A MODEL OF DEFECTS DETECTION IN OBJECT ORIENTED PROGRAMMING LANGUAGES AND CONSOLE BASED APPLICATIONS

Final Report Submitted to

UGC-MRP

New Delhi

By

Dr. S. SARALA, MCA.,Ph.D

Assistant Professor

NOVEMBER 2010

BHARATHIAR UNIVERSITY

COIMBATORE 641 046

REPORT

A Model of Defects Detection in Object Oriented Programming Languages and Console Based Applications

1. Introduction:

The software developers tackled several million lines of complex C++ code, Java code and C# code. The code might be an internally developed as a language with JAVA library, exe files and all supporting files to execute the applications. Although the software or language works under the programmer's intention what they will be? The code has been demonstrated in the testing; we found that there are a large number of logical defects occur in the software. But how can we find them before deliver the software. Hence to solve this problem the mini-model has been helpful to find the defects, localizing the defects with the minimal period of time and low cost of handling this type of defects.

In this report, I describe two techniques that will quickly identify logical defects without doing extensive testing. Instead, these techniques are complementing of the normal testing efforts which are trying. I demonstrated this model with thousands and thousands of logical defects in a very complex software system containing several million lines of C++ code. The model is very confident to detect the defects in JAVA code and C# code.

1.1. Logical Defects

It is a defect that exists in C# and Java applications, but has not been discovered by the testing process. In a very large software system, it is not possible to test all possible execution paths through the source code, to identify these defects we need billions of test cases to cover every combination of conditional branches in that system. Many untested execution paths may contain serious coding errors like logical defects and abnormal termination. The main challenge is to identify the logical defects before the customers find them as a major or serious defect. A large percentage of the defects I uncovered were in various error paths that had not received proper test coverage during previous test cycles. Obviously, correcting defects early in the design

cycle will save time and cost, so identifying as many defects as possible before the software is delivered to the market or customer. The detected logical defects in the object oriented language ie., JAVA and C# fell into few broad categories with exercised problems are listed below.

1.2 System Oriented JAVA Exceptions

The developed model detects the following defects in the given parameters. In order to express the context of object oriented languages such as C++, Java has its own compilers.

1. **Interrupted Exception**

```
Class Sample
{
   Public static void main (String args[])
{
          Try
          {
          Thread.sleep(500); // make the thread to sleep for 500 milli secs
          }
          catch ( InterruptedException e )
          {
          // Exception block to execute when the user disturb the processor in the sleeping
               time…
          System.out.println("Interrrupted Exception"+e);
          }
    }
}
```

OUTPUT:

```
java.lang.InterruptedException
at java.lang.Object.wait(Native Method)
at java.lang.Object.wait(Object.java:474)
at EDU.oswego.cs.dl.util.concurrent.Semaphore.acquire(Semaphore.java:108)
at
EDU.oswego.cs.dl.util.concurrent.SemaphoreControlledChannel.take(SemaphoreControlled
Channel.java:131)
at org.jboss.resource.adapter.mail.inflow.NewMsgsWorker.run(NewMsgsWorker.java:75)
at org.jboss.resource.work.WorkWrapper.execute(WorkWrapper.java:204)
at org.jboss.util.threadpool.BasicTaskWrapper.run(BasicTaskWrapper.java:275)
at EDU.oswego.cs.dl.util.concurrent.PooledExecutor$Worker.run(PooledExecutor.java:756)
at java.lang.Thread.run(Thread.java:595)
```

2. **ClassCastException**

```
Class Sample
{
   Public static void main (String args[])
{
            try
           {  // statement to get the class name
             // Exception occurs when the object is not valid
              out.println(objRef.getClass().getName() + "<br>");
           } catch ( ClassCastException e ) {
             System.out.println("classcast Exception"+e);
           }
    }
}
```

OUTPUT:

java.lang.ClassCastException: java.lang.Boolean cannot be cast to java.lang.String
at CastException.main(Sample.java:6)

3. **RemoteException**

```
Class Sample
{
   Public static void main (String args[])
{
Try
 {  // statement to get the register Id form system registery.
   int groupID = ActivationGroup.getSystem().registerGroup(group);
//FAILS HERE
} catch(RemoteException e) {
  e.printStackTrace();
  System.exit(1);
}
  }
}
```

