

Examination #2

Name: _____ Instructor's Solution _____ SUID: _____

This is a closed book examination. Please place all your books on the floor beside you. You may keep one page of notes on your desktop in addition to this exam package. All examinations will be collected promptly at the end of the class period. Please be prepared to quickly hand in your examination at that time.

If you have any questions, please do not leave your seat. Raise your hand and I will come to your desk to discuss your question. I will answer all questions about the meaning of the wording of any question. I may choose not to answer other questions.

You will find it helpful to review all questions before beginning. All questions are given equal weight for grading, but not all questions have the same difficulty. Therefore, it is very much to your advantage to answer first those questions you believe to be easiest.

1. Describe the structure of a WPF program. You may wish to do this by describing the purpose of each file in a WPF application.

Answer:

A WPF program is composed of Xaml markup and C# code. The WPF wizard generates four files: App.xaml, App.xaml.cs, MainWindow.xaml, and MainWindow.xaml.cs. For the MainWindow Xaml describes its window's artifacts and their connectivity, e.g., panels and graphical objects like Buttons and TextBlocks. The C# code implements event handlers for user actions in the Window. The two App files compile into a message loop, dispatching mechanisms, and startup code to support the main windows activities.

Most significant applications will have other packages that handle the applications processing details, possibly running on child threads. Those threads communicate with the applications main thread using the App's Dispatcher object via a call to Dispatcher.invoke.

When the WPF application is compiled the Xaml code is translated into C# source which is then compiled, along with the developer defined C# code, into MSIL which runs in the .Net virtual machine. An application can replace all the Xaml with C# code. In fact this is the only way to build a WPF application using C++ in Visual Studio 2010 because the C++ team had not implemented the Xaml translator.

2. Suppose that you have been asked to describe the Software Architecture of your Project #2. What will you discuss?

Answer:

You need to discuss all the expected parts of an OCD for Project #2, e.g., uses and user groups, project package structure, responsibilities and interactions of each of the packages, and any critical issues faced by the development team.

Often you will sketch user interfaces, discuss any interesting data structures and the sharing of data across the application, and document any algorithms or unique design ideas that will make the developed code flexible and robust. The Parser is a good example.

Specific to project #2 you should discuss the adequacy of the distance metric (see my solution for MT1Q7), and the need to gracefully handle small projects for individual developers and very large sets of files that make up a project baseline for Quality Assurance analysis.

3. In C# what is the difference between a value type and a reference type. Give an example of each. Describe the syntax and semantics of passing a reference type by value.

Answer:

A value type, an int for example, resides in stack or static memory. Its name is a label for that memory location. It can be copied and assigned which results in the value being copied or assigned to the target location and that is independent of the source value once copied or assigned.

A reference type, an instance of any C# class for example, resides in the managed heap and is accessed via its reference stored in stack or static memory. References can be copied and assigned, but that does not copy or assign the object residing in the managed heap. Thus, these copies and assignments are shallow.

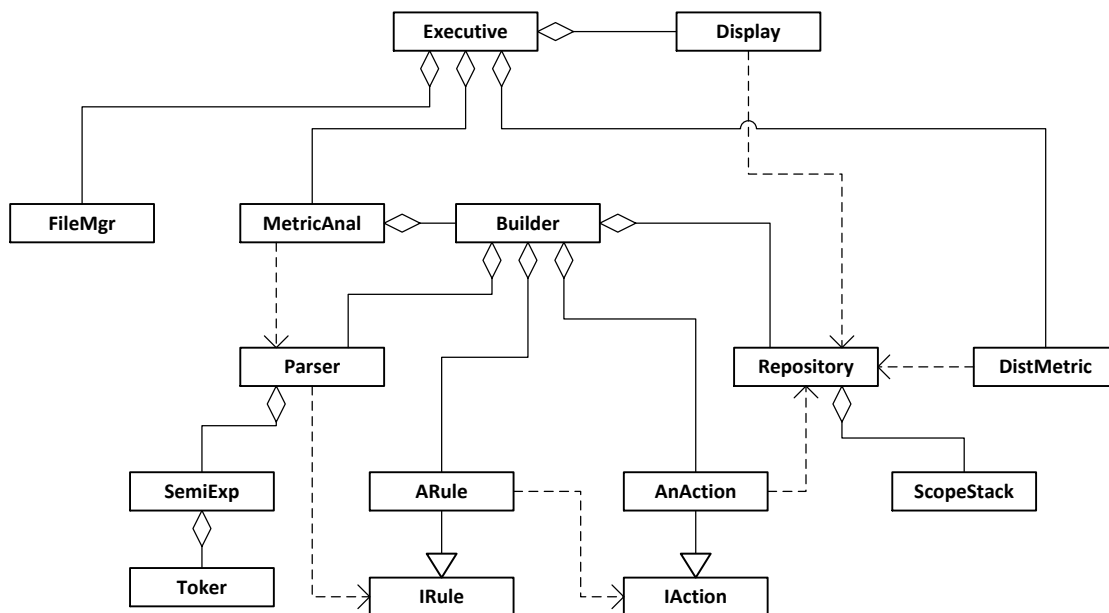
Passing an instance of a reference type by value means that you call a function with the reference argument. That copies the reference to the stack frame of the called function. Any change to the state of the referenced object by the function will use the copied reference and the caller will see the result.

