النظام فاننا نقوم باختبار النظام فان كان النظام مكتملا وقد حقق كل الشروط التي وضعت له وخالي من الاخطاء فاننا نقوم بوضع النسخة النهائية للنظام، اما ان كان هناك أي نقص او خلل بعد اختبار النظام فانه يمكننا ان نعود الى المرحلة الرابعة وهي تطوير النظام من اجل معالجة تلك الاخطاء ونستمر هكذا حتى نصل الى النسخة النهائية.

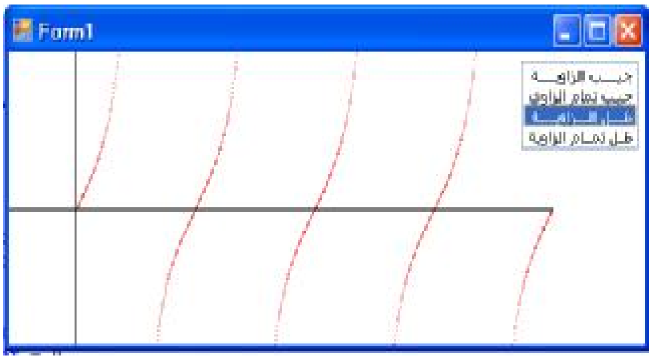
<u>الواجب السادس و السابع : رسم الدوال المثلثية او أي معادلة اخرى</u>