Output:

java.rmi.activation.ActivationException: ActivationSystem not running; nested exception is:
java.security.AccessControlException: access denied (java.net.SocketPermission 127.0.0.1:1098 connect,resolve)
java.security.AccessControlException: access denied (java.net.SocketPermission 127.0.0.1:1098 connect,resolve)
at

```
java.security.AccessControlContext.checkPermission(AccessControlContext.java:272)
at
java.security.AccessController.checkPermission(AccessController.java:399)
at java.lang.SecurityManager.checkPermission(SecurityManager.java:545)
at java.lang.SecurityManager.checkConnect(SecurityManager.java:1044)
at java.net.Socket.<init>(Socket.java:262)
at java.net.Socket.<init>(Socket.java:100)
at
sun.rmi.transport.proxy.RMIDirectSocketFactory.createSocket(RMIDirectSocketFactory.java:25
)
at
sun.rmi.transport.proxy.RMIMasterSocketFactory.createSocket(RMIMasterSocketFactory.java:1
20)
at sun.rmi.transport.tcp.TCPEndpoint.newSocket(TCPEndpoint.java:499)
at
sun.rmi.transport.tcp.TCPChannel.createConnection(TCPChannel.java:190)
at sun.rmi.transport.tcp.TCPChannel.newConnection(TCPChannel.java:174)
at sun.rmi.server.UnicastRef.newCall(UnicastRef.java:318)
at sun.rmi.registry.RegistryImpl_Stub.lookup(Unknown Source)
at java.rmi.Naming.lookup(Naming.java:84)
at
java.rmi.activation.ActivationGroup.getSystem(ActivationGroup.java:455)
at activ_perform.AService.<init>(AService.java:60)
at activ_perform.AService.main(AService.java:37)
```

4. **ClassIllegalException**

```
import java.awt.TextField;
public class illegal {
public static void main (String[] args)
{ // Accesing illegal function to set the textfield position makes classIllegalException
new TextField().setCaretPosition (24);
System.out.println ("Never gets here");
        }
}
```

OUTPUT:

Exception in thread "main" java.lang.NoClassDefFoundError: illegal/class

5. **SQLException**

```
import java.sql.*;
public class ExceptionExample {
public static Connection connection = null;
public static void main(java.lang.String[] args) {
    try {// statement to connect to the database
        Class.forName("com.ibm.db2.jdbc.app.DB2Driver");
```

```
// Establishing connection through connection string
            connection = DriverManager.getConnection("jdbc:db2:*local");
// creating statement using connection variable.
            Statement s = connection.createStatement();
//execute update statement for executing insert statement.
            int count = s.executeUpdate("insert into cujofake.cujofake values(1, 2,3)");
            System.out.println("Did not expect that table to exist.");
        } catch (SQLException e) {
// Exception block when connection to database ends in exception
            System.out.println("SQLException exception: ");
            System.out.println("Message:....." + e.getMessage());
            System.out.println("SQLState:...." + e.getSQLState());
            System.out.println("Vendor Code:." + e.getErrorCode());
            System.out.println("-----------------------------------------------------");
            e.printStackTrace();
        } catch (Exception ex) {
            System.out.println("An exception other than an SQLException was thrown: ");
            ex.printStackTrace();
        } finally {
            try {
                if (connection != null) {
                    connection.close();
                }
            } catch (SQLException e) {
                System.out.println("Exception caught attempting to shutdown...");
            }
        }
    }
}
```

OUTPUT:

```
 nested exception is:
        java.sql.SQLException: Error reading from InputStream java.io.IOException
        at
org.apache.james.mailrepository.JDBCMailRepository.store(JDBCMailRepository.java:7
55)
        at
org.apache.james.mailrepository.JDBCSpoolRepository.store(JDBCSpoolRepository.jav
a:236)
        at
org.apache.james.transport.mailets.RemoteDelivery.service(RemoteDelivery.java:1001)
        at org.apache.james.transport.LinearProcessor.service(LinearProcessor.java:414)
        at
org.apache.james.transport.JamesSpoolManager.process(ExceptionExample.java:397)
```

```
    at org.apache.james.transport.JamesSpoolManager.run(ExceptionExample.java:306)
    at java.lang.Thread.run(Thread.java:595)
```

