University of Zurich[UZH]

MASTER PROJECT – Communication Systems Group, Prof. Dr. Burkhard Stiller

Bitcoin in practice

Jeton Memeti
Stalden VS, Switzerland
Student ID: 07-722-408

Mehmet Ali Bekooglu
Oftringen AG, Switzerland
Student ID: 06-920-771

Simon Kaeser
Eschenbach LU, Switzerland
Student ID: 08-710-873

Supervisor: Christos Tsiaras, Dr. Thomas Bocek, Prof. Dr. Burkhard Stiller
Date of Submission: March 14, 2014

University of Zurich
Department of Informatics (IFI)
Binzmühlestrasse 14, CH-8050 Zürich, Switzerland

ifi

Master Project
Communication Systems Group (CSG)
Department of Informatics (IFI)
University of Zurich
Binzmühlestrasse 14, CH-8050 Zürich, Switzerland
URL: http://www.csg.uzh.ch/

Abstract

Bitcoin, a well-known peer-to-peer digital cryptocurrency, is suited for the global exchange of money via the internet with no need for intermediates as banks. However, every Bitcoin transaction needs a relatively long confirmation time of about 50 minutes so it can be regarded as valid. Therefore, Bitcoin is not applicable to use for everyday transactions such as to pay in a supermarket. Further, transaction fees will play a fundamental role in future. Hence, it is important to keep the amount of Bitcoin core transactions low.
To overcome the aforementioned time constraint, a mobile Bitcoin payment solution (MBPS) that allows instant exchange of bitcoins between users and minimizes the Bitcoin core transactions was designed, implemented, and evaluated. The presented solution consists of a centralized server in combination with an Android application, which processes the transactions by use of NFC. The conducted test run showed that MBPS is suitable for fast everyday transactions, is easy to use and people are interested in this topic. However, it also showed that people are not yet familiar with correct handling of NFC applications. Based on the evaluation results, several topics for future work were elaborated to improve the NFC protocol, the user interface, and to allow the transfer of bitcoins between multiple MBPS systems.

Zusammenfassung

Bitcoin, eine bekannte digitale peer-to-peer Kryptowährung, ist geeignet für den globalen Austausch von Geld über das Internet ohne Bankeneinbezug. Allerdings benötigt jede Bitcoin Transaktion eine relativ lange Bestätigungszeit von ungefähr 50 Minuten um als valide zu gelten. Aus diesem Grund ist Bitcoin nicht geeignet zur Verwendung bei alltäglichen Transaktionen wie zum Beispiel beim Zahlen im Supermarkt. Des Weiteren werden Transaktionsgebühren in der Zukunft eine grosse Rolle spielen, weshalb es wichtig ist, die Anzahl der Bitcoin Netzwerk-Transaktionen gering zu halten.
Um diese Einschränkungen zu umgehen wurde eine mobile Bitcoin Zahlungslösung (MBPS), welche den augenblicklichen Austausch von Bitcoins zwischen zwei Benutzern ermöglicht und die Anzahl an Bitcoin Netzwerktransaktionen minimiert, geplant, entwickelt und ausgewertet. Die präsentierte Lösung besteht aus einem zentralen Server in Kombination mit einer Android Applikation, welche die Transaktionen per NFC durchführt. Der vorgenommene Testlauf zeigt, dass MBPS geeignet ist um schnelle, alltägliche Transaktionen durchzuführen, einfach zu bedienen ist und die Menschen interessiert sind an diesem Thema. Der Testlauf offenbarte allerdings auch, dass die Leute den korrekten Umgang mit NFC Applikationen noch nicht gewohnt sind.
Aus den in der Evaluation gewonnenen Erkenntnissen ergeben sich mehrere Themen zur zukünftigen Verbesserung von MBPS. So sollte das NFC Protokoll verbessert, die Benutzeroberfläche optimiert und der Server ausgebaut werden um den Transfer von Bitcoins zwischen mehreren MBPS Systemen zu ermöglichen.

Acknowledgments

We would like to thank all the people involved in this Master Project. First of all, we thank professor Burkhard Stiller for giving us the opportunity to realize this project at the Communication Systems Group.
Many thanks also go to Christos Tsiaras and Thomas Bocek for providing the idea of this project and supporting us from the beginning to the end.
We would also like to thank Guilherme Sperb Machado for setting up and maintaining the server and Beat Rageth for helping us with administrative tasks.

