

**گزارش پروژه سوم**

**نام و نام خانوادگی**

نگار فتحی

**شماره دانشجویی**

97723137

**استاد**

دکتر سعید پارسا

**درس**

کامپایلر پیشرفته

در پروژه انجام‏شده، یک برنامه نوشته‏شده به زبان C#، براساس دو اصل Coupling و Cohesion پیمانه‏بندی مجدد شده‏است. جهت انجام این پروژه، گام‏های زیر طی شده‏است.

* **انتخاب کد منبع جهت پیمانه‏بندی مجدد:**

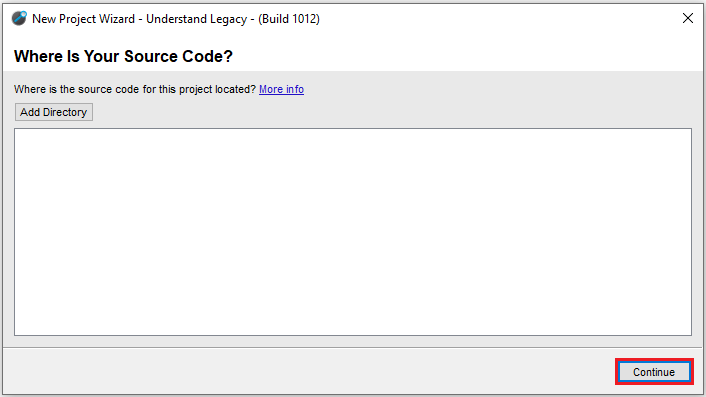
برنامه‏ای که قصد پیمانه‏بندی مجدد آن را داریم، برنامه یک ماشین حساب به زبان C# است که کد آن را می‏توان از آدرس زیر دانلود کرد.

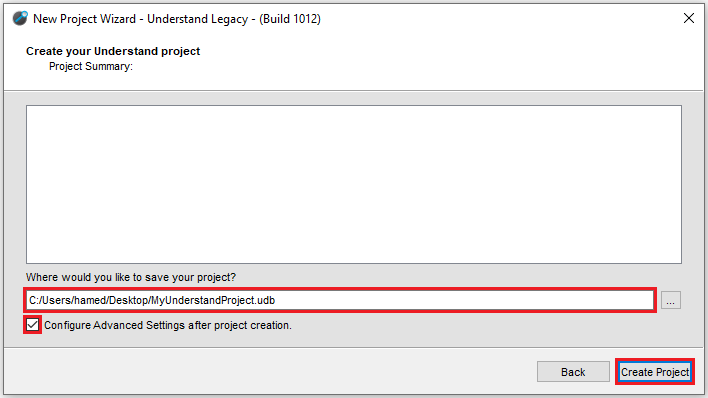
<https://github.com/peteroupc/Calculator>

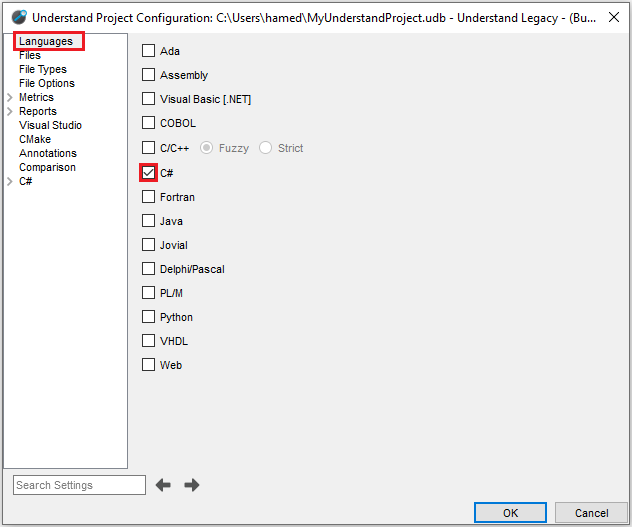
* **استخراج مدل ارتباطی کلاس‏های برنامه به‏کمک ابزار Understand:**