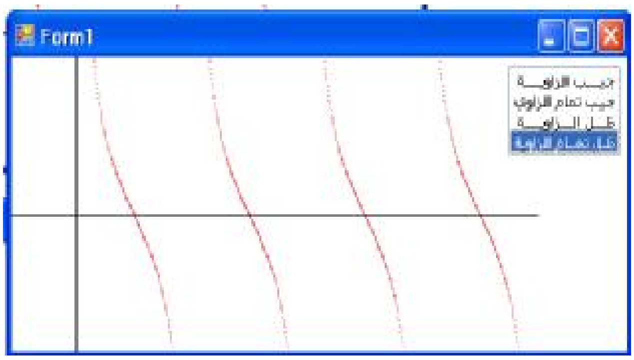
على سبيل المثال المعادلة التالية

$Y=X^2$





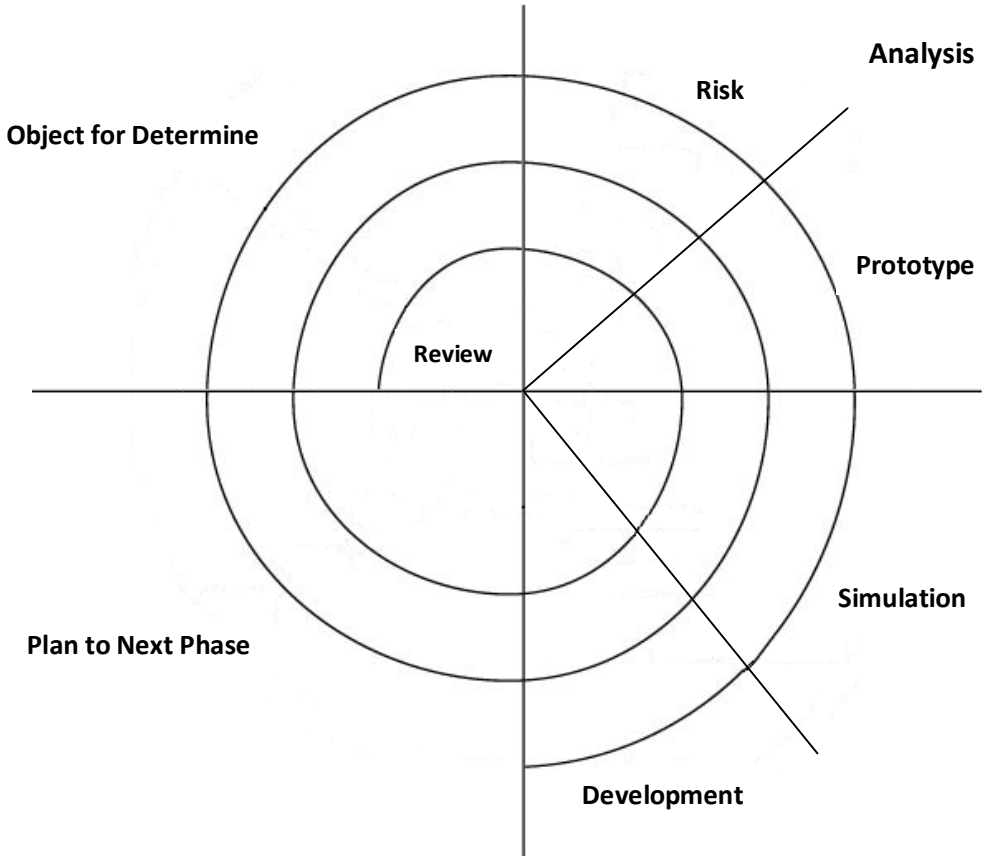




```
Imports System.Drawing.Printing
Imports System.Drawing.Drawing2D

Public Class Form1
    Dim p1 As Integer = 0
    Dim p2 As Integer = 0
    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Timer1.Tick
        Dim gr As Graphics = Me.CreateGraphics
        Dim toDegree As Single = -22 / 7 / 180
        gr.ScaleTransform(0.5, 0.5)
        If p1 < 820 Then
            gr.DrawEllipse(Pens.Black, 100, p1, 1, 1)
            gr.DrawEllipse(Pens.Black, p1, 200, 1, 1)
        End If
        p1 += 1
        If p1 < 720 Then
            If ListBox1.SelectedIndex = 0 Then
                gr.DrawEllipse(Pens.Red, p1 + 100, CInt(Math.Sin(p1 * toDegree) * 100 + 200), 1, 1)
            ElseIf ListBox1.SelectedIndex = 1 Then
                gr.DrawEllipse(Pens.Red, p1 + 100, CInt(Math.Cos(p1 * toDegree) * 100 + 200), 1, 1)
            ElseIf ListBox1.SelectedIndex = 2 Then
                gr.DrawEllipse(Pens.Red, p1 + 100, CInt(Math.Tan(p1 * toDegree) * 100 + 200), 1, 1)
            ElseIf ListBox1.SelectedIndex = 3 Then
                gr.DrawEllipse(Pens.Red, p1 + 100, CInt(1 / Math.Tan(p1 * toDegree) * 100 + 200),
1, 1)
            ElseIf ListBox1.SelectedIndex = 4 Then
                For Each l As Control In Controls
                    If TypeOf l Is Label Then l.Visible = False
                Next
                gr.DrawEllipse(Pens.Red, p1 + 100, CInt(-p1 ^ 2 * 0.05 + 200), 1, 1)
                p2 -= 1
                gr.DrawEllipse(Pens.Red, p2 + 100, CInt(-p1 ^ 2 * 0.05 + 200), 1, 1)
            End If
        End If
    End Sub


    Private Sub ListBox1_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ListBox1.SelectedIndexChanged

        p2 = 0
        p1 = 0
        Me.Refresh()
        Timer1.Enabled = True
        For Each l As Control In Controls
            If TypeOf l Is Label Then l.Visible = True
        Next
        putlabels()
    End Sub
    Private Sub putlabels()
        l1.Location = New Point(55, 120)
        l1.Text = 0
        l2.Location = New Point(90, 120)
        l2.Text = 90
        l3.Location = New Point(128, 120)
        l3.Text = 180
        l4.Location = New Point(170, 120)
        l4.Text = 270
        l5.Location = New Point(225, 120)
        l5.Text = 360
    End Sub
End Class
```

☺ المحاضرة السابعة ☺

رابعا: طريقة التطوير الحلزونى Spiral Development :



