OpenBazaar Redevelopment - Design Document

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

November 12, 2015

Abstract

This documents outlines design for the OpenBazaar redevelopment project.

Table of Contents

Revision History **2**

Introduction **4**
0.1 Purpose 4
0.2 Scope 4

Anticipated and Unlikely Changes **4**
0.3 Anticipated Changes 4
0.4 Unlikely Changes 5

Module Hierarchy **5**

Connection Between Requirements and Design **6**

Module Decomposition **7**
0.5 Hardware Hiding Modules 7
0.6 Behaviour Hiding Modules 7

Traceability Matrix **9**

Use Hierarchy Between Modules **9**

Detailed Timeline **11**

Revision History

Revision Number	Revision Date	Description of Change	A
1	November 4th, 2015	Created Revision History	Dan
2	November 6th, 2015	Added to Introduction, added numbering, created tables	Shand

Table 1: Table to capture the history of the document

Introduction

0.1 Purpose

The purpose of this document is to describe the implementation of the OpenBazaar that was described in the Software Requirements Specification (SRS) document completed earlier this semester. It aims to outline a design that will meet all of the functional and non-functional requirements described in the SRS. It is also meant to be a template for creating the Module Interface Specification document, MIS, which will describe the modules in further detail.

The design principle being used to implement this project is the principle of information hiding which was first described by David Parnas. The idea behind this design strategy is that each module contains some secret, essentially hiding a design decision from the rest of the system. As a result of this method of modularization, aspects of the system that are likely to change are hidden within a module and, when changed, do not affect the rest of the modules. This is important for any software design as technology is constantly evolving and software often needs to be updated in order to remain relevant.

This document is intended for future developers and designers who wish to improve or better understand the design of the OpenBazaar. It is organized into sections of anticipated and unlikely changes to the design, a description of the module hierarchy, a decomposition of each module in the design, and traceability matrices demonstrating the connections between modules and requirements as well as modules and anticipated changes.

0.2 Scope

The purpose of this project is to design and implement OpenBazaar, a free, open market run through a peer-to-peer network that aims to replace centralized services such as eBay or Amazon by providing a means in which to participate in online trade. Major users of the OpenBazaar include buyers, sellers, and notaries. This document describes the implementation details of all major functions that create the OpenBazaar, from every type of user's perspective: buyers, sellers, and notaries.

Anticipated and Unlikely Changes

This section is intended for all possible changes that may occur to the system. They will be listed in order from most likely to least likely.

0.3 Anticipated Changes

AC1 The hardware and operating system the OpenBazaar runs on

AC2 The algorithm to generate users public, and private keys

AC3 A user's generated public and private keys

AC4 The algorithm to search for nodes on the network

AC5 Personalization options for a user's market

AC6 Personalization of a user's search preferences

AC7 The user's currency, (i.e. Bitcoins to another currency)

AC8 The user's current role (buyer, seller, notary)

AC9 The user's location and IP Address

AC10 The user's Bitcoin wallet information

AC11 The user's GUID

AC12 The user's market information (i.e. items, price, pictures, description etc)

AC13 The user's personal settings (i.e. display picture)

AC14 The user's digital signature

AC15 The user's shipping information

AC16 The status of a contract (i.e. active, published)

0.4 Unlikely Changes

The following are aspects of the design that are unlikely to change.

UC1 Bitcoin as a medium of exchange

UC2 Ricardian contract structure

UC3 Absence of Trade Restrictions

UC4 Absence of Price Restrictions

UC5 Absence of Location Restrictions

UC6 Absence of Intermediary Fees

UC7 The user's privacy, security, and anonymity

UC8 The network architecture governing peer-to-peer connections

Module Hierarchy

This section outlines the modules used in the implementation of the application. Each module is organized and decomposed according to the type of secret it contains. The following modules are represented by leaves in the hierarchy tree.