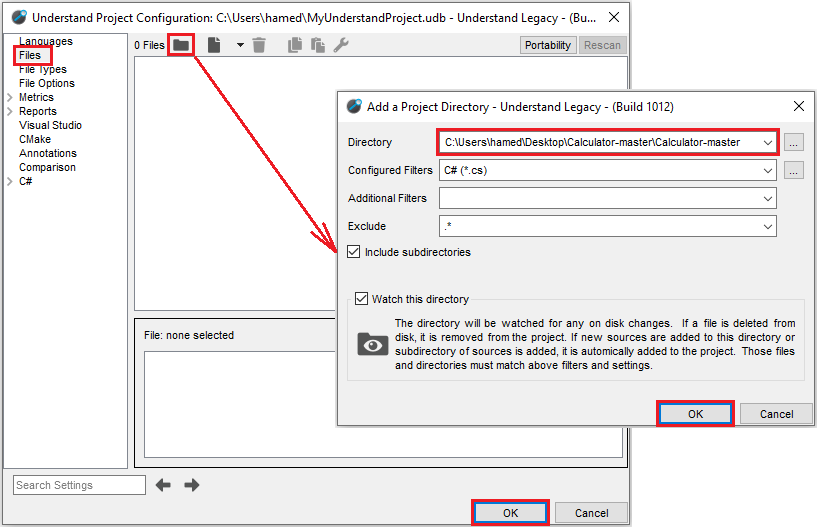
در گام اول جهت پیمانه‏بندی مجدد برنامه موردنظر، باید مدل ارتباطی کلاس‏های برنامه را استخراج کنیم که این کار به‏کمک ابزار Understand قابل انجام است. بدین‏منظور باید ابتدا با طی‏کردن مراحل زیر برنامه موردنظر را به ابزار Understand داد.

Understand ⇒ File ⇒ New ⇒ Project ⇒





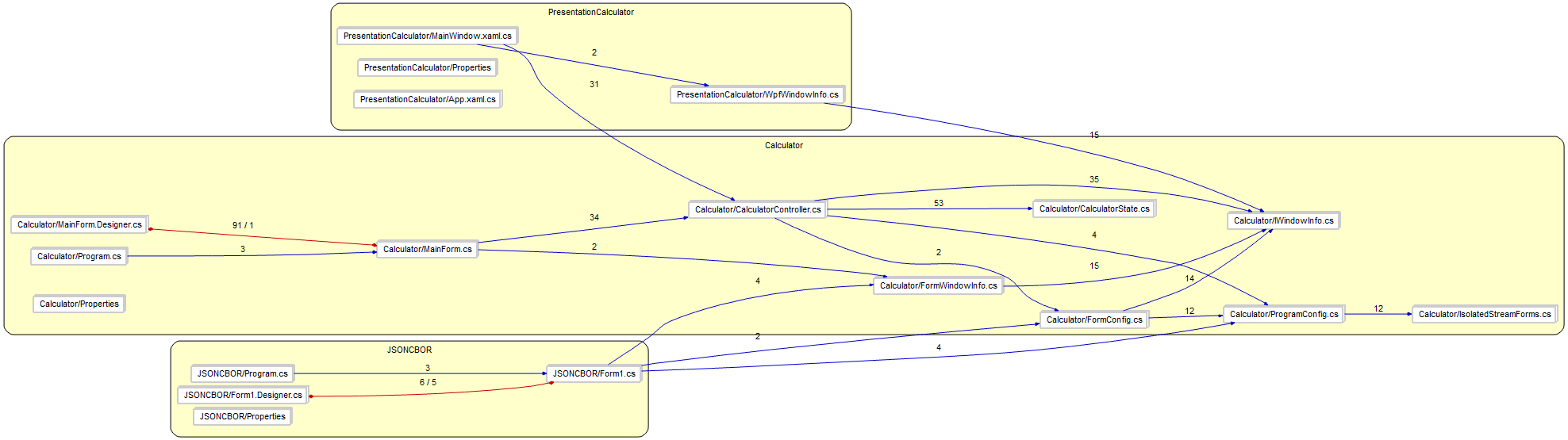




حال می‏توان با طی‏کردن مراحل زیر، مدل ارتباطی کلاس‏های برنامه را در قالب یک گراف مشاهده کرد.

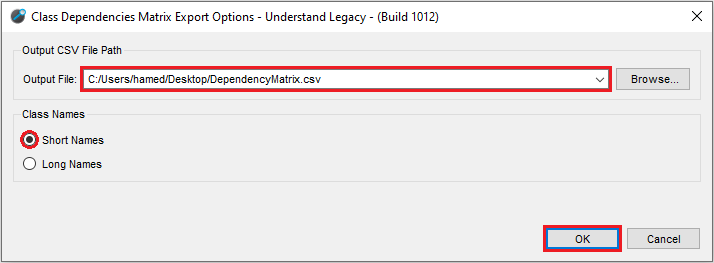
Understand ⇒ Graphs ⇒ Dependency Graphs ⇒ By Directory Structure.

برای مثال، برای برنامه موردنظر، گراف وابستگی به‏صورت زیر است.