1. **Object Determine: تحديد المتطلبات بعد مراجعة انظمة السوق لمعرفة ما هي الاشياء او الوظائف التي لا توجد في الانظمة الموجودة في السوق وهذه الخطوة نسميها Review.**
2. **Analysis: التحليل وفي هذه المرحلة نقوم بتحليل المخاطر الممكن حدوثها وذلك من خلال عمل مجموعة من الانشطة من اجل التقليل من تلك المخاطر او الغاءها ومن المخاطر الممكن حدوثها ان يقدم رئيس الفريق استقالته او ان ينسحب احد الخبراء والمعتمد عليهم في النظام من الفريق وغيرها فتلك مخاطر لكنها ليست مخاطر على حياة الانسان وانما على عملة، ومن ثم تأتي مرحلة الـ Prototypeضمن مراحل التحليل وفي هذه الخطوة نقوم بوضع محاذاة اولية للنظام او تصور اولي على ورق او ان يكون ذلك التصور خوارزميات لوظائف النظام او رسومات توضيحية.**

3. **Simulation: من خلال مرحلة المحاكاة نضع نموذج يعبر عن النظام او برنامج مبدئي (كود برمجي) لكنه ليس نهائي للنظام.**
4. **Development: نبدأ هنا بتطوير النظام والمقصود كتابة الكود البرمجي.**
5. **Plan to Next Phase: التخطيط للمرحة القادمة أي يتم دراسة ماذا نريد ان نفعل في المرحلة التالية طالما ان النظام لم يكتمل.**

وتستمر هذه المراحل بالدوران متنقلين بين انظمة النظام الفرعية حتى نصل الى النسخة النهائية وبالتالي لن ننتقل الى المرحلة رقم خمسة وهي التخطيط للمرحلة القادمة.

Architectural Design التصميم المعماري

هذا العنوان الرئيسي الذي وضعناه ليس المقصود به تصميم الانظمة وفق ما تطرقنا اليه مسبقا وانما نقصد به كيف سيكون اسلوب بناء النظام وكيف سيتم التحكم بالانظمة الفرعية وما هي طريقة التواصل بين تلك الانظمة وطريقة تمرير ومعالجة البيانات فيما بينها، كل ذلك سوف يتم مناقشته في بحثنا القادم ويمكن ان نضع ما سبق في العناوين الثلاثة الرئيسية التالية.

A. **Structural System Models طرق هيكلة الانظمة (بناء الانظمة).**
B. **Control Models طرق التحكم بالانظمة.**
C. **Modular Decomposition طرق الاتصال.**