M1 GUI Module

M2 Backend Module

M3 Published Contract Module

M4 Identity Module

M5 Algorithms Module

M6 Active Contract Module

M7 DHT/Routing Table Module

M8 Settings Module

M9 Store Module

M10 Notary Module

M11 Initialization Module

M12 Node Module

Level 1	**Level 2**	**Level 3**
Hardware-Hiding Module	GUI Module	
Behaviour-Hiding Module	Backend Module	Published Contract Module DHT/Routing Table Module Settings Module Store Module Notary Module Active Contract Module Identity Module Node Module
Software Decision Module	Algorithms Module Initialization Module	

Table 2: Module Hierarchy

Connection Between Requirements and Design

This system is designed to meet the requirements that were created in the SRS document. Table 3 will trace the anticipated changes to the individual modules in order to accomplish each task.

Module Decomposition

Below is a decomposition of each module in the application design, with details of the module's provided services and encapsulated secrets.

0.5 Hardware Hiding Modules

1. GUI Module

    - **Secret:** The underlying machine hardware and operating system environment for the application.
    - **Services:** The GUI module is responsible for handling user interaction with the system. Provides controllers which take inputted data and relay to the frontend-to-backend connector for further analysis and use.
    - **Implemented by:** The module has been partly implemented via the PyQt4 framework. Implementation will be done by creating components which inherit from classes in the PyQt4 module.

0.6 Behaviour Hiding Modules

1. Identity/Backend Module

    - **Secret:** The underlying data and behaviour requirements of the system.
    - **Services:** The backend module is primarily responsible for holding all of the modules relevant to system requirements. It holds user data including all given personalization data, trade contracts and application settings. User interaction will pass through the connector module to this module.

2. Node module

    - **Secret:** Information related to the peer-to-peer networking component of the application.
    - **Services:** This module provides all data and behaviours that make a machine a valid network node.

3. DHT Module

- **Secret:** The contents and implementation of the distributed hash table and routing tables for the Kademlia peer-to-peer network.
- **Services:** The DHT module provides information about the distributed hash table used for networking. The module does node lookups and returns information about the node.

4. Published Contract Module

   - **Secret:** Implementation structure of published contracts available on the network.
   - **Services:** The Published Contract module holds information about all the published contracts known to the network node. Published contracts include information about the publisher that can be used to request detailed store, user or notary information to display.

5. Settings Module

   - **Secret:** Encryption and storage of confidential user data.
   - **Services:** The Settings Module provides the implementation and storage of user settings and personal information.

6. Store Module

   - **Secret:** Implementation of store data.
   - **Services:** Holds information about stores that have been visited on the application for future use. For example if the node is no longer on the network (machine is offline, application is not active) the store can still be visited for viewing.

7. Notary Module

   - **Secret:** Implementation of notary data.
   - **Services:** Holds information about known notaries for viewing and messaging.

8. Active Contract Module

   - **Secret:** Storage of confidential contract and ledger data.
   - **Services:** Securely stores details about contracts currently undertaken by the user. This includes all the potential roles of notary, buyer or seller.

9. Algorithms Module

   - **Secret:** Algorithms used by application.
   - **Services:** Contains and performs algorithmic calculations such as hashes, public and private key generation.

10. Initialization Module

    - **Secret:** Bootstrap procedure and node initialization information.
    - **Services:** Provides initialization steps of application for first use including pubkey and privkey generation, GUID creation and connecting the node to the network.

Description

The OpenBazaar modules are broken up into logical components, abstracting away portions of the application that do not depend on one another. The first logical decomposition of the application is to abstract the details of the graphical user interface from the details of the data implementation. Each of these respective components will run as its own thread in the application environment. The data implementation can then be manipulated and accessed by the user via interaction with the GUI. An additional connector module may need to be implemented to expose an interface for the GUI to interact with that submits and returns data for graphical display to the user.

Traceability Matrix

Below are two traceability matrices. The first demonstrates the connection between the functional requirements and modules while the second describes the connection between the anticipated changes and modules.

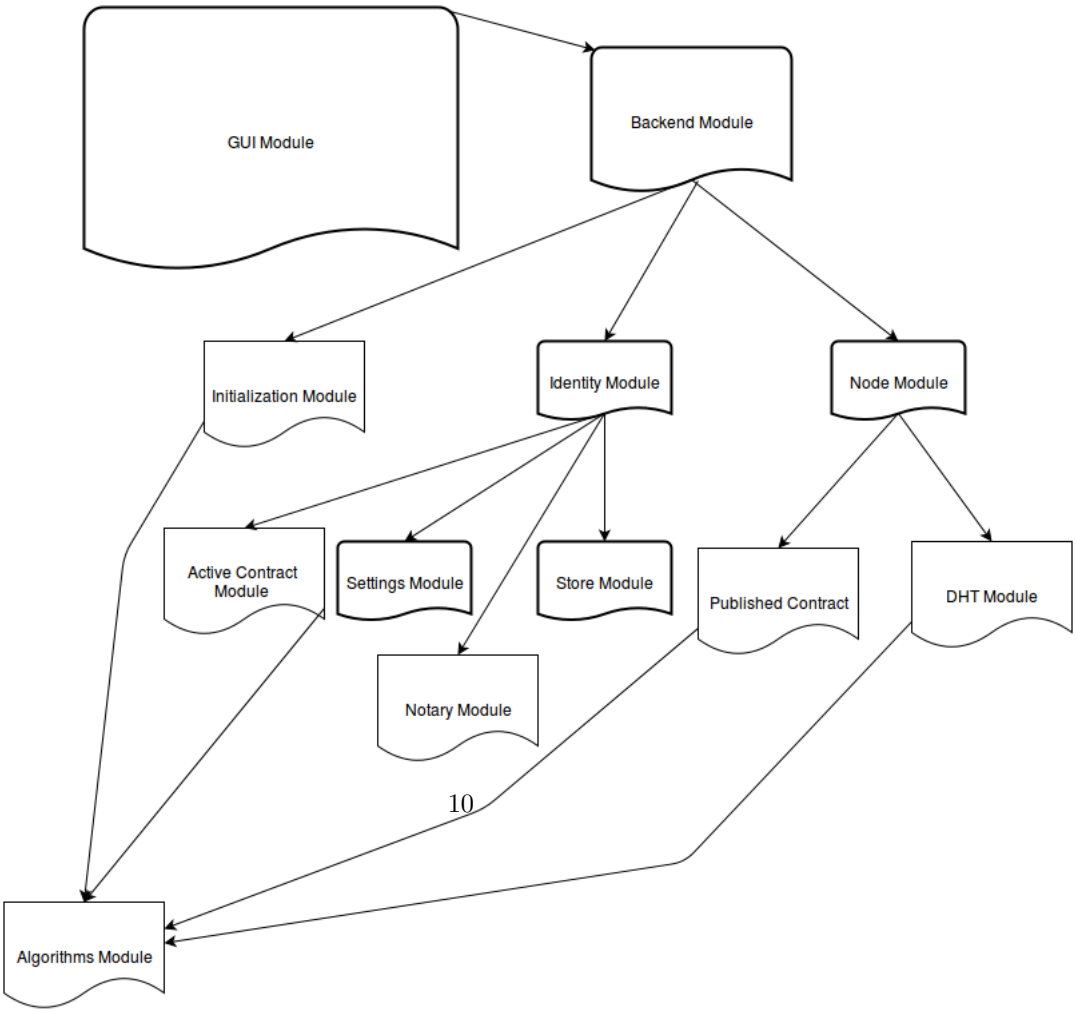
Req.	**Modules**
R1	M7, M11, M12
R2	M2, M4, M5, M7, M12
R3	M3, M6, M9
R4	M1, M9
R5	M9
R6	M3, M6
R7	M3, M6
R8	M10
R9	M7, M12
R10	M8
R11	M8, M9
R12	M10
R13	M10

Table 3: Table 2: Trace Between Requirements and Modules

AC	Modules
AC1	M1
AC2	M5
AC3	M11, M2
AC4	M5
AC5	M9
AC6	M8
AC7	M4
AC8	M12, M10
AC9	M11
AC10	M8
AC11	M11
AC12	M9
AC13	M8
AC14	M7
AC15	M8
AC16	M3, M6

Table 4: Table 3: Trace Between Anticipated Changes and Modules

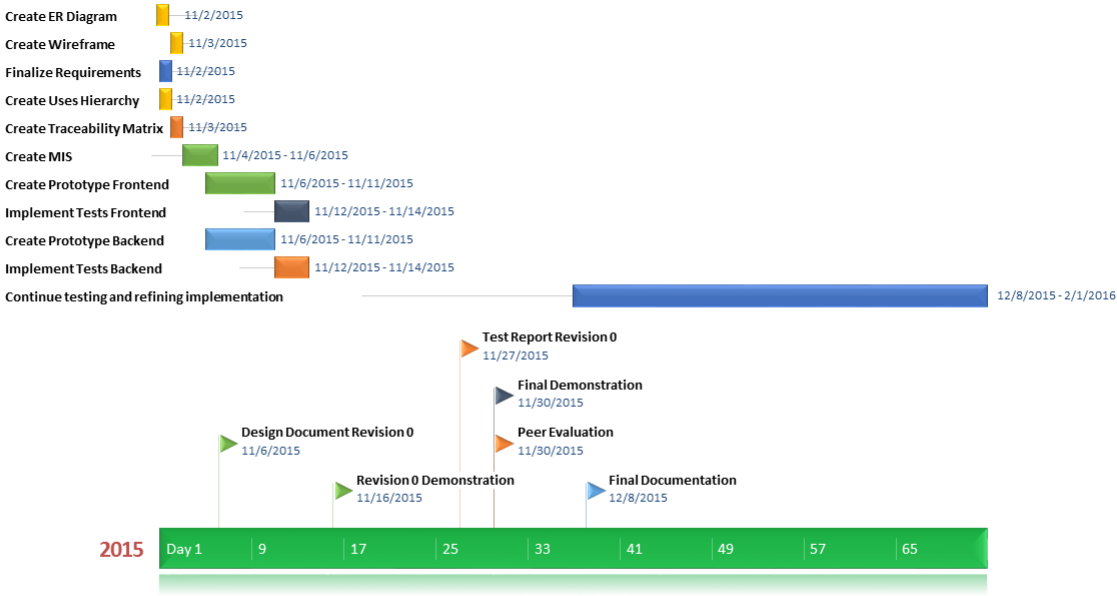
Use Hierarchy Between Modules



Above is a use hierarchy for the modules in the OpenBazaar application.

Detailed Timeline

Gantt Chart



Included above is a Gantt chart outlining project deadlines. The chart describes in detail the dates that the implementation and documentation will have to be completed in the form of milestones. Additionally, the period of time where those tasks will be focused is described in the top section of the chart.

Pert Chart

Milestones:
M1 Design Document Revision 0
M2 Revision 0 Demonstration
M3 Test Report Revision 0
M4 Peer Evaluation
M5 Final Demonstration
M6 Final Documentation

Tasks:
T1 Create ER Diagram Duration 1
T2 Create Wireframe Duration 1
T3 Create Uses Hierarchy Duration 1
T4 Create Traceability Matrix Duration 1
T5 Create MIS Duration 1
T6 Create Prototype Frontend Duration
T7 Implement Tests Frontend Duration
T8 Create Prototype Backend Duration 6
T9 Implement Tests Backend Duration 2
T10 Finalize Requirements Duration 1