OpenBazaar - Test Report

The Fair Traders
Daniel Mandel - 1303865 - mandeldr
Shandelle Murray - 1303109 - murras25
Connor Sheehan - 1330964 - sheehacg

Friday, November 27th, 2015

Abstract

This report contains the test results for the OpenBazaar redevelopment project, for McMaster University SFWR 3XA3.

Contents

Introduction

1 Objective

The objective of this document is to report the test results of the OpenBazaar application, revision zero. Testing at this phase is used to demonstrate the application's usability and to improve the robustness of the product. Developing a test suite will ensure that changes in one section of code do not cause bugs in other sections. Further, if the secret of a module is changed for any reason (improved performance, new technology, etc), a pre-defined test framework will ensure the application can function after the change.

2 Approach

Our approach involves the use of the `unittest2` library that Python has available. Test cases will be written using the same design as any NUnit or JUnit implementation (Arrange Act Assert) [1]. This means that all variables at first will be created, manipulated, and then verified. This is the most effective way of organizing unit tests and is extremely modular. All test cases will be run from the command line by simply running the test fixture (ex `python OBTests.py`)

3 Definitions

Black Box Testing:	The process of testing a function or module without taking the internal implementation into account.
White Box Testing:	The process of testing which takes the internal implementation into account, and ensures its correctness.
Usability Testing:	The process of evaluating a program by testing it on users. This provides direct input on how users will interact with the system.
Ricardian Contract:	A contractual agreement for a trade which aims to be exact and unchangeable, generally used for the purpose of issuing digital currency. Any contract is comprised of several modules which provide information about the interaction, the terms for which a value is redeemable, as well as a hash of the data to ensure that the contract is not changed retroactively. Ricardian contracts are human readable as well as machine parsable.
Kademlia DHT:	A distributed hash table for decentralized peer-to-peer networks. This is the infrastructure used in the OpenBazaar application.

Test Results

4 Black Box Testing

Initialization Module Black box testing for the initialization module was performed by testing the `initialize_bazaar` method. This method should be called on the first execution of OpenBazaar. After this method call completes, the identity module should contain several files related to GPG keys and an identity pickle file. The node module should also contain a pickle file.

Identity Module

5 White Box Testing

This section contains all unit test results that do not cover usability, performance or security. This is because the internal design of the software is known to the tester.

`Class: RicardianContractTests`

- - Test Case Name: `test_constructor`
  - Initial State: Nothing initialized
  - Input: `RicardianContract(contract_dict, seller_settings)`
  - Expected Results: RicardianContract Object Instance
  - Results Match: Yes

- - Test Case Name: `test_get_module`
  - Initial State: testRicardianContract, `seller_settings`, `contract_dict` objects initialized
  - Input:
    - tradeDict = testRicardianContract.get_module('trade')
    - price = tradeDict['price']
    - item_name = tradeDict['name']
    - keywords = tradeDict['keywords']
  - Expected Results:
    - `keywords == 'Head'`
    - `item_name == 'Hat'`
    - `price == "50"`
  - Results Match: Yes

- - Test Case Name: `test_contract_hash`
  - Initial State: testRicardianContract, testRicardianContract2 objects initialized
  - Input:

    * `hash1 = testRicardianContract.contract_hash()`
    * `hash2 = testRicardianContract.contract_hash()`
    * `hash3 = testRicardianContract2.contract_hash()`
  - Expected Results:
    * `hash1 == hash2`
    * `hash1 != hash3`
  - Results Match: Yes

- - Test Case Name: `test_get_dict`
  - Initial State: `contract_dict`, idthing, `seller_settings`, and testRicardianContract objects are initialized
  - Input: `type(testRicardianContract.get_dict())`
  - Expected Results: `type(testRicardianContract.get_dict()) == dict`
  - Results Match: Yes

`Class: InitializationModTests`

- - Test Case Name: `test_create_GUID`
  - Initial State: `setUp()` is called generating list of ten GUID's
  - Input: `len(guids) == len(set(guids))`
  - Expected Results: True
  - Results Match: Yes

- - Test Case Name: `test_initialize_Bazaar`
  - Initial State: setUp() is called generating a list of ten GUID's
  - Input:
    * `self.assertTrue(os.path.isfile('identity/identity.p'))`
    * `self.assertTrue(os.path.isfile('identity/pubring.gpg'))`
    * `self.assertTrue(os.path.isfile('identity/random_seed'))`
    * `self.assertTrue(os.path.isfile('identity/secring.gpg'))`
    * `self.assertTrue(os.path.isfile('identity/trustdb.gpg'))`
    * `self.assertTrue(os.path.isfile('node/node.p'))`
  - Expected Results: True
  - Results Match: Yes

- - Test Case Name: `gen_keys_test`
  - Initial State: setUp() is called generating a list of ten GUID's
  - Input: `self.assertTrue(len(self.key_list) == len(set(self.key_list)), "Generated keys are not all unique")`
  - Expected Results: True
  - Results Match: Yes

6 Usability Testing

Usability testing for OpenBazaar is one of the most important phases of development. Concurrently, it is also one of the most difficult to develop adequate metrics for. The goal of OpenBazaar is to develop a decentralized trade medium that can compete with centralized services already in use. [2] To determine if OpenBazaar could be considered a viable replacement for such a service, testers were asked to proceed through the process of adding an item for sale, attempting to buy an item and messaging users.

7 Performance Testing

Several components of the OpenBazaar application were tested for their performance and reliability.

8 Security Testing

Security testing of OpenBazaar is potentially the most important variety of testing due to the nature of the application. OpenBazaar must be secure in several different areas, specifically secure communication between network nodes and local data security. A person who wishes to perform an attack on the OpenBazaar could do so by intercepting communications between nodes (a man in the middle attack), hacking the locally stored user information, or spoofing the identity of a trusted user to gain access to privileged information.

Test Summary

9 Changes Derived from Tests

After running through all of our tests, some changes were necessary. The following is a list of documented changes to our code after testing:

- Class: RicardianContract
- Test: test contract hash
- Output: Hash Object
- Expected: String

After discovering that the contract hash was returning the wrong object, we went back and reimplemented the contract hash function for the RicardianContract class. On testing of the RicardianContract module, it was discovered that due to dynamic and duck typing, the `contract_hash` method was returning a full hash object instead of the hexadecimal representation as desired. This test failure was fixed by forcing the object to return the value of `hexdigest` instead of `__str__`.

References

[1]

[2] The Fair Traders. Openbazaar redevelopment - requirements. Technical report, McMaster University, 2015.