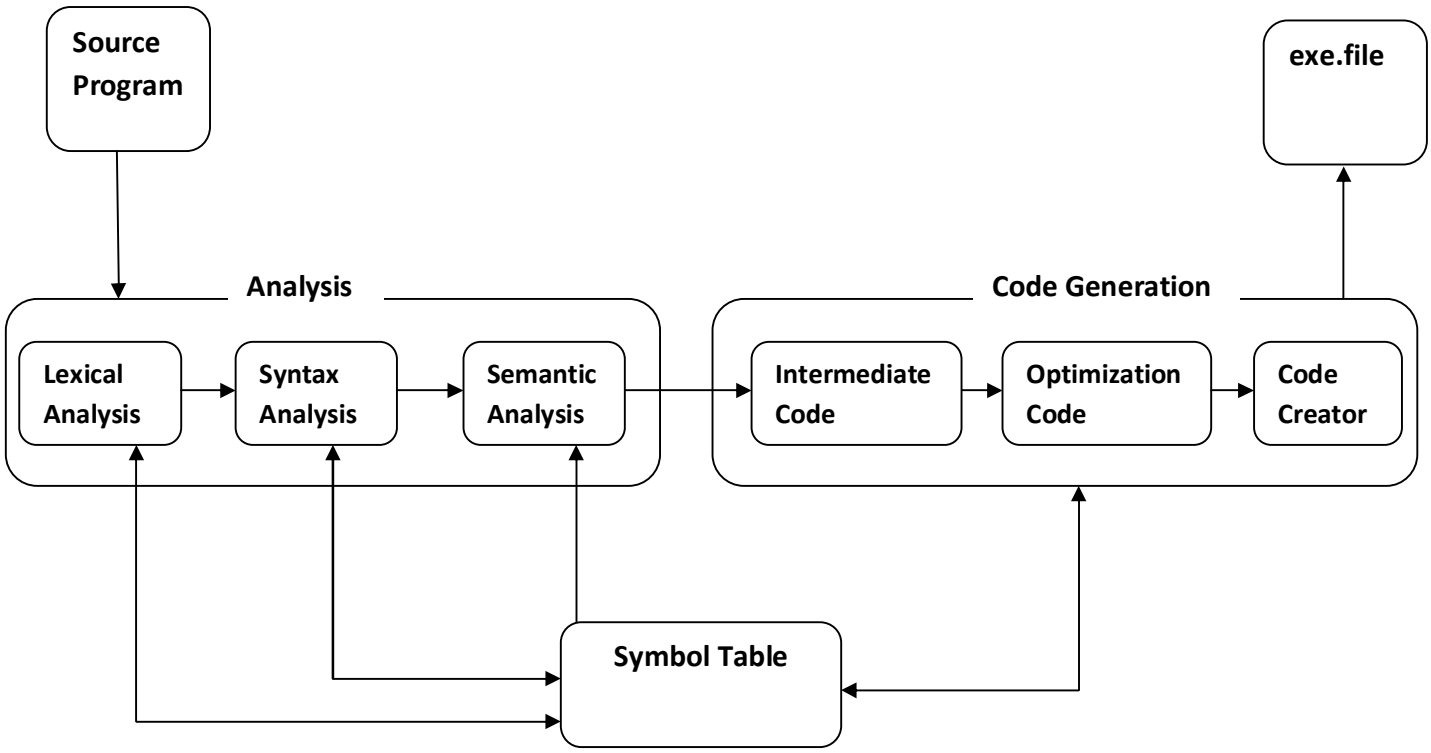
A- Structural System Models:

أي ما هي الطريقة التي سيتم بناء النظام بها بحيث يمكن لنا من خلال هذه الطريقة ان نستغل موارد النظام بشكل اكبر وكذا امكانية معالجة الاخطاء وغيرها، ونحدد الطريقة من خلال رسم صندوقي يصف طريقة بناءنا للنظام .

ولدينا ثلاث طرق لبناء الانظمة هي كالتالي :

1. **Repository model: في هذه الطريقة يكون لدينا شيئ مشترك بين كل دوال او اجراءات النظام كأن يكون جداول (قاعدة بيانات) او ان يكون اجراء معين فيه شرط معين مثل اجراء الترقيم التلقائي للتقارير ، بمعنى ان احد الانظمة الفرعية يكون مرتبط بباقي الانظمة والانظمة الاخرى تراقب ذلك النظام في حالة حدوث أي حدث لتلك الانظمة فانها تقوم بالتحرك نحو النظام الفرعي وبالتالي عمل تصرف معين.**

وخير مثال لهذه الطريقة هي المترجمات، حيث ان المترجمات ترتبط بجدول معين به كل الكلمات المحجوزة للغة البرمجية فلو كان هناك أي خطا في البرنامج فان المترجم يتحسس ذلك الخطا بعد ان يقوم بالمقارنة مع الجدول المشترك .



1- Source Program: وهو البرنامج المكتوب من المستخدم بغض النظر عن اللغة المكتوب بها.

2-Analysis: مرحلة التحليل وفيها :

- Lexical Analysis: تحليل البرنامج المكتوب من قبل المستخدم وتحديد الكلمات المحجوزة للغة مثل forوif وغيره.
- Syntax Analysis: مقارنة مفردات البرنامج مع الكلمات المكتوبة في الجدول Symbol Tableمن حيث الاخطاء المحتملة عند كتابة الكود كان يكتب المستخدم الكلمة Fore بدلا من for او ان ينسى الفاصلة المنقوطة في حالة اللغات التي بها الفاصلة مثل C++.
- Semantic Analysis: تحليل المعاني أي ماذا تعني كلمة If من حيث انه ياتي بعدها مقارنة تنتج Trueاو False او ماذا تعني كلمة forوكل ذلك يتم بالمقارنة مع الجدول الوسيط.Symbol Table

(من الملاحظ اننا نعتمد في كل خطوة على الجدول المشتركSymbol Table.