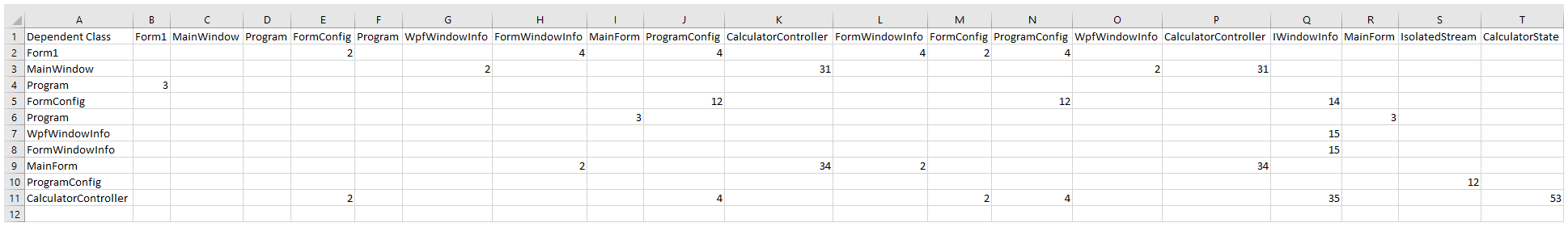


حال جهت دستیابی‏به هدف پیمانه‏بندی مجدد برنامه موردنظر، باید مدل ارتباطی کلاس‏های برنامه را در قالب یک فایل .CSV استخراج نمود. بدین‏منظور باید مراحل زیر طی شود.

Understand ⇒ Reports ⇒ Dependency ⇒ Class Dependencies ⇒ Export Matrix CSV ⇒

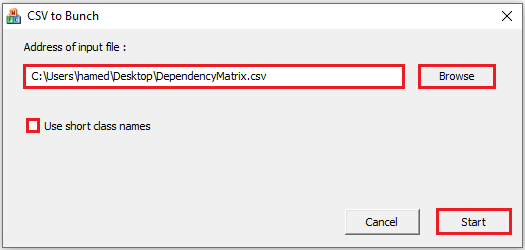


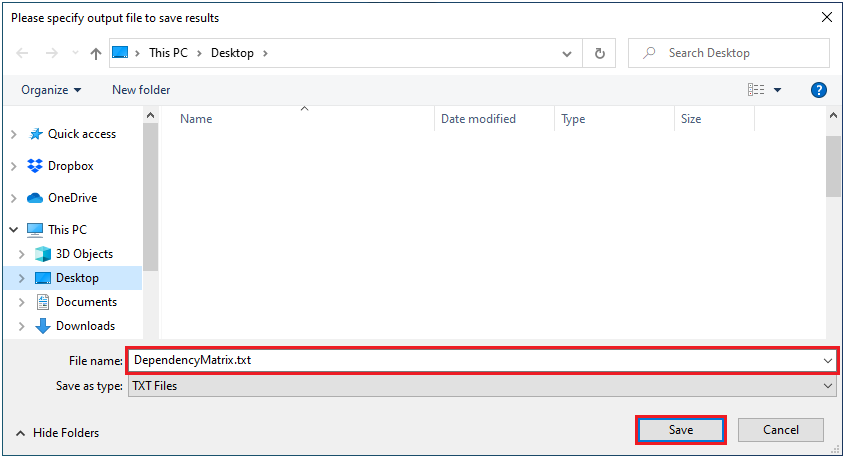
برای مثال، برای برنامه موردنظر، ماتریس وابستگی به‏صورت زیر است.

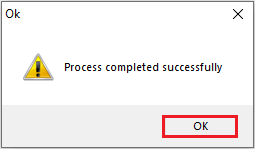


* **تبدیل خروجی ابزار Understand به ورودی مورد پذیرش توسط ابزار Bunch:**

در گام دوم جهت پیمانه‏بندی مجدد برنامه موردنظر، باید مدل ارتباطی کلاس‏های برنامه را که در قالب یک فایل .CSV استخراج کردیم، آماده پذیرش توسط ابزار Bunch کنیم. ورودی ابزار Bunch، یک حالت خاص از ماتریس وابستگی فوق، به‏صورت سطری است. برای این کار از نرم‏افزار CSV2Bunch که توسط آقای مقدسی به زبان C++ نوشته شده‏است، استفاده می‏کنیم. مراحل طی‏شده جهت تبدیل خروجی ابزار Understand به ورودی مورد پذیرش توسط ابزار Bunch به‏کمک نرم‏افزار CSV2Bunch به‏صورت زیر است.







برای مثال، برای برنامه موردنظر، خروجی نرم‏افزار CSV2Bunch یک فایل .TXT به‏صورت زیر است.

