iTester: C#.NET User Manual

By Leonard Koh

iTester is an open source software performance testing framework originally developed by Dr Shiping Chen from CSIRO Australia ICT division.
This is the user manual for version 1.0 of iTester on the .NET platform for Windows Vista. If you are using an older version please update to the new version for bugs fixes and improvements.

Includes:

GoogleTester.cs
GoogleTestRun.cs
ATMclient.cs
ATMserver.cs
object.cs
WebFetch.cs

C#.NET: Requirements

iTester 1.0 requires the following software packages to be installed:

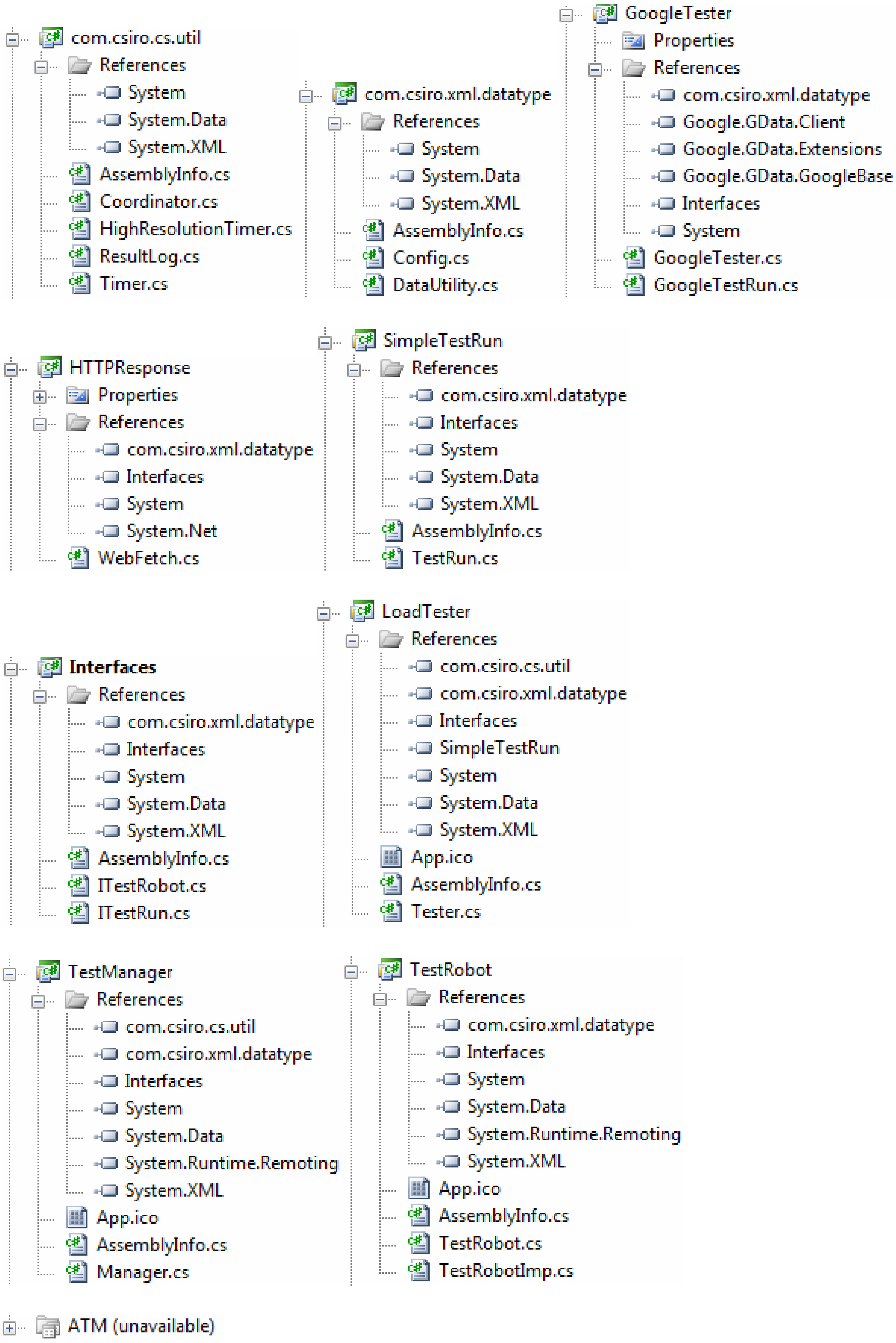
Microsoft Visual Studio 2008 *http://msdn.microsoft.com/en-au/vstudio/products/bb931328.aspx*

.NET Framework 2.0 Software Development Kit (SDK) or greater
http://www.microsoft.com/downloads/details.aspx?FamilyID=fe6f2099-b7b4-4f47-a244-c96d69c35dec&displaylang=en

While it is possible to manually compile the packages and it may work on a lower version of the .NET SDK, it is neither supported nor recommended.