3- Code Generation: مرحلة بناء الكود ،أي ان يتم في هذه المرحلة انشاء الكود البرمجي الخاص بالاله وغيره وفيها :

- Intermediate Code: تحويل واعادة بناء الكود الذي قام بكتابة المبرمج الى كود وسيط وهو كود ما بين لغة الالة وكود بلغة الانسان.

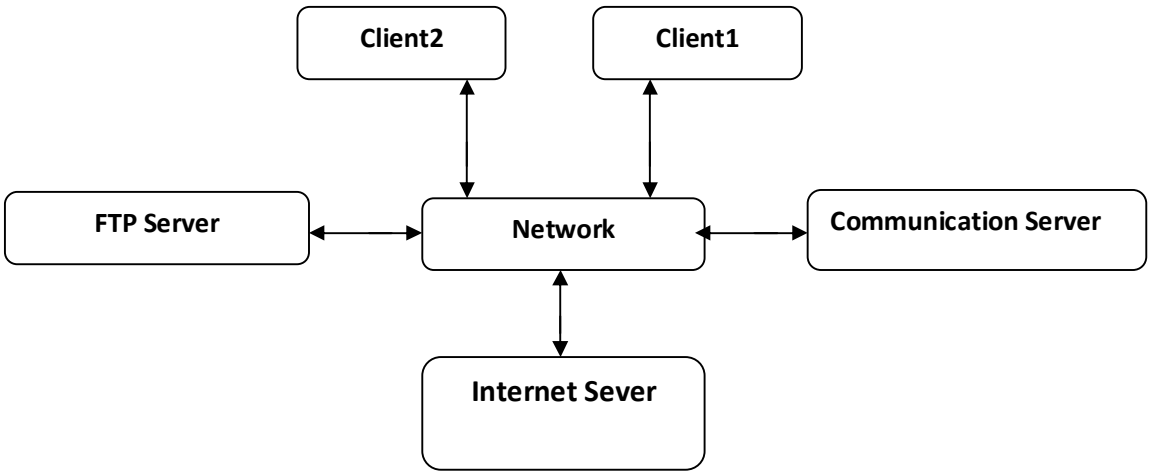
- **Optimization Code: تحسين الشفرة أي الغاء ماليس ضروري في البرنامج كان يتم الغاء التعليقات Commit او الفواصل المنقوطة.**
- **Code Creator: بناء الكود البرمجي من الشفرة العادية شفرة بلغة فرعية من لغة الاوامر وهي اللغة التي تحول الشيفرة الى (01) Binary.**

4- exe.File: انتاج البرنامج التنفيذي.

2. Client/Server Model: وفي هذه الطريقة نقوم بانشاء برنامج Client أي مستفيد ومرتبط ببرنامج Server وفق شبكة والـServer خادم لتلك الاجهزة المستفيدة وكذلك يقوم بعملية التحكم والمراقبة.

وفي هذه الهيكلية من البرامج فاننا لا نعتمد على بيانات مشتركة كما كان في الطريقة الاولى.