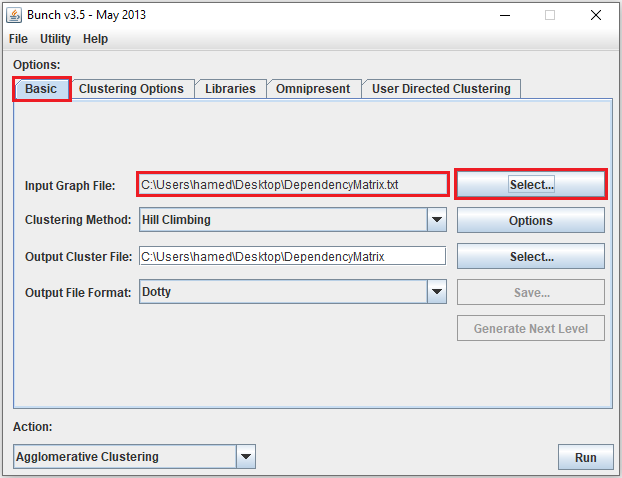
|  |
| --- |
| Form1 FormConfig 2  Form1 FormWindowInfo 4  Form1 ProgramConfig 4  Form1 FormWindowInfo 4  Form1 FormConfig 2  Form1 ProgramConfig 4  MainWindow WpfWindowInfo 2  MainWindow CalculatorController 31  MainWindow WpfWindowInfo 2  MainWindow CalculatorController 31  Program Form1 3  FormConfig ProgramConfig 12  FormConfig ProgramConfig 12  FormConfig IWindowInfo 14  Program MainForm 3  Program MainForm 3  WpfWindowInfo IWindowInfo 15  FormWindowInfo IWindowInfo 15  MainForm FormWindowInfo 2  MainForm CalculatorController 34  MainForm FormWindowInfo 2  MainForm CalculatorController 34  ProgramConfig IsolatedStream 12  CalculatorController FormConfig 2  CalculatorController ProgramConfig 4  CalculatorController FormConfig 2  CalculatorController ProgramConfig 4  CalculatorController IWindowInfo 35  CalculatorController CalculatorState 53 |

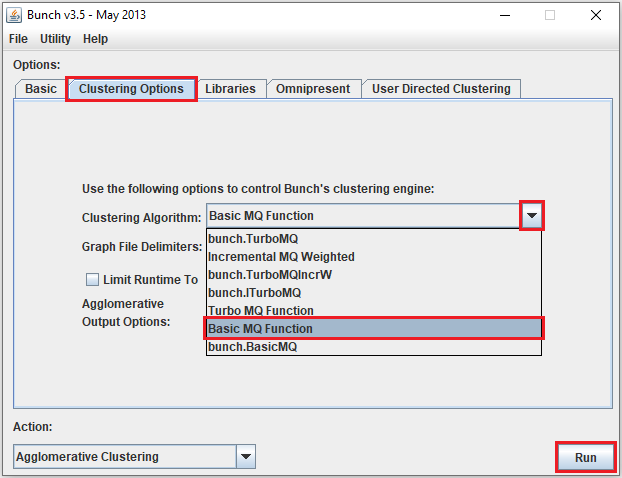
* **خوشه‏بندی مدل ارتباطی کلاس‏های برنامه به‏کمک ابزار Bunch:**

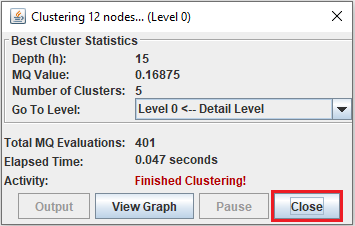
در گام سوم جهت پیمانه‏بندی مجدد برنامه موردنظر، باید مدل ارتباطی کلاس‏های برنامه را که در قالب یک فایل .TXT استخراج کردیم، جهت خوشه‏بندی به ابزار Bunch بدهیم. بدین‏منظور باید ابتدا با وارد کردن دستور زیر در Command Prompt، ابزار Bunch را اجرا کنیم.

java -jar Bunch-3.5.jar

سپس باید مراحل زیر جهت خوشه‏بندی مدل ارتباطی کلاس‏های برنامه طی شود.





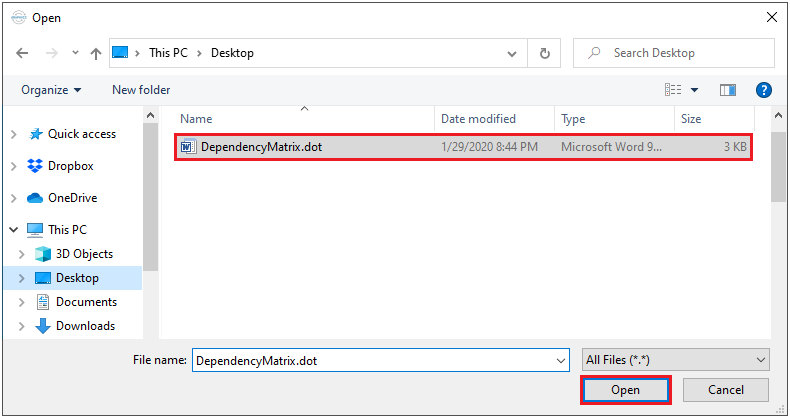


برای مثال، برای برنامه موردنظر، خوشه‏بندی مدل ارتباطی کلاس‏ها یک فایل .DOT به‏صورت زیر است.