C#.NET: Installation

A screenshot of the C# files, their respective directories and the required references are shown below:



- com.csiro.cs.util
  - References
    - System
    - System.Data
    - System.XML
  - AssemblyInfo.cs
  - Coordinator.cs
  - HighResolutionTimer.cs
  - ResultLog.cs
  - Timer.cs

- com.csiro.xml.datatype
  - References
    - System
    - System.Data
    - System.XML
  - AssemblyInfo.cs
  - Config.cs
  - DataUtility.cs

- GoogleTester
  - Properties
  - References
    - com.csiro.xml.datatype
    - Google.GData.Client
    - Google.GData.Extensions
    - Google.GData.GoogleBase
    - Interfaces
    - System
  - GoogleTester.cs
  - GoogleTestRun.cs

- HTTPResponse
  - Properties
  - References
    - com.csiro.xml.datatype
    - Interfaces
    - System
    - System.Net
  - WebFetch.cs

- SimpleTestRun
  - References
    - com.csiro.xml.datatype
    - Interfaces
    - System
    - System.Data
    - System.XML
  - AssemblyInfo.cs
  - TestRun.cs

- **Interfaces**
  - References
    - com.csiro.xml.datatype
    - Interfaces
    - System
    - System.Data
    - System.XML
  - AssemblyInfo.cs
  - ITestRobot.cs
  - ITestRun.cs

- LoadTester
  - References
    - com.csiro.cs.util
    - com.csiro.xml.datatype
    - Interfaces
    - SimpleTestRun
    - System
    - System.Data
    - System.XML
  - App.ico
  - AssemblyInfo.cs
  - Tester.cs

- TestManager
  - References
    - com.csiro.cs.util
    - com.csiro.xml.datatype
    - Interfaces
    - System
    - System.Data
    - System.Runtime.Remoting
    - System.XML
  - App.ico
  - AssemblyInfo.cs
  - Manager.cs

- TestRobot
  - References
    - com.csiro.xml.datatype
    - Interfaces
    - System
    - System.Data
    - System.Runtime.Remoting
    - System.XML
  - App.ico
  - AssemblyInfo.cs
  - TestRobot.cs
  - TestRobotImp.cs

- ATM (unavailable)

Note: In each of these directories listed above, Microsoft Visual Studio will create **bin** and **obj** directories, where the library(.dll)/executable(.exe) is built into **bin/Debug/**.

In addition to these C# files:

/LoadTester2/ATM will include: *ATMclient.cs*, *ATMserver.cs, object.cs, build.bat, clean.bat, README.txt, com.csiro.xml.datatype.dll, Interfaces.dll*

/LoadTester2/com.csiro.cs.util: *comcsiroutil.xml*

/LoadTester2/com.csiro.xml.datatype: *Address.xsd*

/LoadTester2/GoogleTester: *README.txt*

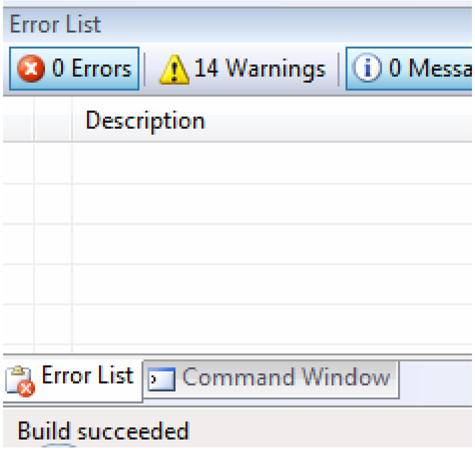
/LoadTester2/HTTPResponse: *README.txt*

Please check you have all the required files, directories and have included the required package references before attempting to begin the installation.

To build all the files, into their respective directories (except for ATM):

Select “Build” → “Build solution” (F6) in Microsoft Visual Studio 2008

The screen should appear as such:



Build succeeded

To build ATM:

Open a .NET SDK 2.0 command prompt window and run **/LoadTester2/ATM/build.bat**

Read **/LoadTester2/ATM/README.txt** for further building, cleaning and running instructions.