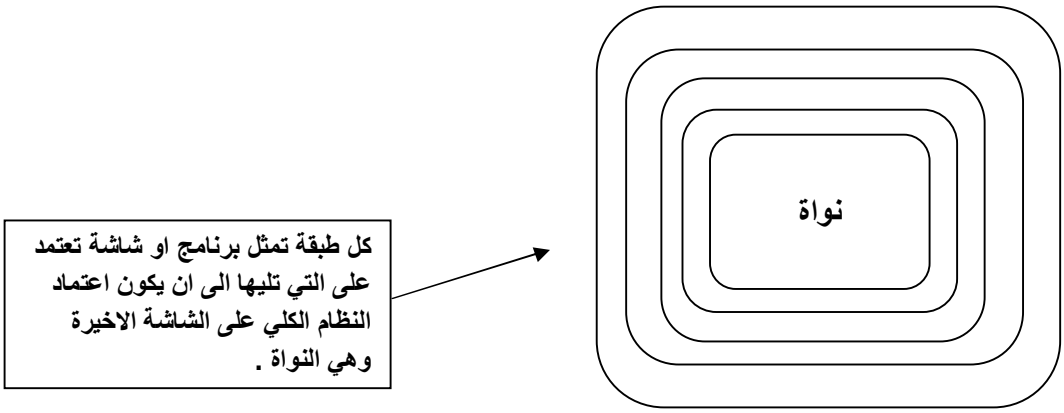
لناخذ المثال التالي على برنامج يعتمد على هذه الطريقة وهي تقنية الانترنت والتي تتكون من Server و اجهزة مستفيدة اخرى Client وتربط بينهم شبكة Network مع وجود خدمة الاتصالات وتقنية تنزيل البرامج التطبيقية على شكل ملفات مضغوطة وفق صلاحيات محددة FTP :



3. Abstract Machine Model:

تعتمد هذه الطريقة على بناء الانظمة على شكل طبقات وكل طبقة من الطبقات تعتمد على التي تحتها فعلى سبيل المثال شاشات الMDI والتي من خلالها نقوم بانشاء شاشة رئيسية هي الشاشة الام والتي قد تحوي على اكثر من شاشة بداخلها فاذا ما اغلقنا حد تلك الشاشات فان الشاشة الام لا تتاثر بينما لو اغلقنا الشاشة الام فان جميع الشاشات سوف تغلق ، وكمثال اكثر وضوحا انظمة التشغيل والتي تعتمد اساسا على هذه الطريقة فعند انشاء انظمة التشغيل يؤخذ في عين الاعتبار ان يحتوي بداخله على اكثر من برنامج و تلك البرامج في نفس الوقت قد تحوي على برامج اخرى بداخلها فاذا ما اغلقنا احد تلك البرامج فان نظام التشغيل لن يتاثر وبالتالي فان نظام التشغيل هو نواة النظام ككل .

والشكل التالي يوضح تلك العلاقة بين البرامج في الطبقات المختلفة :



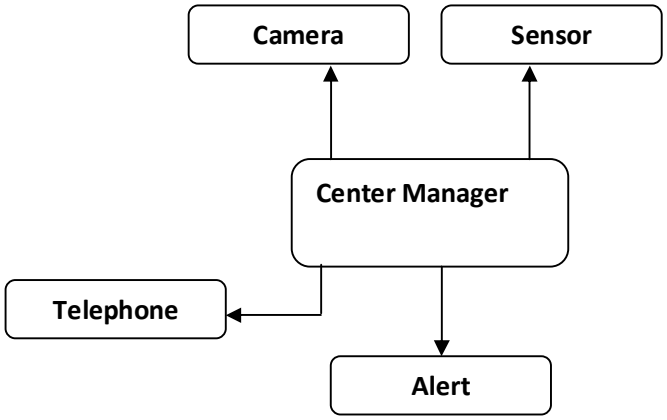
B– Control Models طرق التحكم بالانظمة:

وبعد ان نحدد الهيكلية التي سيكون عليها النظام لابد ان نحدد كيف سنقوم بالتحكم بالنظام أي ماهي استراتيجية السيطرة على الانظمة الفرعية او الاجراءات اوغيره .