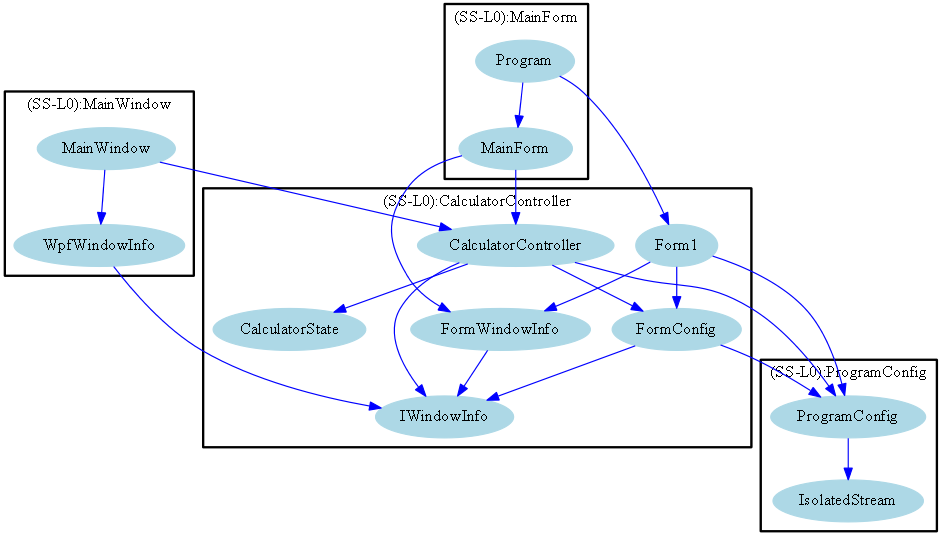
|  |
| --- |
| /\* ------------------------------------------------------------ \*/  /\* created with bunch v3 \*/  /\* Objective Function value = 0.1875\*/  /\* ------------------------------------------------------------ \*/  digraph G {  size= "10,10";  rotate = 90;  subgraph cluster0 {  label = "(SS-L0):MainForm";  color = black;  style = bold;  "Program"[label="Program",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  "MainForm"[label="MainForm",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  }  subgraph cluster1 {  label = "(SS-L0):CalculatorController";  color = black;  style = bold;  "FormWindowInfo"[label="FormWindowInfo",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  "IWindowInfo"[label="IWindowInfo",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  "FormConfig"[label="FormConfig",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  "Form1"[label="Form1",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  "CalculatorState"[label="CalculatorState",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  "CalculatorController"[label="CalculatorController",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  }  subgraph cluster2 {  label = "(SS-L0):MainWindow";  color = black;  style = bold;  "WpfWindowInfo"[label="WpfWindowInfo",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  "MainWindow"[label="MainWindow",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  }  subgraph cluster3 {  label = "(SS-L0):ProgramConfig";  color = black;  style = bold;  "IsolatedStream"[label="IsolatedStream",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  "ProgramConfig"[label="ProgramConfig",shape=ellipse,color=lightblue,fontcolor=black,style=filled];  }  "MainForm" -> "CalculatorController" [color=blue,font=6];  "MainForm" -> "FormWindowInfo" [color=blue,font=6];  "CalculatorController" -> "CalculatorState" [color=blue,font=6];  "CalculatorController" -> "IWindowInfo" [color=blue,font=6];  "CalculatorController" -> "FormConfig" [color=blue,font=6];  "CalculatorController" -> "ProgramConfig" [color=blue,font=6];  "Program" -> "MainForm" [color=blue,font=6];  "Program" -> "Form1" [color=blue,font=6];  "MainWindow" -> "WpfWindowInfo" [color=blue,font=6];  "MainWindow" -> "CalculatorController" [color=blue,font=6];  "Form1" -> "FormConfig" [color=blue,font=6];  "Form1" -> "ProgramConfig" [color=blue,font=6];  "Form1" -> "FormWindowInfo" [color=blue,font=6];  "FormConfig" -> "IWindowInfo" [color=blue,font=6];  "FormConfig" -> "ProgramConfig" [color=blue,font=6];  "FormWindowInfo" -> "IWindowInfo" [color=blue,font=6];  "WpfWindowInfo" -> "IWindowInfo" [color=blue,font=6];  "ProgramConfig" -> "IsolatedStream" [color=blue,font=6];  } |

می‏توان به‏کمک ابزار Graphviz، به نمایش گرافیکی خوشه‏بندی صورت‏گرفته دست یافت. بدین‏منظور باید مراحل زیر طی شود.