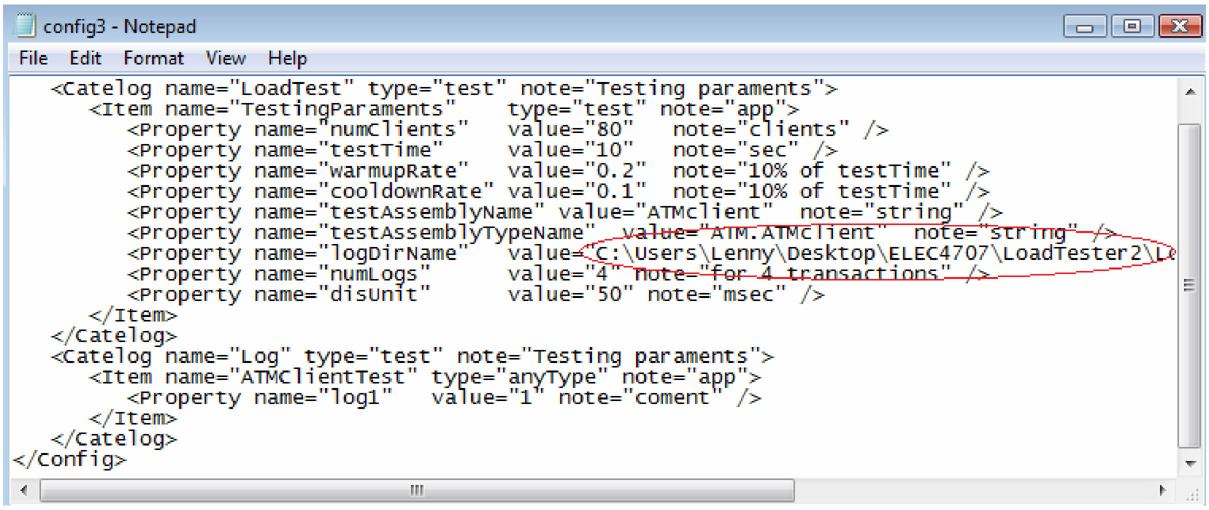
C#.NET: Running iTester

After successful compilation, ensure that the test assemblies to be tested and their respective references are moved to **/LoadTester2/LoadTester/ bin/Debug/** before testing.

0. Log File Setting

After this, create the folder **/LoadTester2/LoadTester/ log.**

Next edit the path of the “logDirName” property in the config.xml file to the **/LoadTester2/LoadTester/ log** folder created.



```
    <Catelog name="LoadTest" type="test" note="Testing paraments">
       <Item name="TestingParaments"    type="test" note="app">
          <Property name="numClients"   value="80"   note="clients" />
          <Property name="testTime"     value="10"   note="sec" />
          <Property name="warmupRate"   value="0.2"  note="10% of testTime" />
          <Property name="cooldownRate" value="0.1"  note="10% of testTime" />
          <Property name="testAssemblyName" value="ATMClient"  note="string" />
          <Property name="testAssemblyTypeName"  value="ATM.ATMClient"  note="string" />
          <Property name="logDirName"   value="C:\Users\Lenny\Desktop\ELEC4707\LoadTester2\L
          <Property name="numLogs"      value="4"  note="for 4 transactions"  />
          <Property name="disUnit"      value="50" note="msec" />
       </Item>
    </Catelog>
    <Catelog name="Log" type="test" note="Testing paraments">
       <Item name="ATMClientTest" type="anyType" note="app">
          <Property name="log1"   value="1" note="coment" />
       </Item>
    </Catelog>
</Config>
```

1. To run GoogleTester implementation

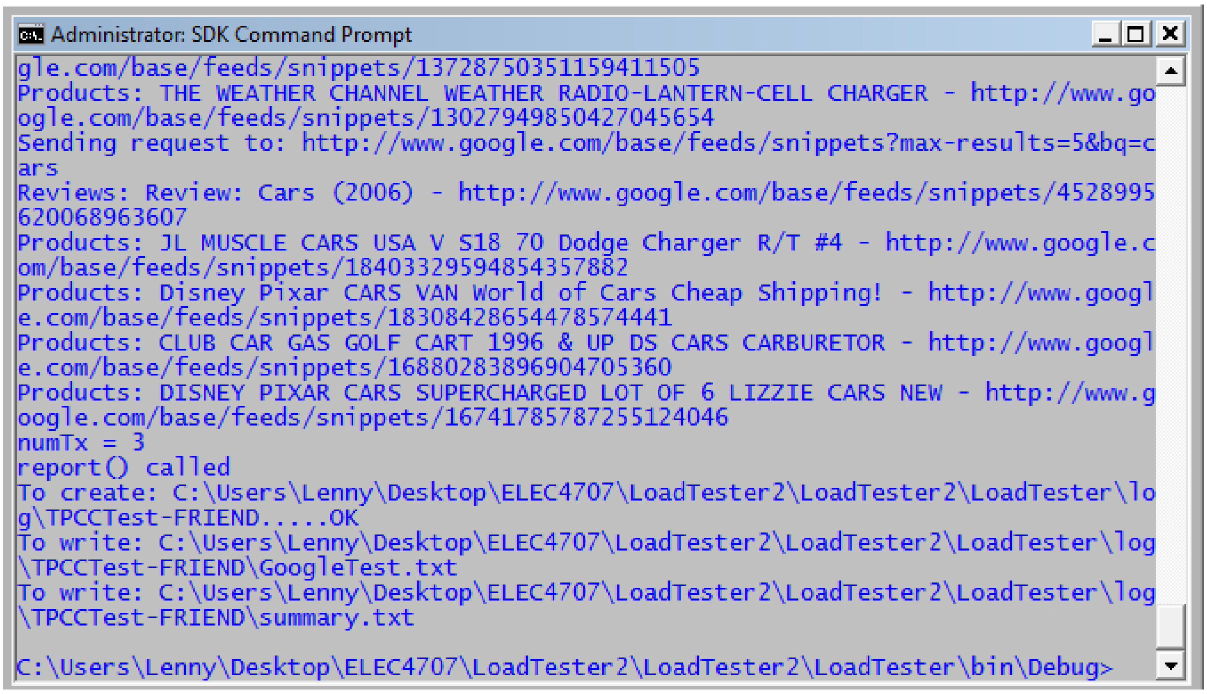
Open the .NET SDK 2.0 command prompt window. Change directory to **/LoadTester2/LoadTester/ bin/Debug/**