4. Draw a class diagram for your design of Project #2.

Answer:

In the diagram below we only show one rule and one action class. The application will have several of these, not shown to keep the diagram simple enough to understand readily.

The Executive class manages program flow. Parser and all its related rules, actions, and tokenizing classes manage the parsing of code to evaluate the size and cyclomatic complexity for each function. The Repository holds intermediate analysis results for later processing and display.



5. Write code to programmatically create an XML representation of the information needed to capture the quality output of Project #2, e.g., file name, each function name, cyclomatic complexity, and size.

Answer:

```
<?xml version="1.0" encoding="utf-8" ?>
<QualOut>
  <file>
    <name>file1.cs</name>
    <function>
      <name>fun1</name>
      <cc>8</cc>
      <size>23</size>
    </function>
    <function>
      <name>fun2</name>
      <cc>5</cc>
      <size>42</size>
    </function>
  </file>
  <file>
    <name>file2.cs</name>
    <function>
      <name>fun3</name>
      <cc>9</cc>
      <size>48</size>
    </function>
  </file>
</QualOut>

class Program
{
    static public XElement createFunctionElement(string name, int cc, int size)
    {
        XElement fname = new XElement("name", name);
        XElement fcc = new XElement("cc", cc.ToString());
        XElement fsize = new XElement("size", size.ToString());
        XElement function = new XElement("function");
        function.Add(fname);
        function.Add(fcc);
        function.Add(fsize);
        return function;
    }
    static public XElement createFileElement(string name)
    {
        XElement fname = new XElement("name", name);
        XElement file = new XElement("file");
        file.Add(fname);
        return file;
    }
    static void Main(string[] args)
    {
        XElement qout = new XElement("QualOut");
        XElement file1 = createFileElement("file1.cs");
        XElement fun1 = createFunctionElement("fun1", 8, 23);
        XElement fun2 = createFunctionElement("fun2", 5, 42);
        file1.Add(fun1);
        file1.Add(fun2);
        qout.Add(file1);
        XDocument doc = new XDocument();
        doc.Add(qout);
        Console.WriteLine("\n{0}", doc.ToString());
        Console.WriteLine("\n\n");
    }
}
```

6. Write a service contract for a communication channel that sends instances of a Message class to a remote machine.

Answer:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;

namespace CommunicationPrototype
{
    [ServiceContract]
    public interface ICommService
    {

        [OperationContract(IsOneWay=true)]
        void PostMessage(Message msg);

        // Not a service operation so only server can call
        Message GetMessage();
    }

    [DataContract]
    public class Message
    {
        [DataMember]
        Content cnt = QualData;
        [DataMember]
        string body = "<? Xml version='1.0' ?><QualOut></QualOut>";

        public enum Content
        {
            [EnumMember]
            QualData,
            [EnumMember]
            DistanceData
        }

        [DataMember]
        public Content content
        {
            get { return cnt; }
            set { cnt = value; }
        }

        [DataMember]
        public string XmlText
        {
            get { return body; }
            set { body = value; }
        }
    }
}
```

7. Write all the code for a Display package that writes Project #2¹ quality data² to the console.

Answer:

```
public class Display
{
    XDocument doc;

    public Display(string path)
    {
        doc = XDocument.Load(path);
    }
    public void Write()
    {
        var files = from e in
                        doc.Elements("QualOut")
                        .Elements("file")
                        select e;

        foreach (var file in files)
        {
            var q2 = from e in file.Elements("name") select e;
            Console.WriteLine("\n {0}", q2.ElementAt(0).Value);
            var q3 = from e in file.Elements("function").Elements() select e;
            int numFuncs = q3.Count()/3;
            for (int i = 0; i < numFuncs; ++i)
            {
                Console.WriteLine("\n {0}", q3.ElementAt(3*i).Value); // name
                Console.WriteLine("\n {0}", q3.ElementAt(3*i+1).Value); // cc
                Console.WriteLine("\n {0}", q3.ElementAt(3*i+2).Value); // size
            }
        }
        Console.WriteLine("\n\n");
    }
}
class Program
{
    static void Main(string[] args)
    {
        Display dsp1 = new Display(@"..\..\Qual.xml");
        dsp1.Write();
    }
}
```

Note: Using same XML as in MT2Q5.

¹ You may wish to write the relevant class declarations and come back to fill in methods as time permits.

² See Question #1 for definition.