6. **FileInputStreamException**

```
import java.io.*;
class Demo1 {
  public static FileInputStream f1(String fileName)
    throws FileNotFoundException
  {
// creating new file input stream
    FileInputStream fis = new FileInputStream(fileName);
    System.out.println("f1: File input stream created");
    return fis;
  }
  public static FileInputStream f2(String fileName)
  {
    FileInputStream fis = null;
    try
    {
//creating new file input stream using existing file fileName
      fis = new FileInputStream(fileName);
    }
    catch (FileNotFoundException ex)
    {
// Exception occurs when filename not exists
      System.out.println("f2: Oops, FileNotFoundException caught");
    }
    finally
    {
      System.out.println("f2: finally block");
    }
    System.out.println("f2: Returning from f2");
    return fis;
  }
  public static void main(String args[])
  {
    FileInputStream fis1 = null;
    FileInputStream fis2 = null;
    String fileName = "foo.bar";
    // String fileName = null;
    System.out.println(  "main: Starting " + Demo1.class.getName()
                  + " with file name = " + fileName);
    // get file input stream 1
    try {
      fis1 = f1(fileName);
    }
```

```
            catch (FileNotFoundException ex)
            {
              System.out.println("main: Oops, FileNotFoundException caught");
            }
            catch (Exception ex)
            {
              System.out.println("main: Oops, genreal exception caught");
            }
            // get file input stream 2
            fis2 = f2(fileName);
            System.out.println("main: " + Demo1.class.getName() + " ended");
          }
        }
```

Output:

```
java.io.FileNotFoundException:
    at java.io.FileInputStream.<init>(FileInputStream.java)
    at java.io.FileInputStream.<init>(FileInputStream.java)
    at Demo1.readMyFile(Demo1.java:19)
    at Demo1.main(Demo1.java:7)
```