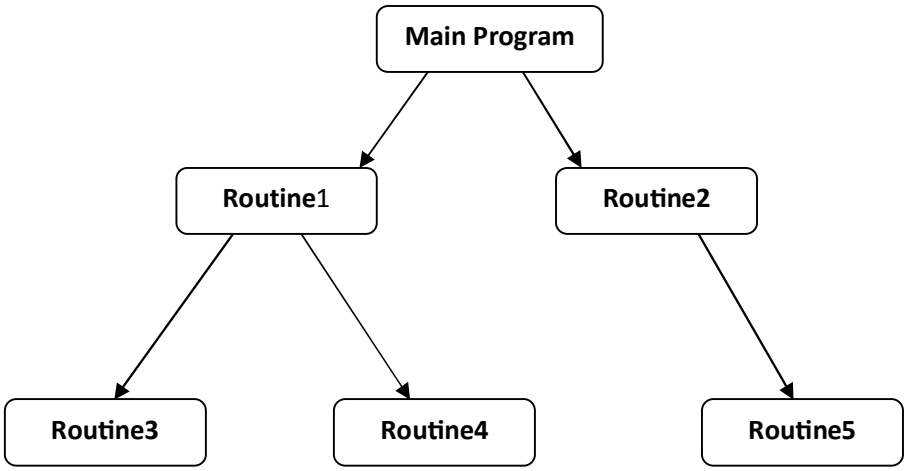
ولدينا نوعين من التحكم هما :

A. Centralized: التحكم المركزي أي ان يتم التحكم بالنظام ككل من خلال برنامج رئيسي او دالة رئيسية ومن انواعها:

i. Manager: وفي هذه الطريقة يتم التحكم بالانظمة بشكل مركزي بحيث يقوم البرنامج الرئيسي بالتحكم بكل البرامج الاخرى والايعاز لها بالعمل او التوقف عن العمل والشكل التالي يوضح العلاقة بين البرنامج الرئيسي والبرامج الاخرى وفق هذه الطريقة :



ii. وفي هذه الطريقة يكون التحكم مركزي لكن في نفس الوقت يمكن للبرنامج الرئيسي ان يتحكم بمجموعة من البرامج ولكن لا يقدر ان يتحكم بتلك البرامج التي تتحكم بها البرامج التي يتحكم هو بها !!!! L بمعنى اخر ان سيطرة البرنامج الرئيسي ليست الا في البرامج المرتبطة به مباشرة وتلك البرامج المرتبطة به تتحكم بالبرامج التي ترتبط بها على كلٍ الشكل التالي يوضح المقصود :



B. **Events Based**: في هذه الطريقة فان التحكم يكون على اساس الاحداث فعند انطلاق الحدث يتم تنفيذ عمل معين ومنها نوعين هما :

i. **Real Time**: وتتم في هذه الطريقة تنفيذ الاعمال عند حدوث حدث مرتبط بالوقت الحقيقي للجهاز كان نقوم بعملية النسخ الاحتياطية كل ثلاثة ايام وخمس ساعات واربع دقائق وثلاث ثواني وخمسين ملي ثانية.

ii. **Procedural**: أي يتم تنفيذ اجراءات بمجرد انطلاق الحدث ولا تعتمد على الوقت الحقيقي بل عند الضغط مثلا على احد الازرار او عند تغير قيمة ما وخير مثال لذلك ما نقوم بتطبيقة في الواجبات السابقة كاملة (في الواجب الثالث توجد بها طريقة انشاء حدث من قبل المبرمج ولا يعتمد على احداث اللغة ان اراد ذلك حيث سينطلق الحدث بمجرد ان ترتفع قيمة المتغير par الى اكثر من خمسين).

C– Modular Decomposition طرق الاتصال:

وبعد ان ننتهي من تحديد نوع التحكم للنظام ياتي دور تحديد كيفية الاتصال بين البرامج الفرعية المختلفة أي استراتيجية الاتصال بين مكونات النظام ولها طريقتين هما

A. **Object Oriented Programming (OOP)**: أي تتم عملية الاتصال على اساس البرمجة الكائنية او الشيئية.

B. **Data Flow Model**: تدفق البيانات.

انتهى المقرر ،،

بالتوفيق والنجاح للجميع وان اخطات فمن نفسي وان اصبت فمن الله وحده وليعذرني من وجد الخطأ

بسام الكبسي / نظم معلومات