Team Schfifty-Five – Deliverable #2

Jason Daniel
Brett Ostwalt
Chad Hobbs
Drew Rodman
Jake Wisse

The testing process

The system testing process will consist of testing several components of the OpenRemote Controller software suite. The subsystems of the Controller software to be tested will be the Panel interface system, the read/write system, the sensor status interface system, and the xml file generation system. The testing process will follow the specifications and requirements laid out in the rest of this report.

Requirements traceability

No official user requirements document is published for contributors to OpenRemote, so user requirements must be derived from the user and developer documentation and possibly the source code itself. That said, it is likely that the team will be coming up with a version of the user requirements that can be used in determining necessary test cases. These tests can then be logically grouped and the output of the testing suite evaluated to determine whether or not user requirements are satisfied by the current version.

Tested items

There are three major components to the OpenRemote software suite, the cloud based Designer, the app/browser based Panel, and the server based Controller. The scope of this test plan will include testing the OpenRemote Controller 2.0 software. In order to properly test the Controller software, other components of the software suite may be necessary.

Testing schedule

Currently the testing schedule is dependent upon the timetable laid out by the course. We are allowing for one week to implement all test cases, record all results, and analyze the records for effectiveness and application to the development cycle. One week should allow for any constraints placed upon the project and a fair amount of time to handle unforeseen circumstances. Given that the third deliverable is not required until October 30th, there should be a sufficient buffer in the event that any unforeseen circumstances do actually occur, allowing team members to consolidate on the best course of action and to complete the necessary work.

Test recording procedures

Recording the output of the testing suite execution should be simple since part of the requirements include outputting the results of the tests to a human-readable file format. In order to perform regression testing and trace the stability of the project, these files can be archived and analyzed either manually or with a custom script in order to highlight any broken tests or regressions. The test suite output will also provide details about which tests were executed and note all successful runs, skips, errors, or failures which will allow for a manual audit to evaluate whether all tasks were successfully carried out.

Hardware and software requirements

Software Requirements:
Unix or Debian-based Linux
Java JRE 6 or newer
OpenRemote Controller 2.0

Hardware Requirements:
448mb RAM, (384mb for selected Operating System, 64mb for JRE)
Single core 400Mhz CPU or greater (estimated)

Constraints

Foreseen constraints placed upon the testing process will be the absence of one team member for a week during the testing process. This should not create a critical issue but it is worth noting.

System tests

Test Case 1: OpenRemoteController's server architecture includes a class called IPAutoDiscovery Server, which listens for any UDP packets and, after receiving one, spawns a new thread for an IPResponseTCPClient instance to handle further connections. A test will be needed to ensure that the server successfully spawns the client instance and that the client is accepting connections.

Test Case 2: OpenRemoteController version 2.0 uses HTTP/REST as a response type for its application. The result of the request is an application/xml file. Among the available requests is 'Request Panel Identity List'. This essentially requests local panels from the localhost and returns any found. This API specifies that there are no parameters and returns an application/XML files. There are two possible errors that are documented. A test to make sure errors are correctly thrown shall be made, and if no errors are tested then correctness in the resulting app/xml file shall be tested elsewhere. The expected errors are: Error Code: 424 which indicates an invalid panel.xml, and Error Code: 426 which is the panel.xml is not found.

Test Case 3: OpenRemoteController version 2.0 returns sensor status based on a GET request. The request requires a single parameter included in the GET request when issued to the localhost. The result of the request should be an xml file including the status of the sensor with it's condition, including an on or off status, in a predetermined form. There are seven different errors that can be thrown, Error Code:418 (Command Build Error), Error Code:419 (No Such Component Error), Error Code:420 (No Such Command Builder Error), Error Code:422 (controller.xml Not Found Error), Error Code:423 (No Such Command Error), Error Code:424 (Invalid controller.xml Error), or Error Code:429 (Invalid XML element Error). A test to make sure that the Command Build Error is generated when it is supposed to shall be made and implemented.

Test Case 4: OpenRemoteController Fail-over (Round-Robin) Group Member Service will return all cluster group members' Controller URLs. The Group Member Service will respond to a GET request issued to the localhost. The Controller will return a list of the URLs in an XML form. There are four errors that could be generated, Error Code:450, (Failed to start tcp server), Error Code:451 (Failed to start udp server), Error Code:452 (Failed to establish udp client), and Error Code:453 (Invalid groupmember service url). A test to make sure that a tcp start failure is accurately reported shall be made and implemented.

Test Case 5: The Sensor Polling Request is used for delayed HTTP request on server side, and will only return new status when the status of any sensor in sensor_id list is changed, or return 504 response code if no change occurs within 50 seconds. This is done via a GET request to the localhost and returns a predetermined XML form. The error codes returned are Error Code:418 (Command Build Error), Error Code:419 (No Such Component Error), Error Code:420 (No Such Command Builder Error), Error Code:422 (controller.xml Not Found Error), Error Code:423 (No Such Command Error), Error Code:424 (Invalid controller.xml Error), Error Code:429 (Invalid XML element Error). A test to make sure that No Such Component Error is properly reported shall be made and implemented.