7. **IOException**

```
            import java.io.File;
            import java.io.IOException;
            import junit.framework.*;
            public class FileFinderTest extends TestCase {
               private FileFinder fileFinder;
               private FileFixture fileFixture;
               public void testGetSourceDirectoryList() {
// Checking the occurrence of the file
                  assertEquals(4, fileFinder.getSourceDirectoryList().size());
                  assertTrue(new FileFinder().getSourceDirectoryList().isEmpty());
                  FileFinder ff = new FileFinder();
                  ff.addSourceDirectory( FileFixture.SOURCE_DIRECTORY_IDENTIFIER[0]);
            //Adding source directory
                  assertEquals( 1, ff.getSourceDirectoryList().size());
                  ff.addSourceDirectory( FileFixture.SOURCE_DIRECTORY_IDENTIFIER[3]);
                  assertEquals( 2, ff.getSourceDirectoryList().size());
                  ff = new FileFinder();
                  ff.addSourceFile( FileFixture.SOURCE_DIRECTORY_IDENTIFIER[0],
            "com/example/Sample1.java");
                  ff.addSourceFile( FileFixture.SOURCE_DIRECTORY_IDENTIFIER[0],
            "com/example/Sample2.java");
                  assertEquals( 1, ff.getSourceDirectoryList().size());
                  ff.addSourceFile( FileFixture.SOURCE_DIRECTORY_IDENTIFIER[1],
            "com/example/Sample3.java");
```

```
        assertEquals( 2, ff.getSourceDirectoryList().size());
        ff.addSourceDirectory( FileFixture.SOURCE_DIRECTORY_IDENTIFIER[3]);
        assertEquals( 3, ff.getSourceDirectoryList().size());
    }
    private void checkFile( String fileName, String baseName, int sourceNum) throws
IOException {
            File file = fileFinder.getFileForSource( fileName);
            assertTrue( file.getAbsolutePath(),
file.getAbsolutePath().indexOf(FileFixture.SOURCE_DIRECTORY_IDENTIFIER[so
urceNum])!=-1);
            assertTrue( baseName.equals( file.getName()));
            assertTrue( file.exists());
            assertTrue( file.isFile());
    }
    public void testFindFile() throws IOException {
            checkFile("com/example/Sample1.java", "Sample1.java", 0);
            checkFile("com\\example\\Sample2.java", "Sample2.java", 0);
            checkFile("com/example\\Sample3.java", "Sample3.java", 1);
            checkFile("com/example/Sample4.java", "Sample4.java", 1);
            checkFile("com/example/Sample5.java", "Sample5.java", 2);
            checkFile("com/example/Sample6.java", "Sample6.java", 2);
            checkFile("com\\example/Sample7.java", "Sample7.java", 3);
    }
    public void testFindFile_NotFound() {
            try {
// Checking file source for the fileFinder
                    fileFinder.getFileForSource("com/example/Sample19.java");
                    fail( "IOException expected");
            } catch( IOException ex) {}
            try {
                    fileFinder.getFileForSource("com/example/Sample1.jav");
                    fail( "IOException expected");
            } catch( IOException ex) {}
            try {
                    fileFinder.getFileForSource("com/example/Sample7.java2");
                    fail( "IOException expected");
            } catch( IOException ex) {}
            try {
                    fileFinder.getFileForSource("Sample3.java");
                    fail( "IOException expected");
            } catch( IOException ex) {}
            try {
                    // This file exist, but is not added to fileFinder
                    fileFinder.getFileForSource("com/example/Sample8.java");
                    fail( "IOException expected");
            } catch( IOException ex) {}
```

```
    }
    public void testFindFile_null() throws IOException {
            try {
                    fileFinder.getFileForSource(null);
                    fail( "NullPointerException expected");
            } catch( NullPointerException ex) {}
    }
    public void testAddSourceDirectory_null() {
            try {// Adding source directory
                    fileFinder.addSourceDirectory(null);
                    fail( "NullPointerException expected");
            } catch( NullPointerException ex) {}
    }
    public void testAddSourceFile_null() {
            try {
                    fileFinder.addSourceFile(null,"com/example/Sample1.java");
                    fail( "NullPointerException expected");
            } catch( NullPointerException ex) {}
            try {
                    fileFinder.addSourceFile(
FileFixture.SOURCE_DIRECTORY_IDENTIFIER[0], null);
                    fail( "NullPointerException expected");
            } catch( NullPointerException ex) {}
    }

    protected void setUp() throws Exception {
       super.setUp();
       fileFixture = new FileFixture();
       fileFixture.setUp();
fileFinder = new FileFinder();
fileFinder.addSourceDirectory(fileFixture.sourceDirectory(FileFixture.SOURCE_DIR
ECTORY_IDENTIFIER[0]).toString());
fileFinder.addSourceDirectory(fileFixture.sourceDirectory(FileFixture.SOURCE_DIR
ECTORY_IDENTIFIER[1]).toString());
       fileFinder.addSourceFile(
fileFixture.sourceDirectory(FileFixture.SOURCE_DIRECTORY_IDENTIFIER[2]).to
String(), "com/example\\Sample5.java");
       fileFinder.addSourceFile(
fileFixture.sourceDirectory(FileFixture.SOURCE_DIRECTORY_IDENTIFIER[2]).to
String(), "com/example/Sample6.java");
       fileFinder.addSourceFile(
fileFixture.sourceDirectory(FileFixture.SOURCE_DIRECTORY_IDENTIFIER[3]).to
String(), "com/example/Sample7.java");
       // Do not add com/example/Sample8.java
```

```
        // fileFinder.addSourceFile(
    fileFixture.sourceDirectory(FileFixture.SOURCE_DIRECTORY_IDENTIFIER[3]).to
    String(), "com/example/Sample8.java");
      }
      protected void tearDown() throws Exception {
        super.tearDown();
        fileFixture.tearDown();
      }
    }
```

OUTPUT:

```
Exception in thread "main" java.io.IOException: Cannot run program "ls": java.io.IOException:
error=12, Cannot allocate memory
        at java.lang.ProcessBuilder.start(ProcessBuilder.java:474)
        at java.lang.Runtime.exec(Runtime.java:610)
        at java.lang.Runtime.exec(Runtime.java:448)
        at java.lang.Runtime.exec(Runtime.java:345)
        at prova.main(prova.java:6)
Caused by: java.io.IOException: java.io.IOException: error=12, Cannot allocate memory
        at java.lang.UNIXProcess.<init>(UNIXProcess.java:164)
        at java.lang.ProcessImpl.start(ProcessImpl.java:81)
        at java.lang.ProcessBuilder.start(ProcessBuilder.java:467)
        ... 4 more
```

