Test embedded system developed in C or other program languages

TestOptimal

Introduction
Why choose TestOptimal for embedded system

- Requirement Coverage
- Test Coverage
- Test Automation

Test Optimal  Test Optimal  Test Optimal

Copyright by TestOptimal ©2017 All Rights Reserved
Contact Amy Xia: xia.amy@testoptimal.com
Sample embedded system developed in C

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

1. \texttt{Add(double x, double y);}
2. \texttt{Substract(double x, double y);}
3. \texttt{Multiply(double x, double y);}  
4. \texttt{Divide(double x, double y);}
// dllmain.cpp : Defines the entry point for the DLL application.

#include "stdafx.h"
#include "calculator.h"

double sum = 0;

void Init()
{
    sum = 0;
}

double Add(double x, double y)
{
    return x + y;
}

double Subtract(double x, double y)
{
    return x - y;
}

double Multiply(double x, double y)
{
    return x * y;
}

double Divide(double x, double y)
{
    return x / y;
}

BOOL APIENTRY DllMain(HMODULE hModule, DWORD ul_reason_for_call, LPVOID lpReserved)
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

Bridge???

Contact Amy Xia: xia.amy@testoptimal.com
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.
public final class EmbeddedSysPlugin extends PluginAncestor {

    private ModelMgr modelMgr;

    public static native double add(double x, double y);
    public static native double subtract(double x, double y);
    public static native double multiply(double x, double y);
    public static native double divide(double x, double y);

    static {
        String dllName = "demo_64";
        String libFilePath = Config.getRootPath() + "/lib/" + dllName + ".dll";
        try {
            TestOptimalServer.logInfo("Loading library from " + libFilePath);
            System.load(libFilePath);
            TestOptimalServer.logInfo("Loaded library " + dllName);
        } catch (Throwable e) {
            TestOptimalServer.logError("Error loading library " + libFilePath, e);
        }
    }

    @SCRIPT_METHOD
    public double add(String p1, String p2) throws NEXCEPTION {
        try {
            double p1Value = Double.parseDouble(p1);
            double p2Value = Double.parseDouble(p2);
            double result = add(p1Value, p2Value); // Invoke C function by JNI
            return result;
        } catch (Throwable e) {
            this.scriptExecutor.error("Embedded.add Error: " + e.toString());
        }
        throw new NEXCEPTION (e.getMessage() + ": " + e.getMessage());
    }
}
TestOptimal invokes functions defined in Custom Plugin
Embedded Systems

- Developed in C or other program languages

Custom Plugin

- Developed in Java to expose necessary functions for mscripts
- Mscripts invokes functions defined in custom plugin and drives the model running
public String execute(RemoteCmd remoteCmd) throws Exception {
    String cmdSyntax = remoteCmd.getActionSyntax();
    this.info("Remote cmd: "+cmdSyntax);
    this.cmdJSON = new JSONObject(cmdSyntax);
    String cmdAction = this.cmdJSON.getString("ACTION");
    this.info("Action: "+cmdAction);
    if (cmdAction.equalsIgnoreCase("MBT_start")) {
        this.info("Starting vending machine");
        return "OK";
    } else if (cmdAction.equalsIgnoreCase("launchUI")) {
        this.info("Launching the test");
        return "OK";
    } else if (cmdAction.equalsIgnoreCase("resetUI")) {
        this.info("Resetting the test");
        return "OK";
    } else if (cmdAction.equalsIgnoreCase("add")) {
        String p1 = this.cmdJSON.getString("P1");
        String p2 = this.cmdJSON.getString("P2");
        this.info("Add "+p1+" "+p2);
        try {
            double p1Value = Double.parseDouble(p1);
            double p2Value = Double.parseDouble(p2);
            double d = add(p1Value, p2Value);
            return new Double(d).toString();
            //return "add result: "+d;
        } catch (Throwable e) {
            this.error("Failed to call add function: "+e.toString());
            throw new Exception(e.toString());
        }
    }
    return "OK";
}
TestOptimal sends http requests to Remote Agent
Remote Agent

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

Requirement Summary

Requirement Execution Stats

<table>
<thead>
<tr>
<th>Status</th>
<th>High</th>
<th>Medium</th>
<th>Low</th>
<th>Other</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passed</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Failed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blocked</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No-Cover</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

Test Case Summary

Test Case Execution Stats

| Total Test Cases: | 60 |
| Failed:           | 0  |
| Blocked:          | 0  |
| Partial:          | 0  |
| **Total Test Cases**: | 120 |
TestOptimal provides requirement coverage status

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Weight</th>
<th>Steps</th>
<th>Requirement Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC0001</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>TC0002</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>TC0003</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>TC0004</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>TC0005</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>TC0006</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>TC0007</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>TC0008</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>TC0009</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>TC0010</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>TC0011</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>TC0012</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Copyright by TestOptimal ©2017 All Rights Reserved
Contact Amy Xia: xia.amy@testoptimal.com
Test Cases generated after run the model

TC0001 (1)
TC0002 (1)
TC0003 (1)
TC0004 (1)
TC0005 (1)
TC0006 (1)
TC0007 (1)
TC0008 (1)
TC0009 (1)
TC0010 (1)
TC0011 (1)
TC0012 (1)

Test Case Info: Length: 1, Weight: 6, Requirements: 1
Requirements: ADD
Test Setup / Pre-Conditions

Step | Action | Assert | Expected Results | Processed
--- | --- | --- | --- | ---
1 | From: Start | | | Processed
   | Do | | passed failed blocked reset | Processed
   | Expected Results: passed failed blocked reset |
   | * Check Unknown eq.15 Record passed as Add 10 passed, received result Unknown Requirements: ADD | (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