Model based Testing using MS Spec Explorer

Problem Statement
Model-based testing is popular in software industry.

In this project, you are given a buggy binary of a TFTP client implementation. You are supposed to identify the bugs in the implementation through model-based testing using Microsoft Spec Explorer. Through the project, you will achieve the following:

- Understand model-based testing and the workflow of Spec Explorer;
- Install and configure Spec Explorer and learn how to use it to do simple model-based testing;
- Gain experience in using Spec Explorer to find bugs.

Spec Explorer
SpecExplorer is a model−based testing tool. It is an add-on to the Microsoft Visual Studio integrated development environment. In the tool, models can be directly expressed by C#.
A script in a special language called Cord (Coordination Language) is often composed by the user to combine models, generate tests and configure test generation strategies. The following references provide a good starting point for learning Spec Explorer.

- Install and configure Spec Explorer
  We assume you have installed Visual Studio 2010 on your machine. If you don't have this version, please go to the link listed in the appendix to download and install a free copy for Purdue CS students. For installing and configuring Spec Explorer, please follow these steps.

- Learn how to use Spec Explorer
  We suggest you to first visit the MSDN page for Spec Explorer and its team blog to get a general understanding on how Spec Explorer works and how to use it. Then, you can selectively watch the video classes provided by the development team of Spec Explorer. When you start to build up the model, you can also use a wizard provided by Spec Explorer to get assistance in the modeling and testing process. You can follow the instructions in this MSDN page. Besides, Spec Explorer provides several toy examples once it is successfully installed. You can find the directories of them by checking this MSDN page. If you want to learn more about the theoretical basis of Spec Explorer, this paper is a good starting point.

TFTP Protocol
Trivial File Transfer Protocol (TFTP) is a simplified version of the File Transfer Protocol (FTP). It is a simple protocol to quickly copy files, rather than supporting all of the features of the FTP protocol. For example, when a diskless workstation is booted, we only need to download the operating system image and configuration files. TFTP is designed for this kind of scenarios. The detailed description can be found in the wiki page of TFTP.

Project Instructions
You are provided a buggy TFTP client binary (DLL only). It is a simplified TFTP client to fit our project objectives. You are also provided the file that contains the declaration of APIs involving state transitions. The detail steps are:

1. Create a VS 2010 C# Spec Explorer testing project and follow the modeling guidance in the remaining steps. Detailed instructions: 1, 2.
2. Create a test adaptor called TFTPclientAdaptor and import the given DLL to this adaptor. You will find the declarations (states and APIs) of the TFTP client DLL here. The given file contains the detailed description of the APIs. Please plot a finite state machine for this TFTP client implementation based on the states and state transition APIs. You can draw it in Visio or other

drawing software. You should read this to understand this step.
3. Write the model file and the cord script to generate test cases.
4. Run the test cases to find the bugs
5. Write the test report. In this report, please list the bugs you have found. For each bug, please list the test case ID that exposes it and briefly describe why the test case fails.

Submission

1. By 9/15, 11:59pm, you should finish installing Spec Explorer, constructing the model and drawing the finite state machine of the TFTP client implementation. Please submit the VS2010 working directory (zipped) of your test project, which should contain the model code (1.5 points), the cord script (1.5 points). You also need to submit the finite state machine plot (2 points). You need to define a machine in your cord script which can generate a new finite state machine. This finite state machine can fully cover all the paths in the your plotted finite state machine.
2. By 9/26, 11:59pm, please finish the model construction and test case generation. Please report the bugs which are found by running the generated test cases. Your Spec Explorer code (model, cord script and adapter) and the generated test cases MUST run without any error. Please submit the VS2010 working directory (zipped) of your test project, which should contain the model code (3 points), the cord script(3 points) and the adaptors for the provided APIs (2 points), and the final report (2 points).

The total is 15 points. We will use Purdue Blackboard to collect submissions. Please compress all the submitted files into one zip archive with your purdue career account as the name. Each student should submit one copy before the deadline. If you work in pairs, please concatenate you and your partner's purdue career account as the file name. No paper based submission will be accepted. Late submission rules are listed on the course website.

Where to Get Help

Dong Su (su17@purdue.edu) is managing this project. His office hour is Monday 1pm -- 2pm at LWSN 2161 #16. We recommend you to use the Piazza for sharing your questions and get answers faster. Every day our instructors check the Piazza several times to process students' questions.

Appendix:

- Visual Studio and C#
  Visual Studio is Microsoft's GUI development environment. You will need it for all the development aspects of this project. In this project, we suppose you will use the 2010 Professional version. If you are a CS student and do not have it on your machine, you can download one in here.

  We assume that you know some basics of VS2010 IDE, like, create a solution, import a DLL, compile/build/debug a solution. This MSDN page is a very good tutorial for these skills.

  You will mainly use C#. The software under test and the model are all written in C#. If you are not familiar with the basics of the C# language, just learn it. The Microsoft Developer Network (MSDN) website has a good tutorial. You do not need to become an expert of the language, you can start the project by having some basic knowledge of the following:
  - Expressions, control flow, data types, operators, exception handling
  - Namespaces, interfaces, events, delegates
  - Program structure – Main() entry point, processing command line arguments
  - Basic utility classes: collections, iterators

- TFTP Server and Setting up the testing environment on your machine

If you do not have any access to a TFTP server, you can install the WinAgents TFTP server on your Windows system and do the configuration as follows:

In **Connect to TFTP server** dialogue, choose **Local server**, **Connect as Current Windows user**, check **Save login information for further use**, click **OK**. In the **Server** maue, **Virtual TFTP folders**, **Manage Repositories …** , Click **Add** to add the root directory of your TFTP server.

This is a simple example on using the TFTP client API to upload and download a file.

```
static void Main(string[] args)
{
   TFTPClient t = new TFTPClient("127.0.0.1"); // connec to the local server.
   t.Put(@"rm-1.txt", "loc-1.txt");
   t.Get(@"rm-1.txt", "loc-2.txt");
}
```