8. NumberFormat Exception

```
    public class ConvertStringToNumber
  {

    public static void main(String[] args)
    {
      try
      {
  // String s value to be initialized to integer makes exception
        String s = "FOOBAR";
        int i = Integer.parseInt(s);
        // this line of code will never be reached
        System.out.println("int value = " + i);
      }
      catch (NumberFormatException nfe)
      {
        nfe.printStackTrace();
      }
    }
  }
```

OUTPUT:

```
java.lang.NumberFormatException: For input string: "FOOBAR"
  at java.lang.NumberFormatException.forInputString(NumberFormatException.java:48)
  at java.lang.Integer.parseInt(Integer.java:447)
  at java.lang.Integer.parseInt(Integer.java:497)
  at
com.devdaily.javasamples.ConvertStringToNumber.main(ConvertStringToNumber.java:
11)
```

9. NullPointException

```
class Main {
  String s;
   void initializeStuff() {
      String s = "aaa";
   }
       void displayStuff() {
// Value s is not initialized but checking length results in exception
      System.out.print("The length of '"+s+"' is: ");
      System.out.print(s.length());
   }
   public static void main(String[] args) {
      Main main = new Main();
      main.initializeStuff();
      main.displayStuff();
   }
}
```

OUTPUT:

```
Exception in thread "main" java.lang.NullPointerException
at queueTask.Main.main(Main.java:53)
```

10. DataFormatException

```
package com.citi.blob;
import java.sql.*;
import java.io.*;
import java.util.zip.Inflater;
public class PullBlob {
public static void main(String[] args) {
// DB Connection info
String userName="user";
String passWord="password";
//Production Database
String dataBase="ABCD";
//Production URL
```

```
String url = "jdbc:oracle:thin:...111.2222.333.444:8080:" + dataBase;
System.out.println("args length is" + args.length);
// Sort out args
if(args.length<2) {
System.out.println("Usage: java PullBLOB product acctnum optional-out-file");
System.exit(1);
}
String product = args[0];
String acctnum = args[1];
String fileName = null;
if(args.length>=3) {
fileName = args[2];
}
Connection con = null;
try {
// Establish connection
Class.forName("oracle.jdbc.driver.OracleDriver");
con = DriverManager.getConnection(url, userName, passWord);
Statement stmt = con.createStatement();
// Sort out the output stream
boolean toFile=false;
PrintWriter fout = null;
if(fileName!=null) {
File file = new File(fileName);
FileOutputStream fos = new FileOutputStream(file);
fout = new PrintWriter(fos);
toFile = true;
}
String query = "SELECT stm_cmp_xml FROM pld_stm WHERE prd_nm =
'JACKS_BOX' and act_num='9988556622'";
System.out.println(query);
ResultSet rs = stmt.executeQuery(query);
// System.out.println("Number of records fetchted " + rs.getFetchSize());
if(rs.next()) {
//Blob xml = rs.getBlob("inv_cmp_xml");
Blob xml = rs.getBlob("stm_cmp_xml");
InputStream in = xml.getBinaryStream();
ByteArrayOutputStream byteStream = new ByteArrayOutputStream();
int data;
while ((data = in.read()) != -1) {
byteStream.write((byte)data);
}
byte[] compressedData = byteStream.toByteArray();
in.close();
Inflater decompressor = new Inflater();
decompressor.setInput(compressedData);
```

```
// Create an expandable byte array to hold the decompressed data
ByteArrayOutputStream bos = new ByteArrayOutputStream(compressedData.length);
// Decompress the data
//byte[] buf = new byte[TEMP_BUFFER_SIZE];
// use a 1 meg buffer for improved speed
byte [] buf = new byte[1048576];
while (!decompressor.finished()) {
int count = decompressor.inflate(buf);
bos.write(buf, 0, count);
}
buf=null;
bos.close();
//Get the decompressed data
byte[] decompressedData = bos.toByteArray();
String stmtXml = new String(decompressedData).trim();
if(toFile) {
fout.println(stmtXml);
}
else {
System.out.println(stmtXml);
}
}
else { // No result
System.out.println("Couldn't find stmt for " + acctnum );
}
if(fout!=null) fout.close();
}
catch(Exception e) {
System.out.println(e.getMessage());
e.printStackTrace();
}
finally {
try {
if(con!=null) con.close();
}
catch(Exception e) {
// Whatever
}}}}
```