Graphviz ⇒ File ⇒ Open ⇒ Class Dependencies ⇒



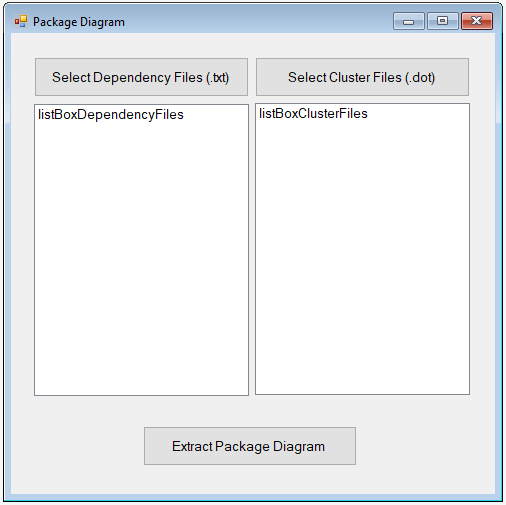
برای مثال، برای برنامه موردنظر، نمایش گرافیکی خوشه‏بندی صورت‏گرفته به‏صورت زیر است.



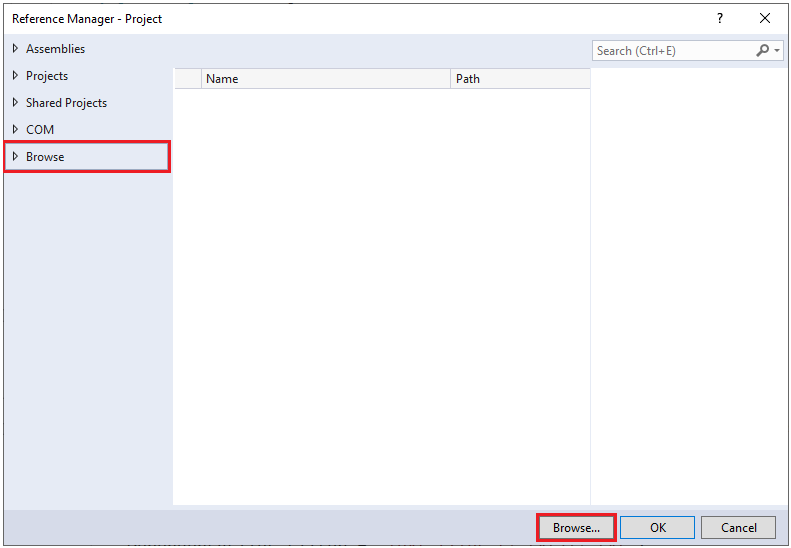
* **نمایش نتیجه خوشه‏بندی در محیط Rational Rose:**

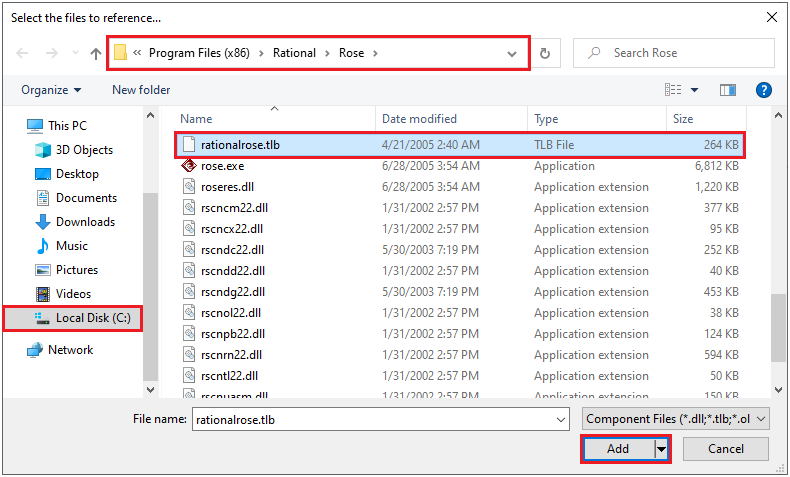
حال نیازمند برنامه‏ای هستیم که فایل‏های .TXT و .DOT تولیدشده را به‏عنوان ورودی گرفته و به‏طور خودکار Package Diagram های مربوطه را در محیط Rational Rose نمایش دهد. بدین‏منظور باید مراحل زیر طی شود.

Visual Studio ⇒ File ⇒ New ⇒ Project ⇒ Visual C# ⇒ Windows Forms App (.NET Framework) ⇒ Neme: Project ⇒ Framework: .NET Framework 4.6.1 ⇒ Ok.