Open **/LoadTester2/LoadTester/ bin/Debug/config.xml** and change the “testAssemblyName” property value to “GoogleTester” and “testAssemblyTypeName” to “GoogleTester.GoogleTestRun”.

Alternatively:

Type **LoadTester –f config2.xml** (as it has already been prepared to run this assembly)

The output should appear as follows:



```
gle.com/base/feeds/snippets/13728750351159411505
Products: THE WEATHER CHANNEL WEATHER RADIO-LANTERN-CELL CHARGER - http://www.go
ogle.com/base/feeds/snippets/13027949850427045654
Sending request to: http://www.google.com/base/feeds/snippets?max-results=5&bq=c
ars
Reviews: Review: Cars (2006) - http://www.google.com/base/feeds/snippets/4528995
620068963607
Products: JL MUSCLE CARS USA V S18 70 Dodge Charger R/T #4 - http://www.google.c
om/base/feeds/snippets/18403329594854357882
Products: Disney Pixar CARS VAN World of Cars Cheap Shipping! - http://www.googl
e.com/base/feeds/snippets/18308428654478574441
Products: CLUB CAR GAS GOLF CART 1996 & UP DS CARS CARBURETOR - http://www.googl
e.com/base/feeds/snippets/16880283896904705360
Products: DISNEY PIXAR CARS SUPERCHARGED LOT OF 6 LIZZIE CARS NEW - http://www.g
oogle.com/base/feeds/snippets/16741785787255124046
numTx = 3
report() called
To create: C:\Users\Lenny\Desktop\ELEC4707\LoadTester2\LoadTester2\LoadTester\lo
g\TPCCTest-FRIEND.....OK
To write: C:\Users\Lenny\Desktop\ELEC4707\LoadTester2\LoadTester2\LoadTester\log
\TPCCTest-FRIEND\GoogleTest.txt
To write: C:\Users\Lenny\Desktop\ELEC4707\LoadTester2\LoadTester2\LoadTester\log
\TPCCTest-FRIEND\summary.txt

C:\Users\Lenny\Desktop\ELEC4707\LoadTester2\LoadTester2\LoadTester\bin\Debug>
```

2. To run ATM client server implementation

Locate **/LoadTester2/ATM/ATMserver.exe** and run.

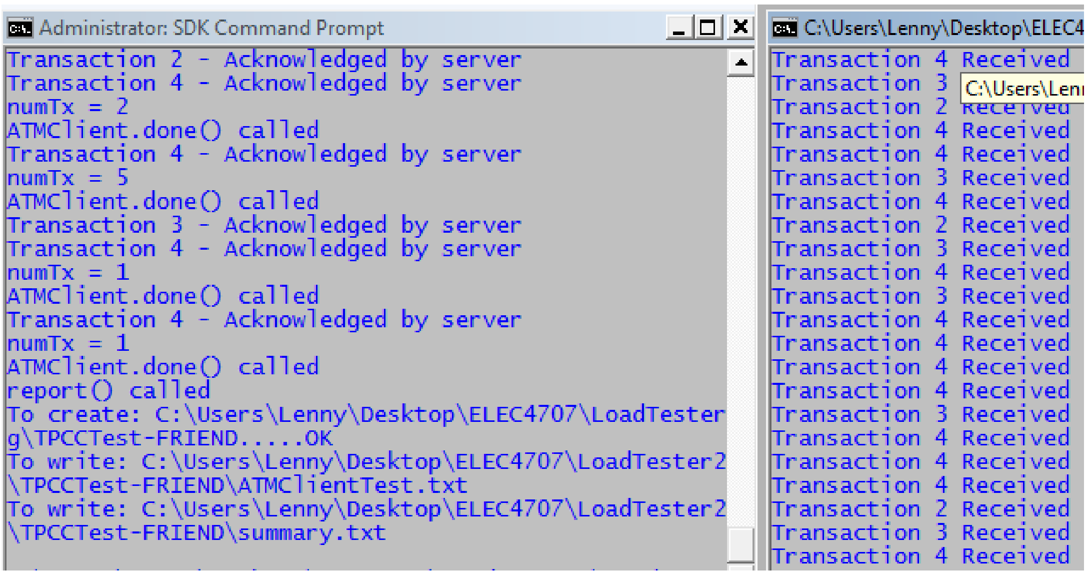
Open a new .NET SDK 2.0 command prompt window. Change directory to **/LoadTester2/LoadTester/ bin/Debug/**

Open **/LoadTester2/LoadTester/ bin/Debug/config.xml** and change the "testAssemblyName" property value to "ATMclient" and "testAssemblyTypeName" to "ATM.ATMclient".

Alternatively:

Type **LoadTester –f config3.xml** (as it has already been prepared to run this assembly) or **LoadTester –f config4.xml**. (config3 runs 80 clients for 10 seconds, config4 runs 1000 clients for 5 seconds)

The output should appear as follows:


Administrator: SDK Command Prompt
C:\Users\Lenny\Desktop\ELEC4
C:\Users\Len

```
Transaction 2 - Acknowledged by server
Transaction 4 - Acknowledged by server
numTx = 2
ATMClient.done() called
Transaction 4 - Acknowledged by server
numTx = 5
ATMClient.done() called
Transaction 3 - Acknowledged by server
Transaction 4 - Acknowledged by server
numTx = 1
ATMClient.done() called
Transaction 4 - Acknowledged by server
numTx = 1
ATMClient.done() called
report() called
To create: C:\Users\Lenny\Desktop\ELEC4707\LoadTester
g\TPCCTest-FRIEND.....OK
To write: C:\Users\Lenny\Desktop\ELEC4707\LoadTester2
\TPCCTest-FRIEND\ATMClientTest.txt
To write: C:\Users\Lenny\Desktop\ELEC4707\LoadTester2
\TPCCTest-FRIEND\summary.txt
```

```
Transaction 4 Received
Transaction 3
Transaction 2 Received
Transaction 4 Received
Transaction 4 Received
Transaction 3 Received
Transaction 4 Received
Transaction 2 Received
Transaction 3 Received
Transaction 4 Received
Transaction 3 Received
Transaction 4 Received
Transaction 4 Received
Transaction 4 Received
Transaction 4 Received
Transaction 3 Received
Transaction 4 Received
Transaction 4 Received
Transaction 4 Received
Transaction 2 Received
Transaction 3 Received
Transaction 4 Received
```

3. To run HTTP Response implementation

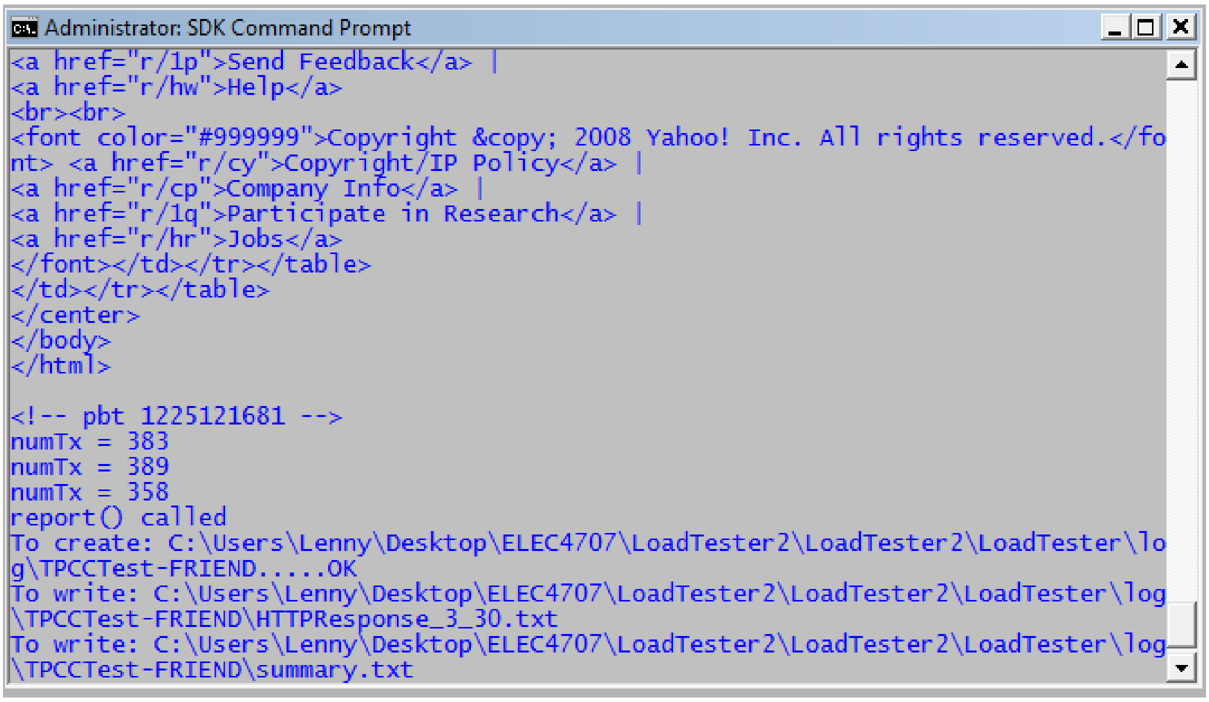
Open the .NET SDK 2.0 command prompt window. Change directory to **/LoadTester2/LoadTester/ bin/Debug/**

Open **/LoadTester2/LoadTester/ bin/Debug/config.xml** and change the "testAssemblyName" property value to "WebFetch" and "testAssemblyTypeName" to "HTTPResponse.WebFetch".

Alternatively:

Type **LoadTester –f config5.xml** (as it has already been prepared to run this assembly) or **LoadTester –f config6.xml**. (config5 runs 20 clients for 10 seconds, config6 runs 3 clients for 30 seconds)

The output should appear as follows:

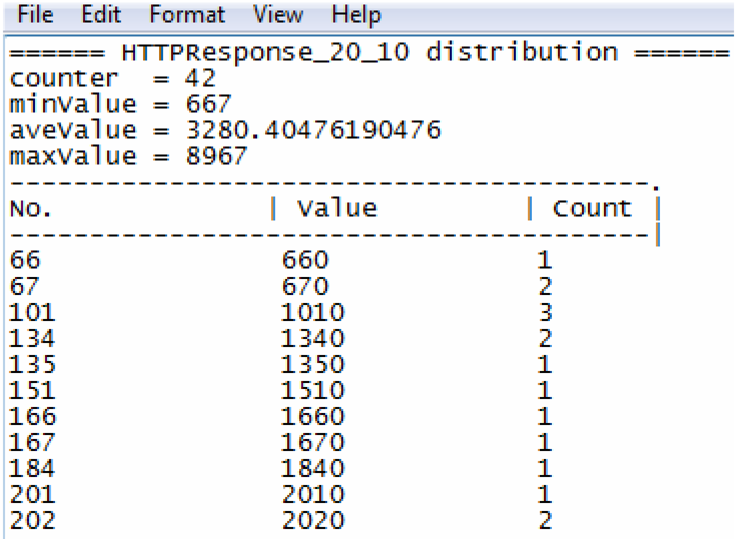


```
<a href="r/1p">Send Feedback</a> |
<a href="r/hw">Help</a>
<br><br>
<font color="#999999">Copyright &copy; 2008 Yahoo! Inc. All rights reserved.</fo
nt> <a href="r/cy">Copyright/IP Policy</a> |
<a href="r/cp">Company Info</a> |
<a href="r/1q">Participate in Research</a> |
<a href="r/hr">Jobs</a>
</font></td></tr></table>
</td></tr></table>
</center>
</body>
</html>

<!-- pbt 1225121681 -->
numTx = 383
numTx = 389
numTx = 358
report() called
To create: C:\Users\Lenny\Desktop\ELEC4707\LoadTester2\LoadTester2\LoadTester\lo
g\TPCCTest-FRIEND.....OK
To write: C:\Users\Lenny\Desktop\ELEC4707\LoadTester2\LoadTester2\LoadTester\log
\TPCCTest-FRIEND\HTTPResponse_3_30.txt
To write: C:\Users\Lenny\Desktop\ELEC4707\LoadTester2\LoadTester2\LoadTester\log
\TPCCTest-FRIEND\summary.txt
```

4. **Test Report**

At the end of each test implementation a report will be called with an output similar to:



```
====== HTTPResponse_20_10 distribution ======
counter  = 42
minValue = 667
aveValue = 3280.40476190476
maxValue = 8967
-----------------------------------------.
No.             | Value          | Count |
-----------------------------------------|
66               660             1
67               670             2
101              1010            3
134              1340            2
135              1350            1
151              1510            1
166              1660            1
167              1670            1
184              1840            1
201              2010            1
202              2020            2
```

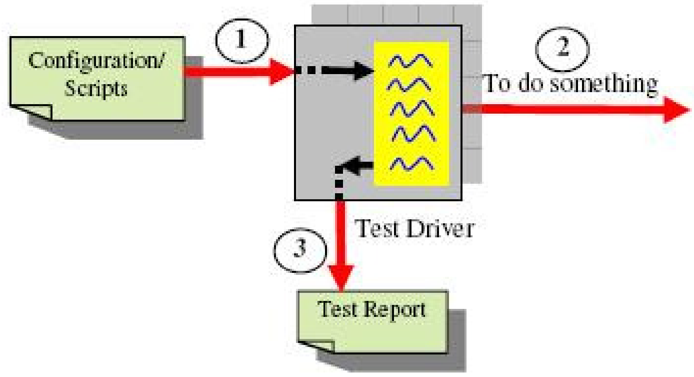
minValue = minimal value or the fastest response time
aveValue = average value of the entire set
maxValue = maximum value or the slowest response time

iTester: C#.NET Developer Manual

This is the developer manual for iTester version 1.0, please update to the newest version in compliance with this manual. This manual will cover standard APIs and troubleshooting for the framework. It is also highly recommended to read the original paper on the open source software performance testing framework released by CSIRO.

- S.Chen, D.Moreland, S.Nepal, J.Zic, "*Yet Another Performance testing Framework*", Networking Technologies Laboratory, CSIRO ICT Centre Australia, 2008

C#.NET: Implementation



To create your own implementations, please first examine the example implementations. From the above diagram there are two key components to an implementation.

- The test class which implements ITestRun.
- The configuration script for the test class (config.xml).

The ITestRun interface is shown below in the API section. Using the SimpleTestRun implementation as a template it is recommended as it is a very basic example. Create a new test assembly (.dll) by using Microsoft Visual Studio to compile.

After a test assembly has been created, a configuration script must be created and stored in the **/LoadTester2/LoadTester/ bin/Debug/** directory. Again, using the SimpleTestRun configuration as a template is highly recommended (config.xml file).

C#.NET: API

ITestRun

ITestRun is iTester's test script implementation interface. Its standard functions are:

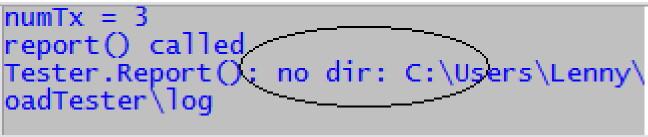
- **Init (Properties prop)** – Init () function acts to initialise any required properties or actions needed to perform the test. Example for this may be setting up a server connection to a database. Each thread will call init () initially however, each test will only call it once.

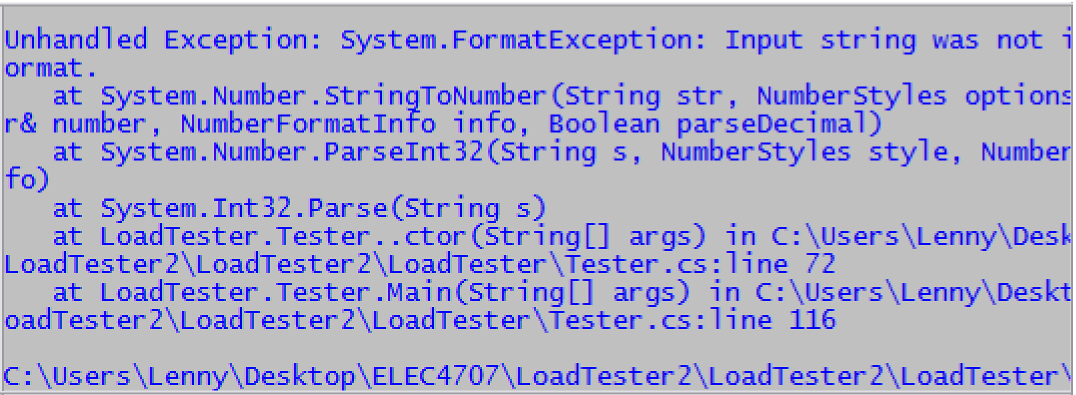
- **Run ()** – is the heart of the test script, the run () function will have the specified code for which the test threads are to perform. Each thread will repeatedly call the run () function to perform what is desired by the test engineer. This allows for a wide degree of flexibility as the test engineer may program whatever actions he chooses.

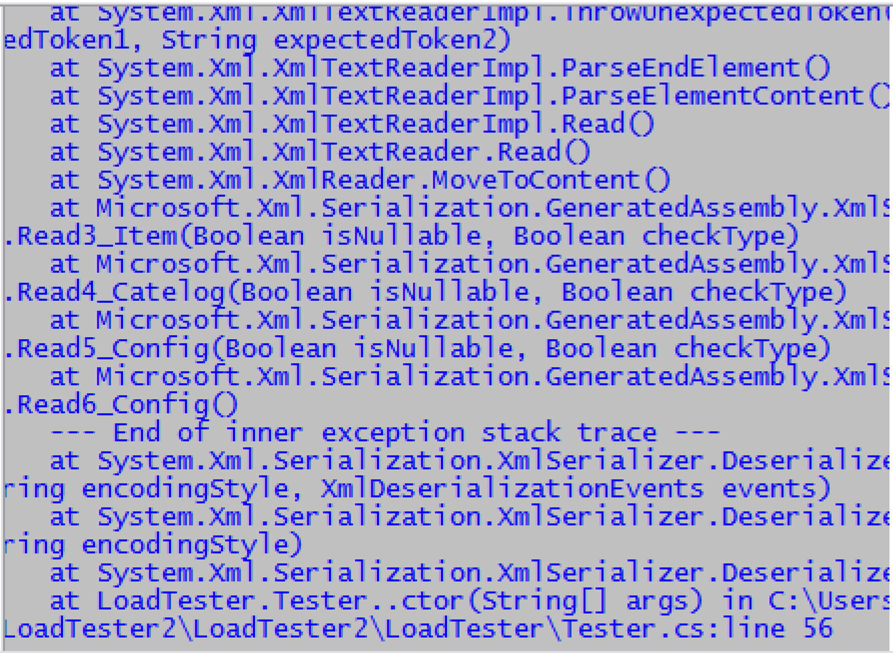
- **Pre and post Run** () – are the preparation and clean up methods, it provides the option and possibility for even greater detail tests. For example, random sleeps to simulate the typing delay of users.

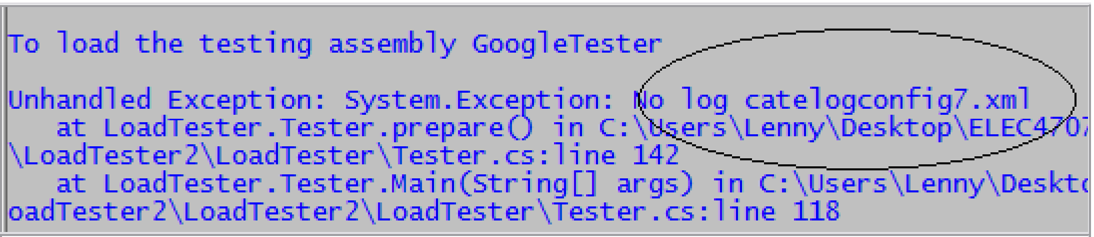
- **Done** () – Called at the completion of all the tests, it allows for final closure and freeing up of resources and any other house keeping that may be required.

C#.NET: Troubleshooting

Profile	 numTx = 3 report() called Tester.Report(): no dir: C:\Users\Lenny\ oadTester\log
Reason	Cannot locate log directory specified in the config file.
Solution	Check that the directory exists, if not create it. Otherwise locate the config file and edit the log path.

Profile	 Unhandled Exception: System.FormatException: Input string was not i ormat. at System.Number.StringToNumber(String str, NumberStyles options r& number, NumberFormatInfo info, Boolean parseDecimal) at System.Number.ParseInt32(String s, NumberStyles style, Number fo) at System.Int32.Parse(String s) at LoadTester.Tester..ctor(String[] args) in C:\Users\Lenny\Desk LoadTester2\LoadTester2\LoadTester\Tester.cs:line 72 at LoadTester.Tester.Main(String[] args) in C:\Users\Lenny\Deskt oadTester2\LoadTester2\LoadTester\Tester.cs:line 116 C:\Users\Lenny\Desktop\ELEC4707\LoadTester2\LoadTester2\LoadTester\
Reason	Property value missing in the config file.
Solution	Locate the config file and add in a value for the missing property.

Profile	 at System.Xml.XmlTextReaderImpl.ThrowUnexpectedToken(edToken1, String expectedToken2) at System.Xml.XmlTextReaderImpl.ParseEndElement() at System.Xml.XmlTextReaderImpl.ParseElementContent(at System.Xml.XmlTextReaderImpl.Read() at System.Xml.XmlTextReader.Read() at System.Xml.XmlReader.MoveToContent() at Microsoft.Xml.Serialization.GeneratedAssembly.XmlS .Read3_Item(Boolean isNullable, Boolean checkType) at Microsoft.Xml.Serialization.GeneratedAssembly.XmlS .Read4_Catelog(Boolean isNullable, Boolean checkType) at Microsoft.Xml.Serialization.GeneratedAssembly.XmlS .Read5_Config(Boolean isNullable, Boolean checkType) at Microsoft.Xml.Serialization.GeneratedAssembly.XmlS .Read6_Config() --- End of inner exception stack trace --- at System.Xml.Serialization.XmlSerializer.Deserialize ring encodingStyle, XmlDeserializationEvents events) at System.Xml.Serialization.XmlSerializer.Deserialize ring encodingStyle) at System.Xml.Serialization.XmlSerializer.Deserialize at LoadTester.Tester..ctor(String[] args) in C:\Users LoadTester2\LoadTester2\LoadTester\Tester.cs:line 56
Reason	XML Syntax error.
Solution	Check and amend the syntax of the XML file.

Profile	 To load the testing assembly GoogleTester Unhandled Exception: System.Exception: No log catelogconfig7.xml at LoadTester.Tester.prepare() in C:\Users\Lenny\Desktop\ELEC470 \LoadTester2\LoadTester\Tester.cs:line 142 at LoadTester.Tester.Main(String[] args) in C:\Users\Lenny\Deskt oadTester2\LoadTester2\LoadTester\Tester.cs:line 118
Reason	Missing the second catelog section.
Solution	Locate another config file and add the missing catelog information. It should look like: *<Catelog name="Log" type="test" note="Testing paraments">* *<Item name="GoogleTest" type="GoogleTest" note="app">* *<Property name="log1" value="1" note="coment" />* *</Item>* *</Catelog>*