Cryptocurrency Wallet

Software Requirement Specification (SRS)

Gina Somara && Dane Leineke

TODO: Simulated Network Function description
TimeLocks

Table of Contents

Sections

Figures

1. INTRODUCTION

NOTE: General background and reference information

1.1 Purpose of this Document

This SRS document goals are to define the features, functions, requirements and development of an simulated hierarchical deterministic cryptocurrency wallet that will be capable of the following: creating/managing addresses via a tree-like structure, constructing/signing bitcoin transactions for a simulated network, and requesting relevant unspent transaction outputs from a simulated network. The technology in this project is formulated around the specifics of the Bitcoin Network.

1.2 Project Scope

This project will focus on the initial wallet creation. Once implemented, the wallet will be able to create and receive transactions on a simulated network. The newest BIPs (Bitcoin Information Protocols) will be utilized for maximum ‘compatibility’ with the blockchain. The wallet’s structure will offer different features of usage, dependent upon the user's specifications. **(Maybe expand)**

2. GENERAL DESCRIPTION

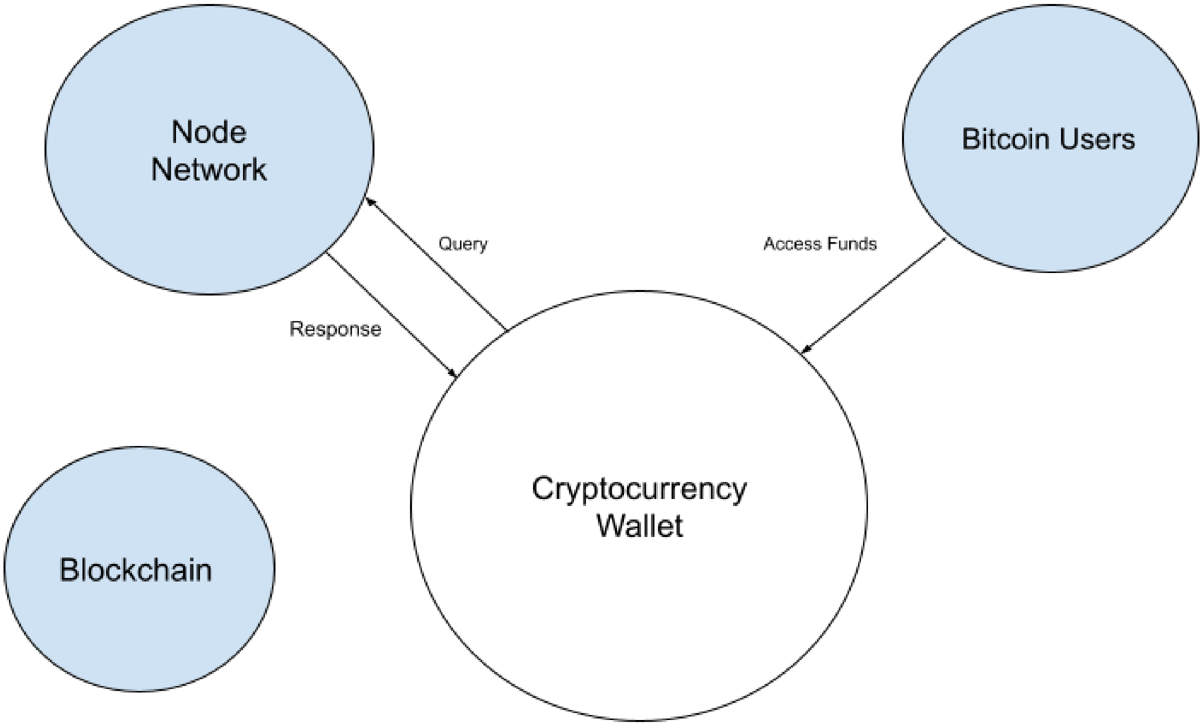
2.1 Glossary (Definitions, Acronyms, and Abbreviations)

BIP	Bitcoin Information Protocol
HD	Hierarchical Deterministic
ECDSA	Elliptical Curve Digital Signature Algorithm
FR	Functional Requirements
HMAC	Hash-Based Message Authentication Code
NFR	Non-Functional Requirement
OOA	Object Oriented Analysis
P2PKH	Pay To Public Key Hash
P2SH	Pay To Script Hash
P2WSH	Pay To Witness Script Hash
SHA	Secure Hash Algorithms
SPV	Simplified Payment Verification
Tx	Transaction
Txid	Transaction Idefinitication
UCD	Use Case Diagram
UTXO	Unspent Transaction Output
Vin	Vector (Transactional) Input
Vout	Vector (Transactional) Output

2.4 Product Perspective (System Context Diagram)

- TODO: Brief description of SCD goals
- *Everything in the diagram is assumed to be simulated and not referencing the real world

Figure 1: SCD



3. Object Oriented Analysis (OOA) –

3.1 Use Case Diagrams:

Figure 2: Parent UCD

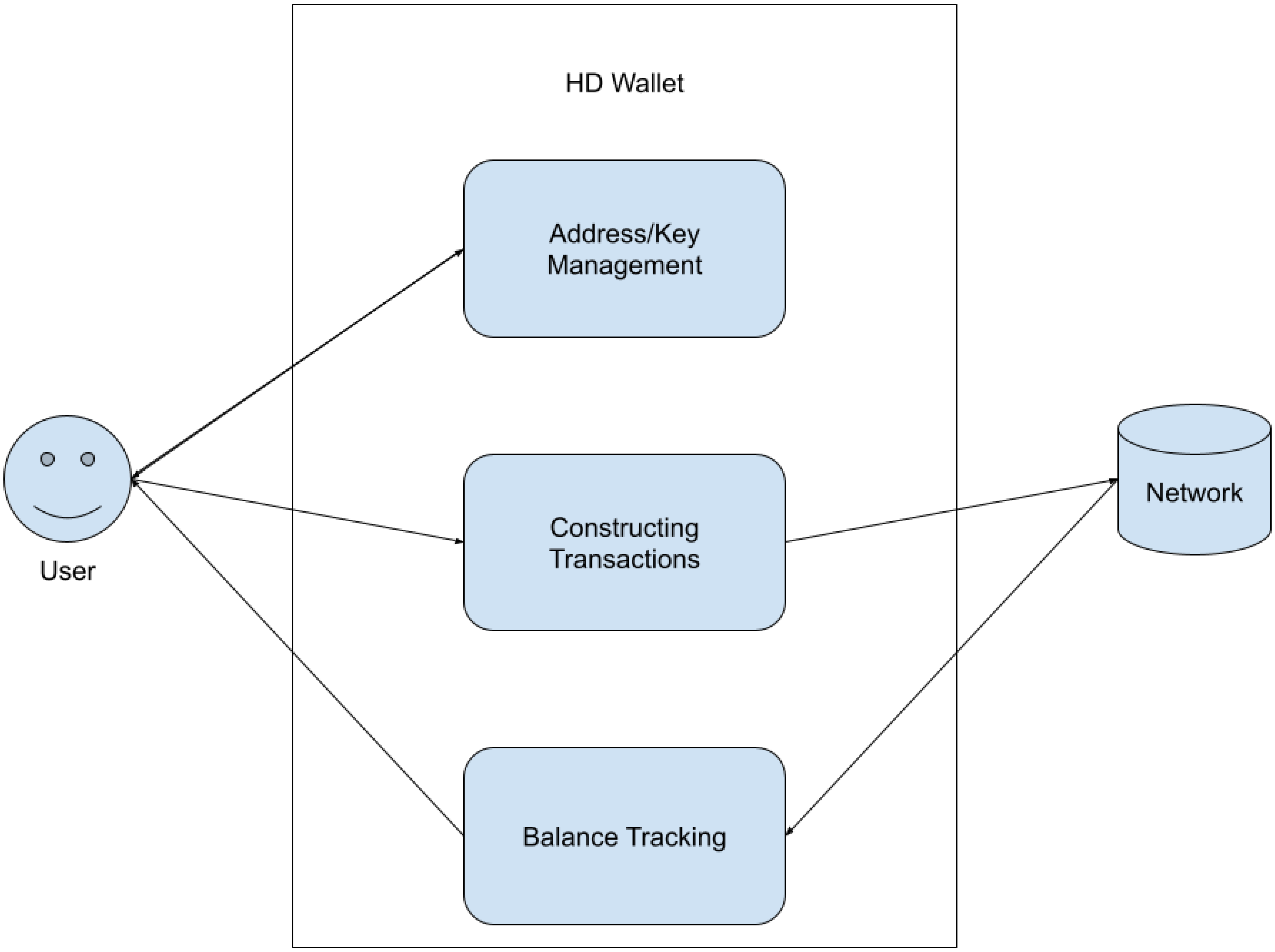


Figure 2a: Child 1

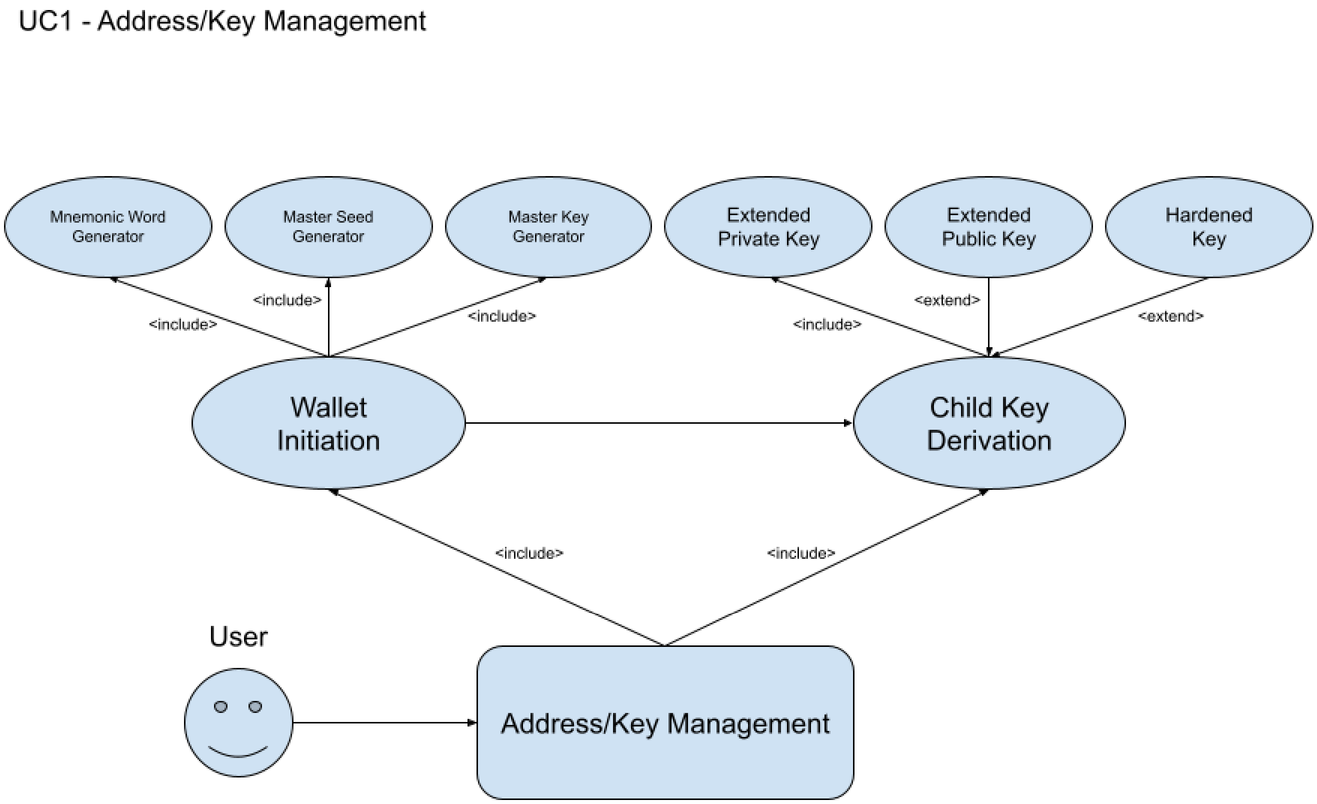


Figure 2b: Child 2

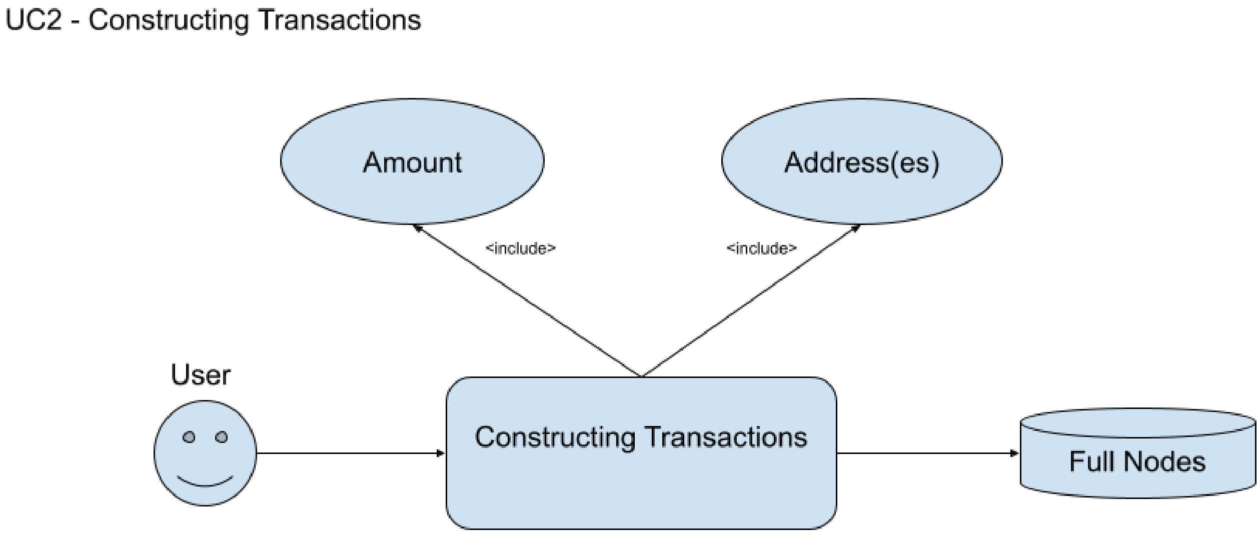
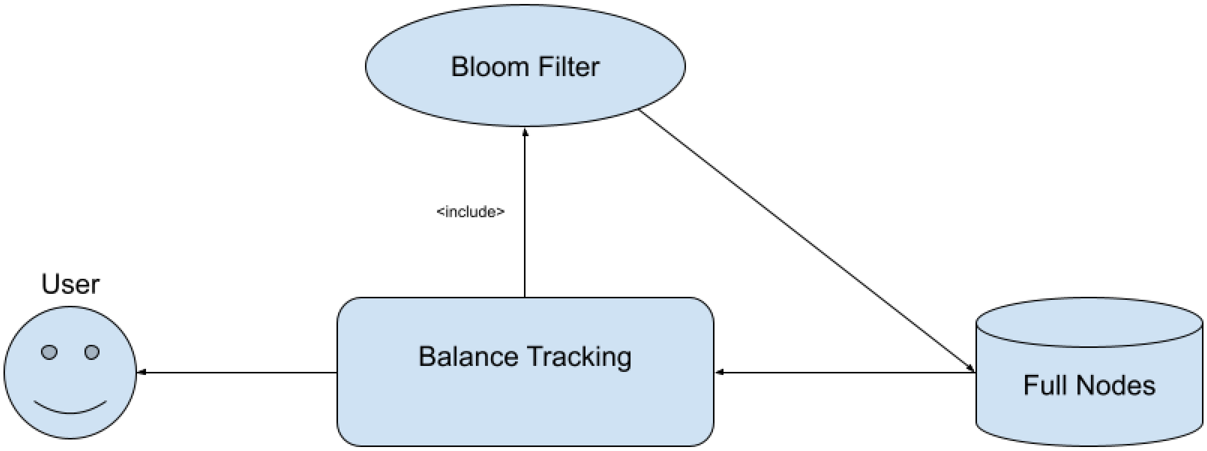


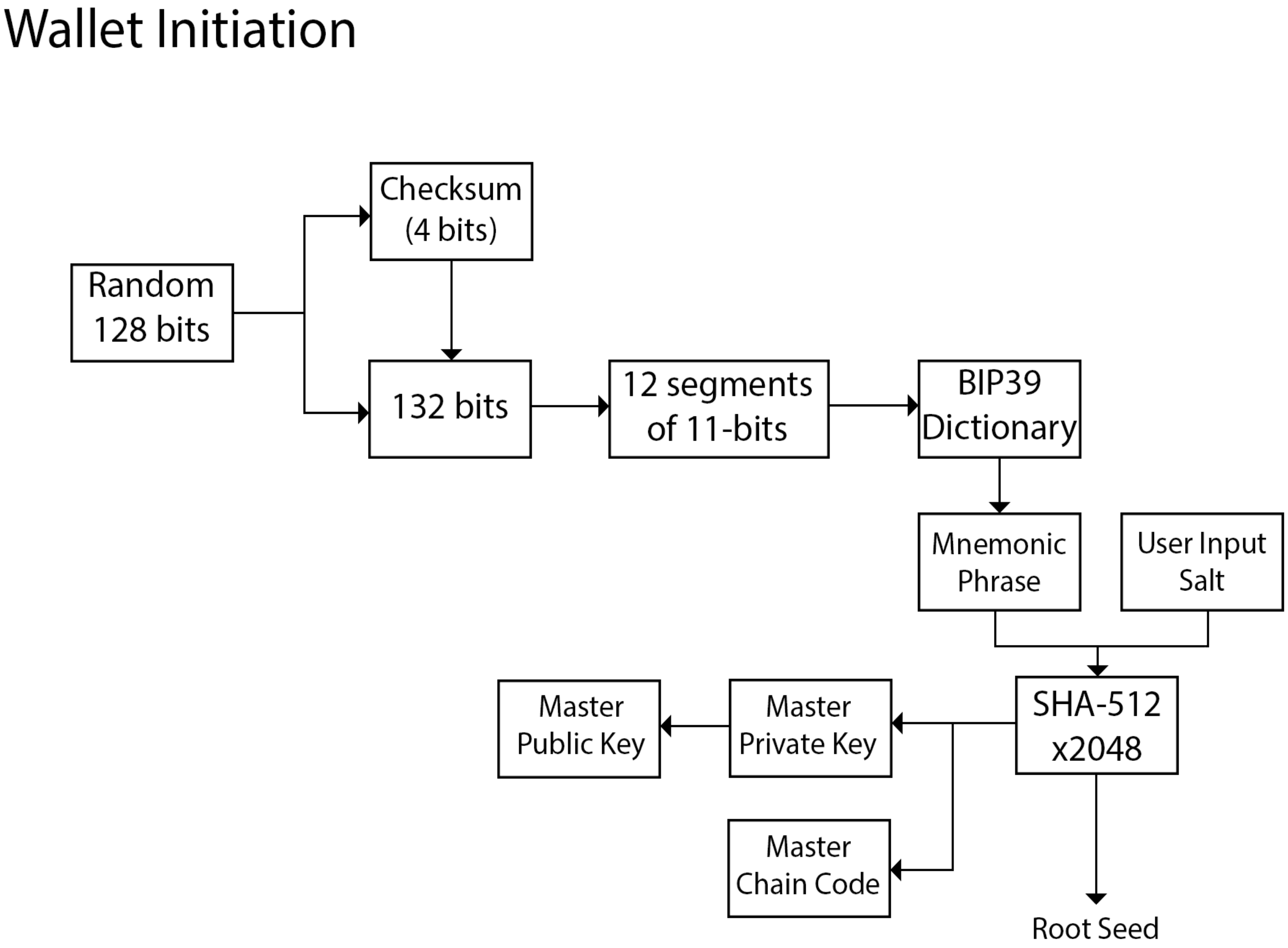
Figure 2c: Child 3

UC3 - Balance Tracking



3.2 Use Case Diagram Descriptions & Typical Flow:

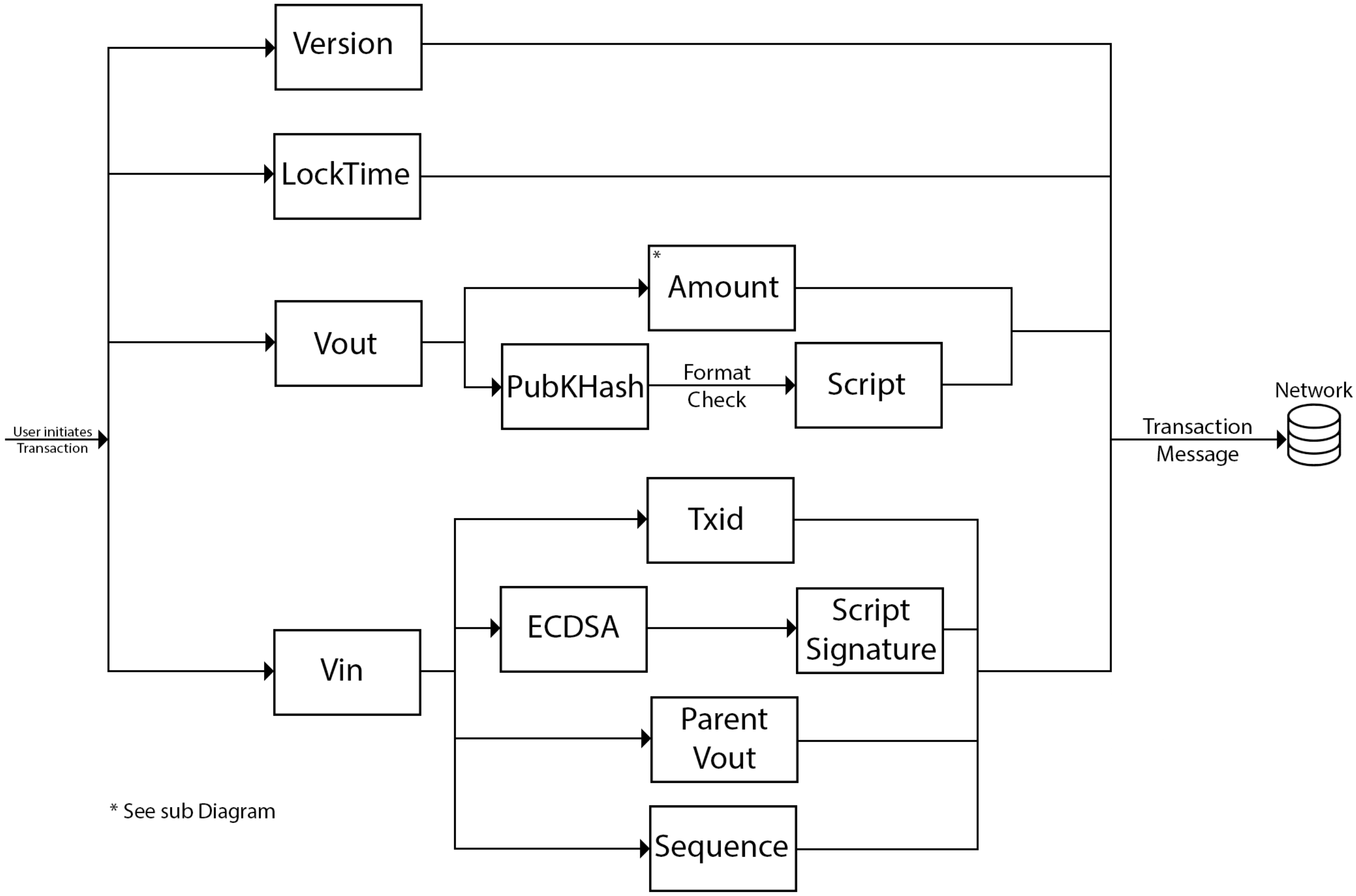
Use Case Name: Address/Key Management: Wallet Initiation
Use Case Number: UC1
Authors:
Actors:
Overview:
References: FR1, FR2.1
Related Use Cases: what other Use Cases are related to this Use Case
Typical Flow Description: (include precondition & post-condition)
Alternative Flow Description: (include precondition & post-condition)



Use Case Name: Address/Key Management: Child Key Derivation
Use Case Number: UC1
Authors:
Actors:
Overview:
References: FR1, FR2.2
Related Use Cases: what other Use Cases are related to this Use Case
Typical Flow Description: (include precondition & post-condition)
Alternative Flow Description: (include precondition & post-condition)

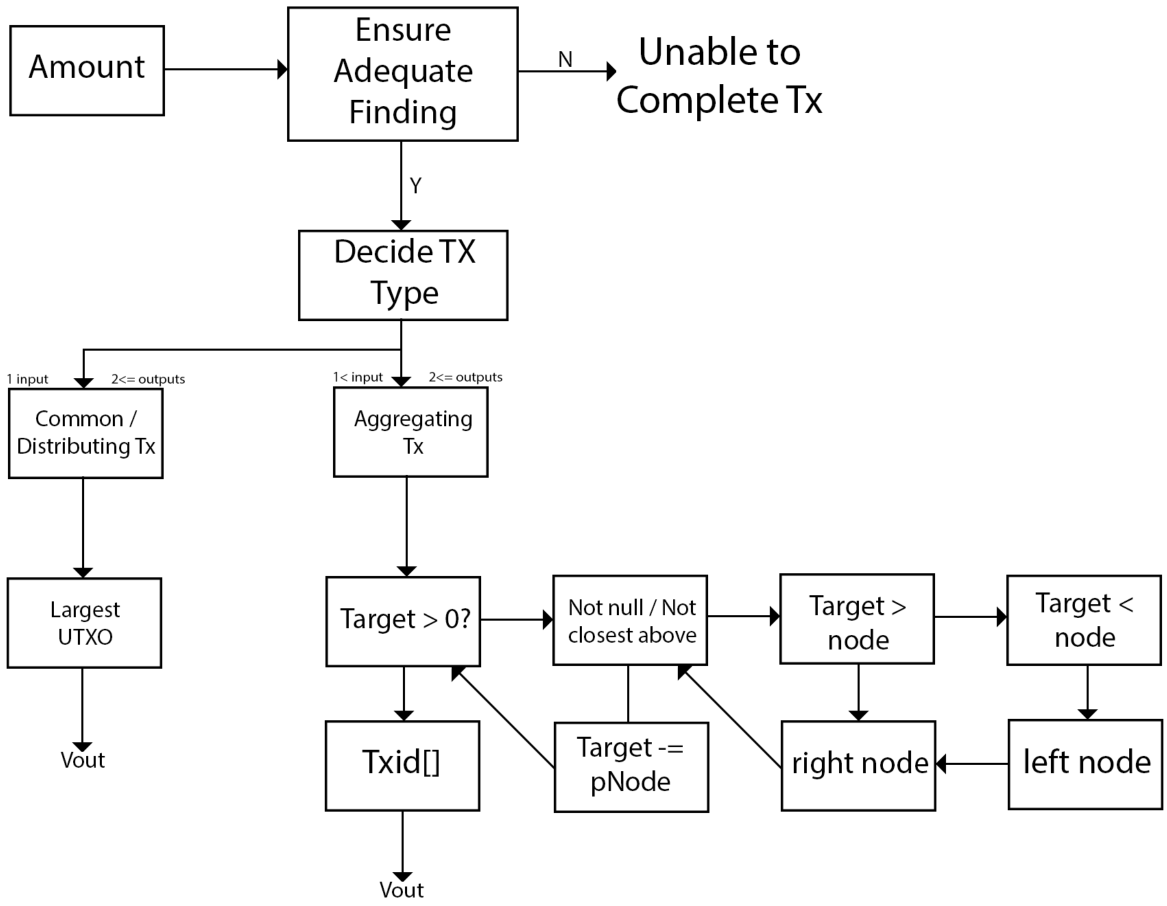
Use Case Name: Constructing Transactions
Use Case Number: UC3
Authors:
Actors:
Overview:
References: FR3
Related Use Cases: what other Use Cases are related to this Use Case
Typical Flow Description: (include precondition & post-condition)
Alternative Flow Description: (include precondition & post-condition)

Constructing Transactions



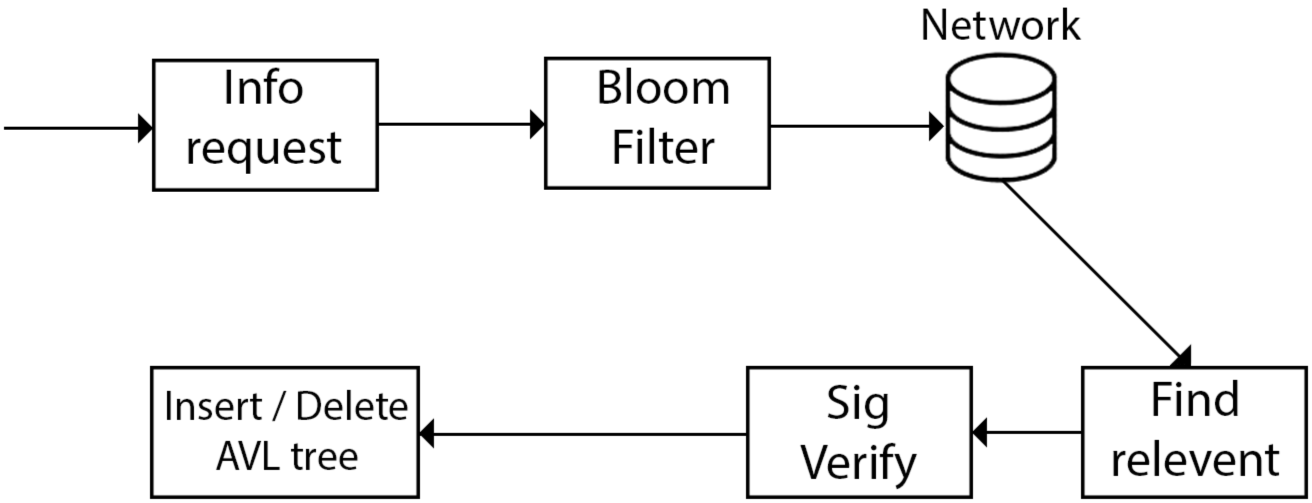
* See sub Diagram

* Tx Amount Algorithm



Use Case Name: Balance Tracking
Use Case Number: UC4
Authors:
Actors:
Overview:
References: FR4
Related Use Cases: what other Use Cases are related to this Use Case
Typical Flow Description: (include precondition & post-condition)
Alternative Flow Description: (include precondition & post-condition)

Balance Tracking



4. System Functional Requirements

FR1 - Basic Operations

FR1.1 - Hashing algorithms (SHA256, HMAC-SHA512)

FR1.2 - Elliptic Curve Multiplication

FR1.3 - Base58Check

FR2 - Wallet Data Structure

FR2.1 - Wallet Initiation

FR2.1a - Mnemonic Word Generation

FR2.1b - Master Seed/Phrase Generation

FR2.1c - Master Key Generation

FR2.2 - Key Generation Options

FR2.2a - Child Private Key

FR2.2b - Child Public Key

FR2.2c - Hardened Child Key

FR3 - Transaction Data Structure (user AND network)

FR3.1 - Transaction I/O

FR4 - Functions (either user AND nodes/network or just nodes/network)

FR4.1 - Bloom Filter (Requesting Wallet Balance)

FR4.2 - ECDSA

ECDSA

Based on the project description and the Use Case model, list all system functional requirements. ***Number the Functional Requirements (FR1, FR2, FR3, etc.) in a systematic manner.*** This section should *not* be design-oriented, a common mistake. Make sure the FRs are clear, complete and concise.

5. Specification (Detailed Description of Functional Requirements)

7.1 Template for describing functional requirements

This section builds on “Section **4**”. Complete for ***each of the functional requirement listed in section 4 the following:***

- **Purpose**
- **Inputs:** which inputs and from what sources
- **Processing:** describes the **outcome** rather than the **implementation**; include any validity checks on the data, and how to handle unexpected or abnormal situations.
- **Outputs:** the form and the destination, of the output; process by which the output is stored or destroyed; process for handling error messages produced as output.

7. Remarks or Comments

The artificially active wallet will pave the road for a future educational mobile application/website release. This release will focus on blockchain technology - wallet and transactional structures specifically - for beginners.

8. References / Resources Used

Mastering Bitcoin 2nd Edition