OUTPUT:

```
args length is3
SELECT stm_cmp_xml FROM pld_stm WHERE prd_nm = 'JACKS_BOX' and
act_num='9988556622'
unknown compression method
java.util.zip.DataFormatException: unknown compression method
```

```
at java.util.zip.Inflater.inflateBytes(Native Method)
at java.util.zip.Inflater.inflate(Unknown Source)
at java.util.zip.Inflater.inflate(Unknown Source)
at com.citi.blob.PullBlob.main(PullBlob.java:95)
```

1.3 C-Sharp Exceptions

1. **FileNotFoundException**

```
using System;
  using System.IO;
  class tryCatchDemo
  {
   // Class definition
     static void Main(string[] args)
     {
        try
        {
  //  File “NonExistingFile” to read the content
           File.OpenRead("NonExistentFile");
        }
        catch(FileNotFoundException fnfex)
        {
  //Exception block executes when the file “NonExistingFile” is not present in  the
  //location specified.
           Console.WriteLine(fnfex.ToString());
        }
     }
  }
```

OUTPUT:

```
System.IO.FileNotFoundException was unhandled
Message: It's not possible to load the file or
assembly SomeName, Version=1.0.0.0,
Culture=neutral, PublicKeyToken=null or one of
its dependencies.The file cannot be found.
```

2. **CaughtSecurityException**

```
using System;
  using System.Security;
  using System.Security.Permissions;
  using System.Runtime.InteropServices;
  class NativeMethods
  {   // This is a call to unmanaged code. Executing this method requires
     // the UnmanagedCode security permission. Without this permission
     // an attempt to call this method will throw a SecurityException:
     [DllImport("msvcrt.dll")]
```

```
    public static extern int puts(string str);
    [DllImport("msvcrt.dll")]
    internal static extern int _flushall();
}
class MainClass
{
    private static void CallUnmanagedCodeWithoutPermission()
    {
        // Create a security permission object to describe the
        // UnmanagedCode permission:
        SecurityPermission perm =
          new SecurityPermission(SecurityPermissionFlag.UnmanagedCode);
        // Deny the UnmanagedCode from our current set of permissions.
        // Any method that is called on this thread until this method
        // returns will be denied access to unmanaged code.
        // Even though the CallUnmanagedCodeWithPermission method
        // is called from a stack frame that already
        // calls Assert for unmanaged code, you still cannot call native
        // code. Because you use Deny here, the permission gets
        // overwritten.
        perm.Deny();
        try
        {
            Console.WriteLine("Attempting to call unmanaged code without permission.");
            NativeMethods.puts("Hello World!");
            NativeMethods._flushall();
            Console.WriteLine("Called unmanaged code without permission. Whoops!");
        }
        catch (SecurityException)
        {
            Console.WriteLine("Caught Security Exception attempting to call unmanaged
code.");
        }
    }
    private static void CallUnmanagedCodeWithPermission()
    {
        // Create a security permission object to describe the
        // UnmanagedCode permission:
        SecurityPermission perm =
          new SecurityPermission(SecurityPermissionFlag.UnmanagedCode);
        // Check that you have permission to access unmanaged code.
        // If you don't have permission to access unmanaged code, then
        // this call will throw a SecurityException.
        // Even though the CallUnmanagedCodeWithPermission method
        // is called from a stack frame that already
        // calls Assert for unmanaged code, you still cannot call native
```

```
        // code. Because you use Deny here, the permission gets
        // overwritten.
        perm.Assert();
        try
        {
            Console.WriteLine("Attempting to call unmanaged code with permission.");
            NativeMethods.puts("Hello World!");
            NativeMethods._flushall();
            Console.WriteLine("Called unmanaged code with permission.");
        }
        catch (SecurityException)
        {
            Console.WriteLine("Caught Security Exception attempting to call unmanaged
code. Whoops!");
        }
    }
    public static void Main()
    {
        // The method itself will call the security permission Deny
        // for unmanaged code, which will override the Assert permission
        // in this stack frame.
        SecurityPermission perm = new
            SecurityPermission(SecurityPermissionFlag.UnmanagedCode);
        perm.Assert();
        CallUnmanagedCodeWithoutPermission();
        // The method itself will call the security permission Assert
        // for unmanaged code, which will override the Deny permission in
        // this stack frame.
        perm.Deny();
        CallUnmanagedCodeWithPermission();
    }
}
```

OUTPUT:

Copy

```
Attempting to call unmanaged code without permission.
Caught Security Exception attempting to call unmanaged code.
Attempting to call unmanaged code with permission.
Hello World!
Called unmanaged code with permission.
```

3. **ArgumentNullException**

```
using System;
        class Program
        {
            static void Main()
```

```
        {
            // Demonstrate the argument null exception.
            try
            {
                A(null);
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex);
            }
            // Demonstrate the general argument exception.
            try
            {
                A("");
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex);
            }
            // Flow path without exception.
            Console.WriteLine(A("test"));
        }
        static int A(string argument)
        {
            // Handle null argument.
            if (argument == null)
            {
                throw new ArgumentNullException("argument");
            }
            // Handle invalid argument.
            if (argument.Length == 0)
            {
                throw new ArgumentException("Zero-length string invalid", "argument");
            }
            return argument.Length;
        }
    }
```

OUTPUT:

```
    Output truncated.
System.ArgumentNullException: Value cannot be null.
Parameter name: argument
at Program.A(String argument) in...
System.ArgumentException: Zero-length string invalid
Parameter name: argument
at Program.A(String argument) in...
```

4. **DivideByZeroException**

```
using System;

class Program
{
   static void Main()
   {
// The following statement results in Exception due to number divide by '0'
      int result = 100 / int.Parse("0");
      Console.WriteLine(result);
   }
}
```

OUTPUT:

Unhandled Exception: **System.DivideByZeroException**: Attempted to divide by zero.
at Program.Main() in ... Program.cs:line 11

5. **NotSupportedException**

```
class ArrayAddition
{
  public static void Main()
  {
// Declaring String
    string[] strings = {"1", "2"};
    try
    {
// Adding new string to the list
      AddToList (strings, "3");
    }
    catch (NotSupportedException e)
    {
//Exception occurs when the added object is not supported to the list
      Console.WriteLine ("Exception: {0}", e);
    }
  }
  public static void AddToList(IList il, object o)
  {
// Method to add the object to the list
    il.Add(o);
  }
}
```

OUTPUT:

NotSupportedException example (program output)
Exception: System.NotSupportedException: Collection was of a fixed size.
at System.Array.System.Collections.IList.Add(Object value)

at ArrayAddition.AddToList(IList il, Object o) in Program.cs:line 17
at ArrayAddition.Main() in Program.cs:line 9

6. PathTooLongException

```
private void AddFilesToList(DirectoryInfo dir)
            {
// declaring files and retrieving it from the directory
FileInfo[] files = dir.GetFiles();
// creating directories
DirectoryInfo[] subdirs = dir.GetDirectories();
foreach(FileInfo fi in files)
{
// From files f1
try
{
// checking the length of the content in the file selected
if(fi.FullName.Length >
Convert.ToInt32(txtFileLengthCutoff.Text))
{
Debug.WriteLine("try file: " + fi.FullName);
// adding long path file to the longfiles
longFiles.Add(fi.FullName); // this is the arraylist
}
}
catch (System.IO.PathTooLongException e)
{
// Exception block to catch the PathTooLong Exception
Debug.WriteLine("catch file: " + fi.FullName);
// string temp = fi.DirectoryName;
// temp = fi.FullName;
longFiles.Add(fi.FullName);
}
}
foreach (DirectoryInfo di in subdirs)
{
//adding directory to directory list
AddFilesToList(di);
}
}
```

OUTPUT:

System.IO.PathTooLongException was unhandled
The specified path, file name, or both are too long. The fully qualified file name must be less than 260 characters, and the Directory name must be less than 248 characters."

7. BadImageFormatException

```
static void Main(string[] args)
{
string[] strFiles =
System.IO.Directory.GetFiles(@"C:\Temp");
System.Collections.ArrayList objSuccesses =
            new System.Collections.ArrayList();
System.Collections.ArrayList objFailures =
            new System.Collections.ArrayList();
foreach (string strFile in strFiles)
{
try
{
 // We're simply copying a file here.
// In a real application,
//something more complicated would happen...
System.IO.File.Copy(strFile, strFile + ".new");
objSuccesses.Add(strFile);
 }
catch (Exception ex)
{
Console.WriteLine("Failed to copy the file: " +
                        ex.Message);
objFailures.Add(strFile);
}
}
// We could send an e-mail here informing an
// administrator of all failures...
```