Visual Studio ⇒ Solution Explorer ⇒ References ⇒ Add References ⇒





|  |
| --- |
| using System;  using System.IO;  using System.Linq;  using RationalRose;  using System.Windows.Forms;  using System.Collections.Generic;  namespace Project  {  public partial class Form1 : Form  {  OpenFileDialog DependencyFiles = new OpenFileDialog();  OpenFileDialog ClusterFiles = new OpenFileDialog();  DialogResult DP\_result;  DialogResult CL\_result;  string[] DepFiles;  string[] CluFiles;  string[] PackageNames;  int PackageNumber = 0;  List<string> PackageList = new List<string>();  List<string> ClassList = new List<string>();  List<RoseCategory> CategoryList = new List<RoseCategory>();  Dictionary<string, RoseCategory> CategoryContainer = new Dictionary<string, RoseCategory>();  int[,] CategoryRelation;  List<RoseSubsystem> SubSystemList = new List<RoseSubsystem>();  int[,] SubSystemRelation;  public Form()  {  InitializeComponent();  }  private void buttonSelectDependencyFiles\_Click(object sender, EventArgs e)  {  listBoxDependencyFiles.Items.Clear();  DependencyFiles.Title = "Select Dependency File(s)";  DependencyFiles.Multiselect = true;  DependencyFiles.Filter = "Text files (\*.txt)|\*.txt";  DP\_result = DependencyFiles.ShowDialog();  if (DP\_result == DialogResult.Cancel) return;  foreach (string filename in DependencyFiles.FileNames)  {  listBoxDependencyFiles.Items.Add(filename);  }  DepFiles = DependencyFiles.FileNames;  }  private void buttonSelectClusterFiles\_Click(object sender, EventArgs e)  {  listBoxClusterFiles.Items.Clear();  ClusterFiles.Title = "Select Cluster File(s)";  ClusterFiles.Multiselect = true;  ClusterFiles.Filter = "Dotty files (\*.dot)|\*.dot";  CL\_result = ClusterFiles.ShowDialog();  if (CL\_result == DialogResult.Cancel) return;  int i = 0;  foreach (string filename in ClusterFiles.FileNames)  {  listBoxClusterFiles.Items.Add(filename);  i++;  }  PackageNumber = i;  CluFiles = ClusterFiles.FileNames;  PackageNames = ClusterFiles.SafeFileNames;  }  private void buttonExtractPackageDiagram\_Click(object sender, EventArgs e)  {  for (int i = 0; i < PackageNumber; i++)  {  string[] parts = PackageNames[i].Split('.');  PackageNames[i] = parts[0];  }  CategoryRelation = new int[PackageNumber, PackageNumber];  SubSystemRelation = new int[PackageNumber, PackageNumber];  RoseApplication rapp = new RoseApplication();  RoseModel rmdl = rapp.NewModel();  RoseClassDiagramCollection RCDC = rmdl.RootCategory.ClassDiagrams;  RoseClassDiagram MainClassDiagram = RCDC.GetFirst("Main");  MainClassDiagram.Visible = true;  RoseModuleDiagramCollection RMDC = rmdl.RootSubsystem.ModuleDiagrams;  RoseModuleDiagram MainModuleDiagram = RMDC.GetFirst("Main");  MainModuleDiagram.Visible = true;  short xPos = 0;  short yPos = 100;  short rowCounter = 1;  for (int j = 0; j < PackageNumber; j++)  {  RoseCategory ClassPackage = rmdl.RootCategory.AddCategory(PackageNames[j]);  CategoryList.Add(ClassPackage);  CategoryContainer.Add(ClassPackage.Name, ClassPackage);  IRoseClassDiagram ClassDiagram = ClassPackage.AddClassDiagram("Main");  MainClassDiagram.AddCategory(ClassPackage);  RoseSubsystem ModulePackage = rmdl.RootSubsystem.AddSubsystem(PackageNames[j]);  SubSystemList.Add(ModulePackage);  IRoseModuleDiagram ModuleDiagram = ModulePackage.AddModuleDiagram("Main");  RoseSubsystemView MainRSSV = MainModuleDiagram.AddSubsystemView(ModulePackage);  MainRSSV.Width = 300;  if (j == 1)  {  xPos = (short)200;  yPos = (short)200;  }  else if (j % 3 == 0)  {  xPos = (short)200;  yPos = (short)((rowCounter) \* 500);  if (j != 1) rowCounter++;  }  else  {  xPos = (short)(xPos + 700);  yPos = (short)(yPos + 50);  }  MainRSSV.XPosition = xPos;  MainRSSV.YPosition = yPos;  PackageModeling(DepFiles[j], CluFiles[j], ClassPackage, ClassDiagram, ModulePackage, ModuleDiagram);  }  for (int i = 0; i < ClassList.Count; i++)  {  string CName = ClassList[i];  string SPName = PackageList[i];  List<string> results = ClassList.FindAll(item => item == CName);  if (results.Count > 1)  {  int SearchIndex = 0;  for (int j = 0; j < results.Count; j++)  {  int FindIndex = ClassList.FindIndex(SearchIndex, item => item == CName);  string DPName = PackageList[FindIndex];  if (FindIndex > i && SPName != DPName)  {  RoseCategory sCat = CategoryContainer[SPName];  RoseCategory dCat = CategoryContainer[DPName];  int sCatIndex = CategoryList.IndexOf(sCat);  int dCatIndex = CategoryList.IndexOf(dCat);  CategoryRelation[sCatIndex, dCatIndex]++;  }  SearchIndex = FindIndex + 1;  }  }  }  RoseCategory SCategory;  RoseCategory DCategory;  RoseCategoryDependency rCatDep;  for (int si = 0; si < PackageNumber; si++)  {  SCategory = CategoryList[si];  for (int di = 0; di < PackageNumber; di++)  {  DCategory = CategoryList[di];  if (si != di)  {  if (CategoryRelation[si, di] > 0)  {  string RelationAmount = Convert.ToString(CategoryRelation[si, di]);  rCatDep = SCategory.AddCategoryDependency(RelationAmount, DCategory.Name);  MainClassDiagram.AddRelationView((RoseRelation)rCatDep);  }  }  }  }  }  private void PackageModeling(string DepFile, string ClusterFile, RoseCategory ClassPackage, IRoseClassDiagram ClassDiagram, RoseSubsystem ModulePackage, IRoseModuleDiagram ModuleDiagram)  {  Dictionary<string, string> classComp = new Dictionary<string, string>();  try  {  string[] depLines = File.ReadAllLines(DepFile);  string[] clusLines = File.ReadAllLines(ClusterFile);  RoseModule theModule;  RoseComponentView rComView;  Dictionary<string, RoseModule> catContainer = new Dictionary<string, RoseModule>();  Dictionary<string, RoseClass> classContainer = new Dictionary<string, RoseClass>();  int clusCounter = 0;  int i = 0;  short xPos = 0;  short yPos = 100;  short rowCounter = 1;  List<RoseModule> modList = new List<RoseModule>();  RoseClass theClass;  while (i < clusLines.Count())  {  if (clusLines[i].StartsWith("subgraph"))  {  clusCounter++;  i++;  if (clusLines[i].StartsWith("label"))  {  string[] parts1 = clusLines[i].Split(':');  string[] parts2 = parts1[1].Split('"');  string label = parts2[0];  theModule = ModulePackage.AddModule(label);  }  else  {  theModule = ModulePackage.AddModule("comp" + clusCounter);  }  theModule.Stereotype = "library";  modList.Add(theModule);  rComView = ModuleDiagram.AddComponentView(theModule);  rComView.Width = 500;  if (clusCounter == 1)  {  xPos = (short)200;  yPos = (short)200;  }  else if (clusCounter % 3 == 0)  {  xPos = (short)200;  yPos = (short)((rowCounter) \* 500);  if (clusCounter != 1) rowCounter++;  }  else  {  xPos = (short)(xPos + 700);  yPos = (short)(yPos + 50);  }  rComView.XPosition = xPos;  rComView.YPosition = yPos;  catContainer.Add(theModule.Name, theModule);  i++;  while (!clusLines[i].StartsWith("}"))  {  if (clusLines[i].StartsWith("\""))  {  RoseClass rClass = new RoseClass();  string[] parts = clusLines[i].Split('"');  rClass.Name = parts[1];  classComp.Add(rClass.Name, theModule.Name);  theClass = ClassPackage.AddClass(parts[1]);  rClass.AddAssignedModule(theModule);  ClassDiagram.AddClass(theClass);  classContainer.Add(rClass.Name, theClass);  PackageList.Add(ClassPackage.Name);  ClassList.Add(rClass.Name);  }  i++;  }  }  i++;  }  i = 0;  int[,] m = new int[clusCounter, clusCounter];  int sModIndex;  int dModIndex;  while (i < depLines.Count())  {  string[] dep = depLines[i].Split(' ');  string cls1 = dep[0];  string cls2 = dep[1];  String classDep = dep[2];  int depVal = Convert.ToInt32(dep[2]);  RoseClass sClass = classContainer[cls1];  RoseClassDependency rClsDep = sClass.AddClassDependency(classDep, cls2);  ClassDiagram.AddRelationView((RoseRelation)rClsDep);  string sourceCom = classComp[cls1];  string desCom = classComp[cls2];  RoseModule sMod = catContainer[sourceCom];  RoseModule dMod = catContainer[desCom];  sModIndex = modList.IndexOf(sMod);  dModIndex = modList.IndexOf(dMod);  m[sModIndex, dModIndex] = m[sModIndex, dModIndex] + depVal;  i++;  }  RoseModule sModule;  RoseModule dModule;  RoseModuleVisibilityRelationship rModRel;  for (int si = 0; si < clusCounter; si++)  {  sModule = modList[si];  for (int di = 0; di < clusCounter; di++)  {  dModule = modList[di];  if (si != di)  {  if (m[si, di] > 0)  {  rModRel = sModule.AddVisibilityRelationship(dModule);  rModRel.Name = Convert.ToString(m[si, di]);  ModuleDiagram.AddRelationView((RoseRelation)rModRel);  }  }  }  }  }  catch (Exception) { }  }  }  } |

برای مثال، برای برنامه موردنظر، خواهیم داشت:

