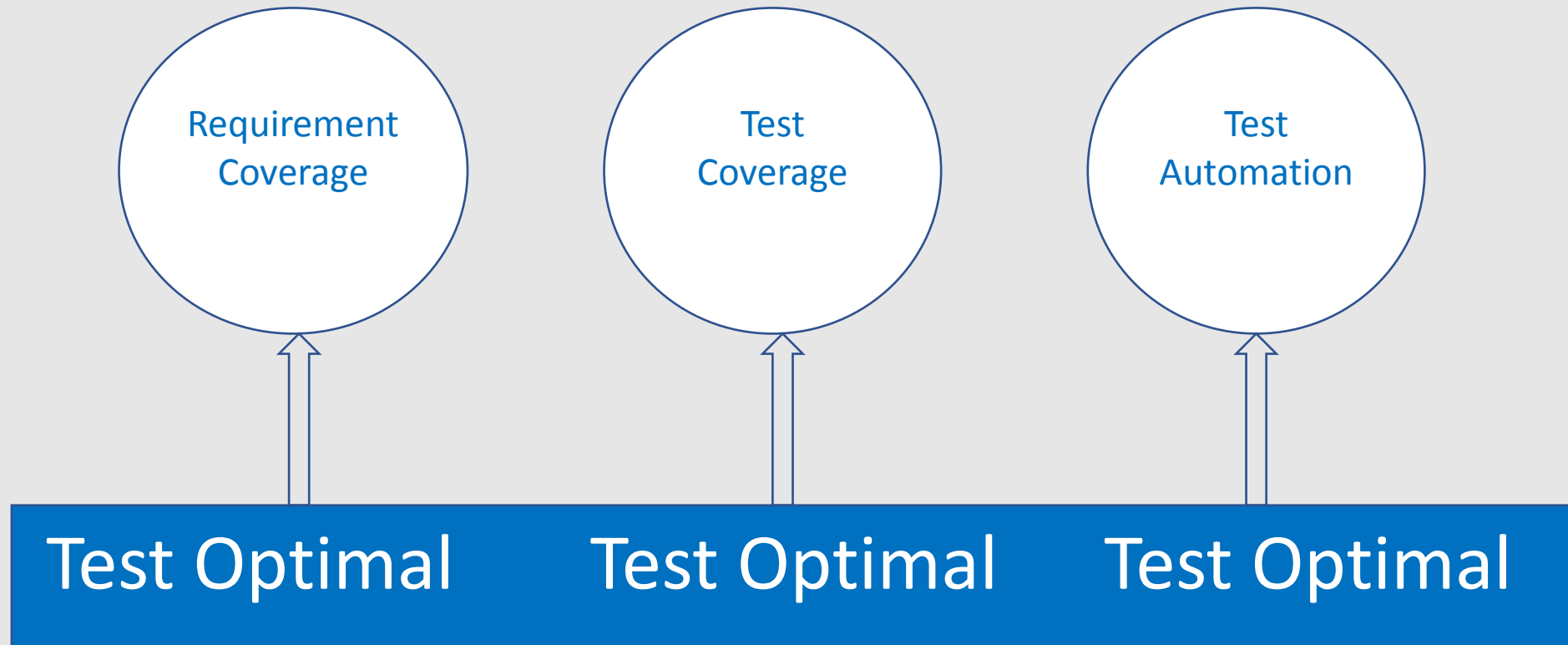


Test embedded system  
developed in C or other  
program languages

# TestOptimal Introduction

# Why choose TestOptimal for embedded system



# Sample embedded system developed in C

To make it simple, suppose the embedded system compose of four functions:

1

Add(double x,  
double y);

2

Substract(double x,  
double y);

3

Multiply(double x,  
double y);

4

Divide(double x,  
double y);

dllmain.cpp

demo

(Global Scope)

```
1 // dllmain.cpp : Defines the entry point for the DLL application.
2 #include "stdafx.h"
3 #include "calculator.h"
4
5 double sum = 0;
6 void Init()
7 {
8     sum = 0;
9 }
10
11 double Add(double x, double y)
12 {
13     return x + y;
14 }
15
16 double Subtract(double x, double y)
17 {
18     return x - y;
19 }
20
21 double Multiply(double x, double y)
22 {
23     return x * y;
24 }
25
26 double Divide(double x, double y)
27 {
28     return x / y;
29 }
30
31
32 BOOL APIENTRY DllMain(HMODULE hModule,
33     DWORD ul_reason_for_call,
34     LPVOID lpReserved
```

91 %

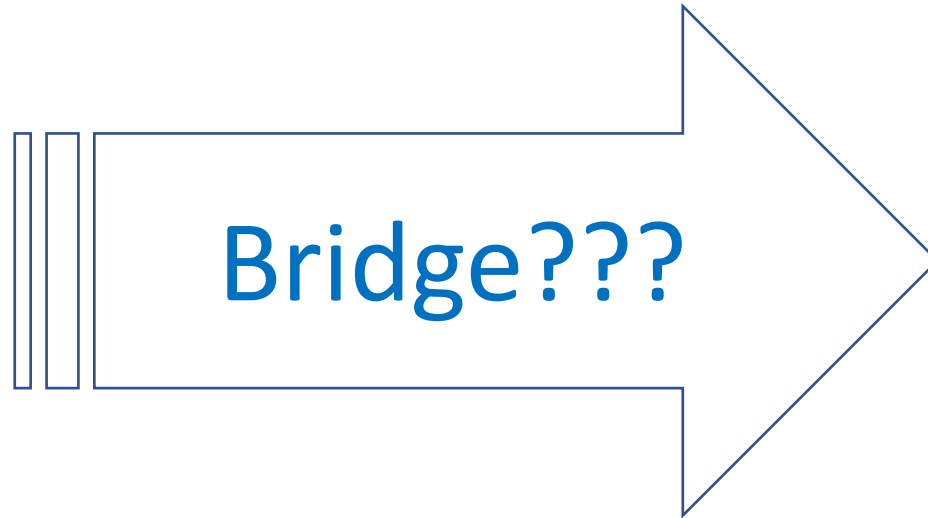
# TestOptimal MScript

*MScript* is an XML based scripting to either directly drive your application under test (AUT) or generate testing script to be executed or read later for offline testing in any programming language syntax.

- ❖ Call custom plugin
- ❖ Call remote agent



**Embedded Systems**



# Two Options of the 'Bridge'

1

## Custom Plugin

*Custom Plugins* is used to extend *TestOptimal* functionality by implementing a set of functions to be used in MScript. Your plugin can then be enabled in [Model Property](#) to call its MScript methods in the model

2

## Remote Agent

*Remote Agent* is used to integrate *TestOptimal* with your embedded system. It operates on the plain http, as long as the external tool can send and receive http requests, you can integrate it with *TestOptimal*. This covers pretty much any tools on the market.

# Custom Plugin – 1<sup>st</sup> Bridge

workspace - <Java EE> - TO\_IDE/src/demo/customPlugin/EmbeddedSysPlugin.java - Eclipse

Edit Source Refactor Navigate Search Project Run Window Help

```
EmbeddedSysPlugin.java
17 public final class EmbeddedSysPlugin extends PluginAncestor {
18     |
19     /**
20      * ModelMgr provides access to model attributes.
21      */
22     private ModelMgr modelMgr;
23
24     public static native double add(double x, double y);
25     public static native double subtract(double x, double y);
26     public static native double multiply(double x, double y);
27     public static native double divide(double x, double y);
28
29
30     static {
31         String dllName = "demo_64";
32         String libFilePath = Config.getRootPath() + "/lib/" + dllName + ".dll";
33         try{
34             TestOptimalServer.logInfo("Loading library from " + libFilePath);
35             System.load(libFilePath);
36             TestOptimalServer.logInfo("Loaded library " + dllName);
37         } catch(Throwable e) {
38             TestOptimalServer.logError("Error loading library " + libFilePath, e);
39         }
40     }
41
42
43     @MScriptMethod
44     public double add (String p1, String p2) throws MBTException {
45         try {
46             double p1Value = Double.parseDouble(p1);
47             double p2Value = Double.parseDouble(p2);
48             double result = add (p1Value, p2Value); // Invoke C function by JNI
49             return result;
50         } catch (Throwable e) {
51             this.scriptExecutor.error("Embedded.add Errored: " + e.toString());
52             throw new MBTException (e.getMessage() + ": " + e.toString());
53         }
54     }
}
```



# TestOptimal invokes functions defined in Custom Plugin

The screenshot displays the ProMBT web interface. On the left, a state machine model is shown with a 'Start' state and an 'End' state. Transitions between these states are labeled with mathematical operations: 'add R V D', 'subtract R V D', 'multiply R V D', and 'divide R V D'. On the right, the 'Monitor' tab is active, showing execution status and progress. Below this, the 'MScript' tab displays the generated code for the model. A blue box highlights a specific line in the MScript code: `<action lid="17" code="$setVar('result', '$Embedded.add('[num]', '[num2]')')"/>`. The interface also shows a menu bar with 'File', 'Model', 'Run', 'Report', 'Session', 'Log', and 'Help'. The URL in the browser is 'localhost:8888/MbtSvr/webmbtMain.html'. The ProMBT version is 5.2.0. The execution status shows 'Status: Success', 'MBT Mode: PriorityPath', 'Debug: false', 'Iterations: 1', 'DryRun: false', and 'Virtual Users: 1'. The execution date and time are '4/9/2017, 2:53:07 PM'. The MScript code includes various tags for state transitions, actions, and scripts, such as `<mbt lid="1">`, `<script lid="2" type="MBT_start">`, `<action lid="3" code="$setVar('PluginID', '$getPluginID()')"/>`, `<log lid="4" level="message" message="Running testing using [PluginID] plugin"/>`, `</script>`, `</mbt>`, `<state lid="5" id="End">`, `<script lid="6" type="onentry"/>`, `<script lid="7" type="onexit"/>`, `</state>`, `<state lid="8" id="Start">`, `<script lid="9" type="onentry">`, `<action lid="10" code="$setVar('result', 'Unknown')"/>`, `</script>`, `<script lid="11" type="onexit"/>`, `<transition lid="12" event="add">`, `<script lid="13" type="action">`, `<action lid="14" code="$setVar('num', '$getTransData('Num')')"/>`, `<action lid="15" code="$setVar('num2', '$getTransData('Num2')')"/>`, `<action lid="16" code="$setVar('expNum', '$getTransData('ExpNum')')"/>`, `<action lid="17" code="$setVar('result', '$Embedded.add('[num]', '[num2]')')"/>`, `</script>`, `<script lid="18" type="prep"/>`, `<script lid="19" type="verify"/>`, `<assert lid="20" code="$assert(result, 'eq', 'eq')"/>`, `</assert>`, `</script>`, `</transition>`, `</state>`, `</state>`, `</mbt>`.

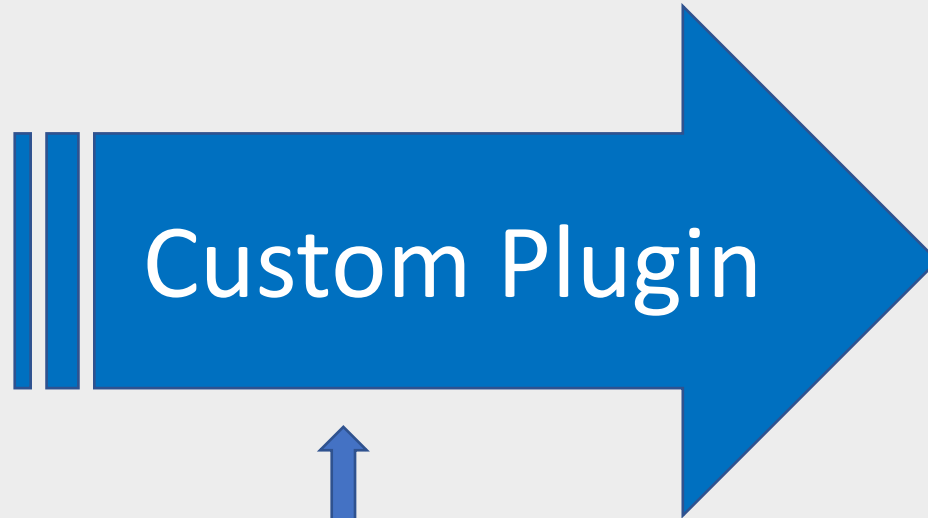
Copyright by TestOptimal © 2017 All Rights Reserved  
Contact Amy Xia: xia.amy@testoptimal.com



## Embedded Systems



Developed in C or other program languages



Developed in Java to expose necessary functions for mscripts



Mscripts invokes functions defined in custom plugin and drives the model running

# Remote Agent – 2<sup>nd</sup> Bridge

workspace - <Java EE> - TOAgent/src/demo/DemoEmbeddedSysAgent.java - Eclipse

```
Edit Source Refactor Navigate Search Project Run Window Help
```

```
Quick Access <Java EE>
```

```
DemoEmbeddedSysAgent.java
```

```
63 @Override
64 public String execute(RemoteCmd remoteCmd_p) throws Exception {
65     String cmdSyntax = remoteCmd_p.getCmdActionSyntax();
66     this.info("Remote cmd: " + cmdSyntax);
67     this.cmdJSON = new JSONObject(cmdSyntax);
68
69     String cmdAction = this.cmdJSON.getString("ACTION");
70     this.info("Action: " + cmdAction);
71
72     if (cmdAction.equalsIgnoreCase("MBT_start")) {
73         this.info("Starting VendingMachine");
74         return "OK";
75     }
76     else if (cmdAction.equalsIgnoreCase("launchAUT")) {
77         this.info("Running the test");
78         return "OK";
79     }
80     else if (cmdAction.equalsIgnoreCase("resetAUT")) {
81         this.info("Resetting the test");
82         return "OK";
83     }
84     else if (cmdAction.equalsIgnoreCase("add")) {
85         String p1 = this.cmdJSON.getString("P1");
86         String p2 = this.cmdJSON.getString("P2");
87         this.info("add " + p1 + ", " + p2);
88         try {
89             double p1Value = Double.parseDouble(p1);
90             double p2Value = Double.parseDouble(p2);
91             double d = add(p1Value, p2Value);
92             return new Double(d).toString();
93             //return "add result: " + d;
94         }
95         catch (Throwable e) {
96             this.error("Failed to call add function: " + e.toString());
97             throw new Exception (e.toString());
98         }
99     }
100 }
```

Copyright by TestOptimal ©2017 All Rights Reserved  
Contact Amy Xia: xia.amy@testoptimal.com

# TestOptimal sends http requests to Remote Agent

The screenshot displays the ProMBT web interface. On the left, a state machine model is shown with a 'Start' state and an 'End' state. Transitions between these states are labeled with mathematical operations: 'add RVD', 'subtract RVD', 'multiply RVD', and 'divide RVD'. On the right, the 'MScript' tab is active, showing XML code for the test execution. The code includes actions for setting variables and sending remote commands. A specific line of code is highlighted with a blue box:

```
<action lid="18" code="$setVar('result', '$RemoteCommand.remoteCmd('add',[num],[num2]')')"/>
```

Below the code, the execution status is shown as 'ENDED'. The interface also includes a menu bar (File, Model, Run, Report, Session, Log, Help) and a toolbar with various icons for navigation and execution.



## Embedded Systems

Developed in C or other program languages



Developed in Java or other program language to handle the http requests between Embedded System and TestOptimal



Mscripts sends to and receives http requests from Remote Agent and drives the model running



# TestOptimal Dashboard

Dashboard - TestOptimal - Google Chrome

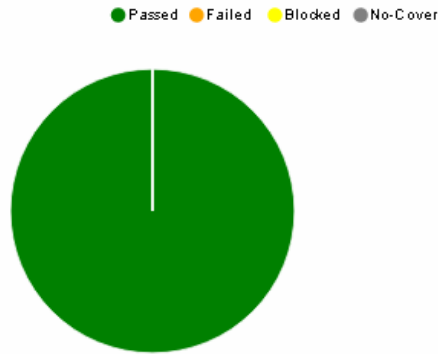
localhost:8888/MbtSvr/Dashboard\_Main.html

ProMBT 5.2.6 TestOptimal - Dashboard & Execution Summary

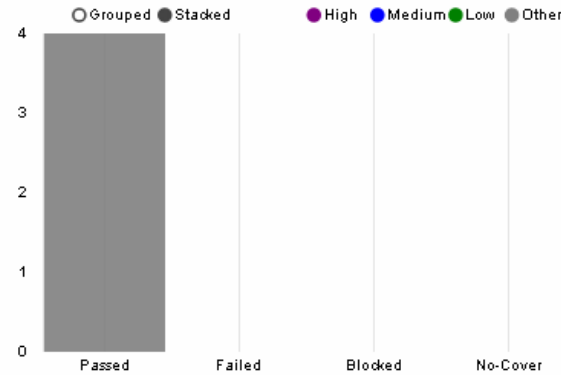
Dashboard Trend Reports Projects KPI Help KPI: TestCases 12

Dashboard Stats contains automated test executions only. Manual test case executions are not considered in these stats.

## Requirement Summary



Requirements Status

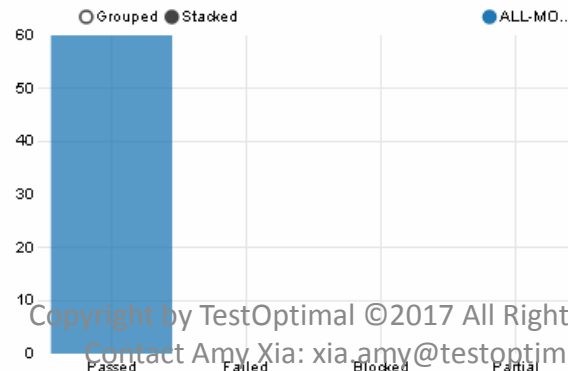
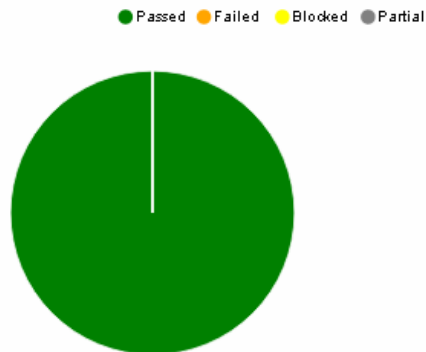


Requirement Execution Status

### Requirement Execution Stats

Status	High	Medium	Low	Other	Total
Passed	0	0	0	4	4
Failed					
Blocked					
No-Cover					
<b>Total</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>4</b>	<b>4</b>

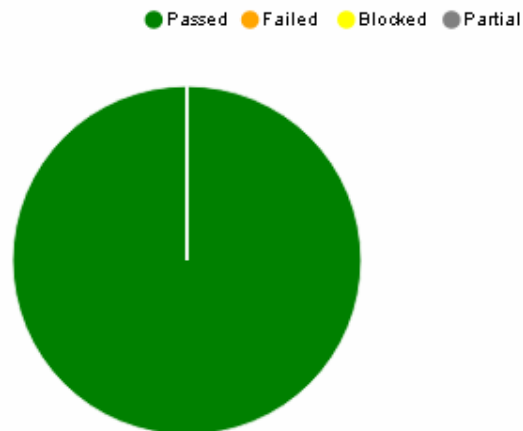
## Test Case Summary



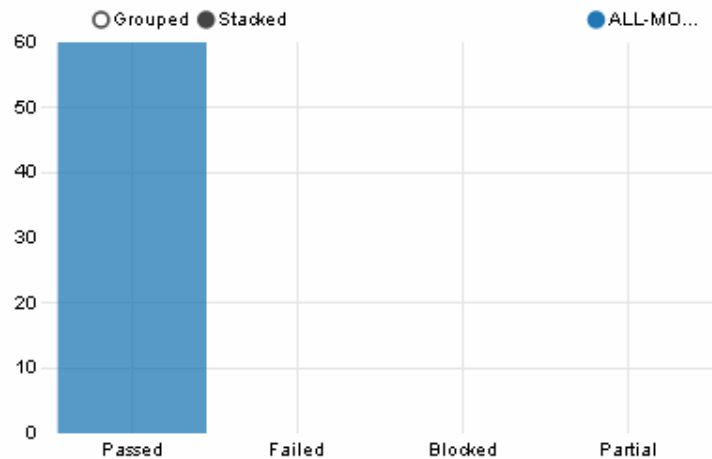
### Test Case Execution Stats

Total Test Cases	60
Failed	0
Blocked	0
Partial	0
Total Test Steps	120

Dashboard Stats contains automated test executions only. Manual test case executions are not considered in these stats.



Test Case Execution Status



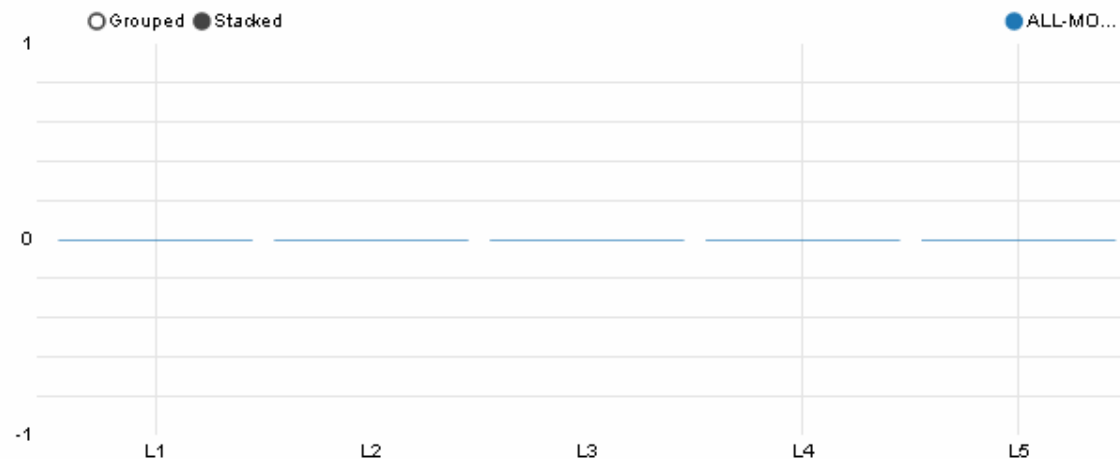
Test Case By Project

Total Test Cases	60
Failed	0
Blocked	0
Partial	0
Total Test Steps	120

Defects Summary

Legend: L1 (Grey), L2 (Blue), L3 (Green), L4 (Purple), L5 (Brown)

Defects By Severity



Defects By Project

# TestOptimal provides requirement coverage status

The screenshot displays the TestOptimal web interface. On the left, a state transition diagram shows a sequence of operations: Start, add, subtract, multiply, divide, and End. On the right, a table titled 'Manual Test Execution - Demo\_EmbeddedSys' shows the requirement coverage for 12 test cases (TC0001 to TC0012) across four requirement tags (1, 2, 3, 4). The table uses blue shading to indicate coverage: TC0001-03 cover tag 1; TC0004-06 cover tag 2; TC0007-09 cover tag 3; and TC0010-12 cover tag 4. A red speech bubble highlights the text 'Requirement Coverage' in the table. Below the table, the execution status is shown as 'Not Covered', 'Covered', 'Partially Passed', 'Passed', 'Failed', and 'Blocked'. Requirement tags are listed as 1: ADD, 2: DIVIDE, 3: MULTIPLY, 4: SUBTRACT.

Test Case	Weight	Steps	Requirement Tags <a href="#">show tag name</a>			
			1	2	3	4
TC0001	5	1	Covered			
TC0002	5	1	Covered			
TC0003	5	1	Covered			
TC0004	5	1		Covered		
TC0005	5	1		Covered		
TC0006	5	1		Covered		
TC0007	5	1			Covered	
TC0008	5	1			Covered	
TC0009	5	1			Covered	
TC0010	5	1				Covered
TC0011	5	1				Covered
TC0012	5	1				Covered

Execution Status: Not Covered | Covered |  - Partially Passed |  - Passed |  - Failed |  - Blocked

Requirement Tags: 1: ADD | 2: DIVIDE | 3: MULTIPLY | 4: SUBTRACT

Requirement Coverage

Copyright by TestOptimal ©2017 All Rights Reserved  
Contact Amy Xia: xia.amy@testoptimal.com



# Test Cases generated after run the model

Manual Test Execution - TestOptimal - Google Chrome

localhost:8888/MbtSvr/app=webrpt&name=TestCaseRpt

ProMBT 5.2.6 Manual Test Execution - Demo\_EmbeddedSys

Summary Requirements Preamble **Test Cases** Postamble Exec Status Progress Defects Help

KPI: TestCases 12

ALM Disabled Reset Reset All Save Print

Test Case List

Test Case: TC0001

Test Case Info: Length: 1; Weight: 5; Requirements: 1

Requirements: [ADD](#)

Test Setup / Pre-Conditions

Step	Action	Assert Expected Results
1	From: Start Do:	At: End Expected Results: <a href="#">passed</a> <a href="#">failed</a> <a href="#">blocked</a> <a href="#">reset</a> <ul style="list-style-type: none"><li>Check: Unknown eq 15. Record passed as Add 10 passed, received result Unknown</li></ul> Requirements: <a href="#">ADD</a> (DefectAdd)

Passed  
Failed  
Blocked

Test Teardown / Post-Conditions

Tester Name: \_\_\_\_\_ Date: \_\_\_\_\_ Effort: \_\_\_\_\_

Notes: \_\_\_\_\_

Copyright by TestOptimal ©2017 All Rights Reserved  
Contact Amy Xia: xia.amy@testoptimal.com