OUTPUT:

```
System.BadImageFormatException was unhandled
  Message=" is not a valid Win32 application. (Exception from HRESULT:
0x800700C1)"
  Source="Something"
  StackTrace:
      at Something.Program.Main()
      at System.AppDomain.nExecuteAssembly(Assembly assembly, String[] args)
      at Microsoft.VisualStudio.HostingProcess.HostProc.RunUsersAssembly()
      at System.Threading.ExecutionContext.Run(ExecutionContext executionContext,
ContextCallback callback, Object state)
      at System.Threading.ThreadHelper.ThreadStart()
```

8. ArrayTypeMismatchException

```
using System;
    public class MyAppException:ApplicationException
    {
    // Constructors
      public MyAppException (String message) : base (message)
      {}
      public MyAppException (String message, Exception inner) :
    base(message,inner) {}
      }
    public class ExceptExample
    {
      public void ThrowInner ()
      {
    // Throwing Exception from the Class manually
      throw new MyAppException("ExceptExample inner exception");
      }
      public void CatchInner()
      {
        try
        {
    // Invoking throwInner() to throw manually declared exception
        this.ThrowInner();
        }
          catch (Exception e)
          {
    // catching all type of Exception
    // Throw manully created exception from the catch
          throw new MyAppException("Error caused by trying ThrowInner.",e);
          }
        }
      }
    public class Test
    {
      public static void Main()
      {
    // Creating objects for the class
      ExceptExample testInstance = new ExceptExample();
        try
        {
    //Invoking caatchInner()
        testInstance.CatchInner();
        }

          catch(Exception e)
          {
```

```
        //Displays the Exception messages
              Console.WriteLine ("In Main catch block. Caught: {0}", e.Message);
              Console.WriteLine ("Inner Exception is {0}",e.InnerException);
              }
          }
        }
```

OUTPUT:

In Main catch block. Caught: Error caused by trying ThrowInner. Inner Exception is MyAppException: ExceptExample inner exception at ExceptExample.ThrowInner() at ExceptExample.CatchInner()

9. NullReferenceException

```
using System;
public class Ints {
public int[] myInts;
}
public class NullRefExample {
public static void Main() {
    Ints ints = new Ints();
    try {
// Assignning null reference to the integer variable
      int i = ints.myInts[0];
    }
    catch( NullReferenceException e ) {
// Exception block to catch the null reference exception
      Console.WriteLine( "Caught error: {0}.", e);
    }
  }
}
```

OUTPUT:

Caught error: System.NullReferenceException: Object reference not set to an instance of an object.
at NullRefExample.Main().

10. IndexOutOfRangeException

```
class Program
{
   static void Main()
   {
      // Allocate an array of one-hundred integers.
      // ... Then assign to positions in the array.
      // ... Assigning past the last element will throw.
```

```
        int[] array = new int[100];
        array[0] = 1;
        array[10] = 2;
        array[200] = 3;
    }
}
```

OUTPUT:

Unhandled Exception: System.**IndexOutOfRangeException**:
Index was outside the bounds of the array.
  at Program.Main() in ...Program.cs:line 8

2. Analyze the JAVA Compiler and C# Compiler

The Compilers generates million lines of source code with warnings, however, not all programmers share my views and complain that many compiler warnings are not really defects. But it affects the efficiency of code. From the access of this model we experienced and displayed many unknown flaws, if only because hide serious problems.

When the C++ linker encountered global symbols with the same name, only the first symbol encountered was linked into the load, and a warning was generated for any subsequent symbol definitions. While investigating the compiler, the model identified many warnings message as well as defected lines. The defects reflected either global variables or global functions with names that are not unique. The various outputs rely on major and minor defects which are affected with serious errors in any application or software.

This model is intended to help with source code inspections by identifying common JAVA coding errors and C# coding errors are explained above. The model is supporting to detect the various defects in the form of index out of range exception, array type mismatch, and path too long and so on. To find the possible solution for this model, a systematic metric can point the potential problems in the overall software organization.

3. Conclusion

These developed techniques will identify logical defects in any software and can be implemented quickly at low cost. Through the analysis the defect identification, localization are fulfilling their needs but are not radically all, because the time consumption is essential to do the tasks.

4. Acknowledgement

I am grateful to UGC for providing this opportunity to enhance my research aspects as well the development in the context of depth of the reducing risks over programmer's and software organization level.