Last but not least we would also like to thank the ZFV team consisting of Fabio Triulzi, Lukas Christen, and Mario Caputo for giving us the opportunity to conduct the test run at the Mensa Binzmühle at the University of Zurich.

Contents

Chapter 1

Introduction and Motivation

Bitcoin [1] is an open, fully distributed P2P digital currency that is gaining more and more popularity. However, everyday transactions such as to pay for a coffee or other kind of micro payments are not yet feasible when it comes to bitcoins. The reason for this is that the seller – or the peer which is going to receive the bitcoins for a service or good – has to wait an amount of time for the transaction to be confirmed. A transaction is confirmed when it is included in a block which is published to the network [2]. At this time, it takes around 8 minutes for a block to be created [3]. This means that Bitcoin transactions cannot be carried out instantly and both seller and buyer have to wait a quite large amount of time before closing the deal.
Based on the nature of the Bitcoin protocol and implementation, it is not enough to have only one confirmation, since a number of blocks can be rejected in favour of another branch. A transaction should only be considered as confirmed after 6 blocks verify that transaction [4]. This means that if the seller is not willing to risk losing bitcoins – even in the micro payments sector – he has to wait for 6 confirmations. Based on the time it takes currently to create one block, this requires both peers to wait more than 48 minutes before closing the deal. Hence, bitcoins in its current implementation and approach are not applicable when it comes to everyday transactions, e.g., paying in a market or in a restaurant.
However, there is the approach to just broadcast a transaction without waiting for any confirmation. This so called fast payment takes around 10 seconds [5]. But there remains the risk of double spending. Therefore, this approach is not taken into consideration for this project in order to eliminate the double spending risk for the seller.

1.1 Motivation

The motivation of this Master Project is to overcome the aforementioned time constraint and to develop a mobile Bitcoin payment method which allows exchanging bitcoins instantly. Although payment solutions such as BIPS [6] or BitcoinPAYFLOW [7] exist, this project should focus in addition on security aspects, two-way Near Field Communication (NFC), and reducing the number of transaction by introducing a clearing center.

The goal of this project is to implement an open source solution where Bitcoin payments will be possible through a mobile device. The Mobile Bitcoin Payment Solution (MBPS) will need a Bitcoin payment server where the seller and the buyer will be able to handle Bitcoin payments. For the communication between the buyer and the seller NFC [8] equipped devices will be used.

1.2 Description of Work

This Master Project covers the design, implementation, and evaluation of a mobile payment system based on bitcoins, so called MBPS. This system facilitates a payment solution between a buyer and a seller with the use of bitcoins, an intermediate Bitcoin payment server and an online wallet. What is out of the scope of this project is a Bitcoin exchange center such as Bitstamp [9]. However, interfaces between the MBPS and such a center should be implemented. Furthermore, for future purposes an interface for communication between multiple MBPSs should be considered during the design period of the project. The Master Project puts an emphasis on four core items of work: (a) a thorough requirements analysis – if possible in collaboration with a partner seller, such as Mensa UZH Binzmühle [10] – (b) system design, implementation, and integrated testing according to the identified requirements, (c) minimization of the total transactions in the Bitcoin environment, and (d) the security and the fail safety of the overall system which is an important and challenging part of this work. A high level architecture of the MBPS is illustrated in Figure 1.1.

1.3 Scope

Based on the description of work and the project goals, the following tasks targeting the required milestones need to be accomplished:

- Come up with a time and resource planning determining major tasks, detailed activities, and the respective milestones in accordance with this topic description. The time and resource plan shall map milestones and the respective activities onto available human resources in the Master Project (responsibilities), and it shall indicate when a milestone is due. Include in this plan a weekly or bi-weekly meeting with your supervisors to discuss progress and potentially needed adaptations to the planning.
- For each of the main areas of work (a) to (d) outlined, determine the suited approach and design, and document both. Approach and design in this context relate for (a) to the method of performing the requirements analysis as well as the conduct of the requirements analysis, for (b) to the complete system design, for (c) to the respective method to be used and for (d) a complete security analysis. Ensure in all parts of the system design that designs of interacting components are aligned. Start with the definition of the overall workflow (covering all components), then determine

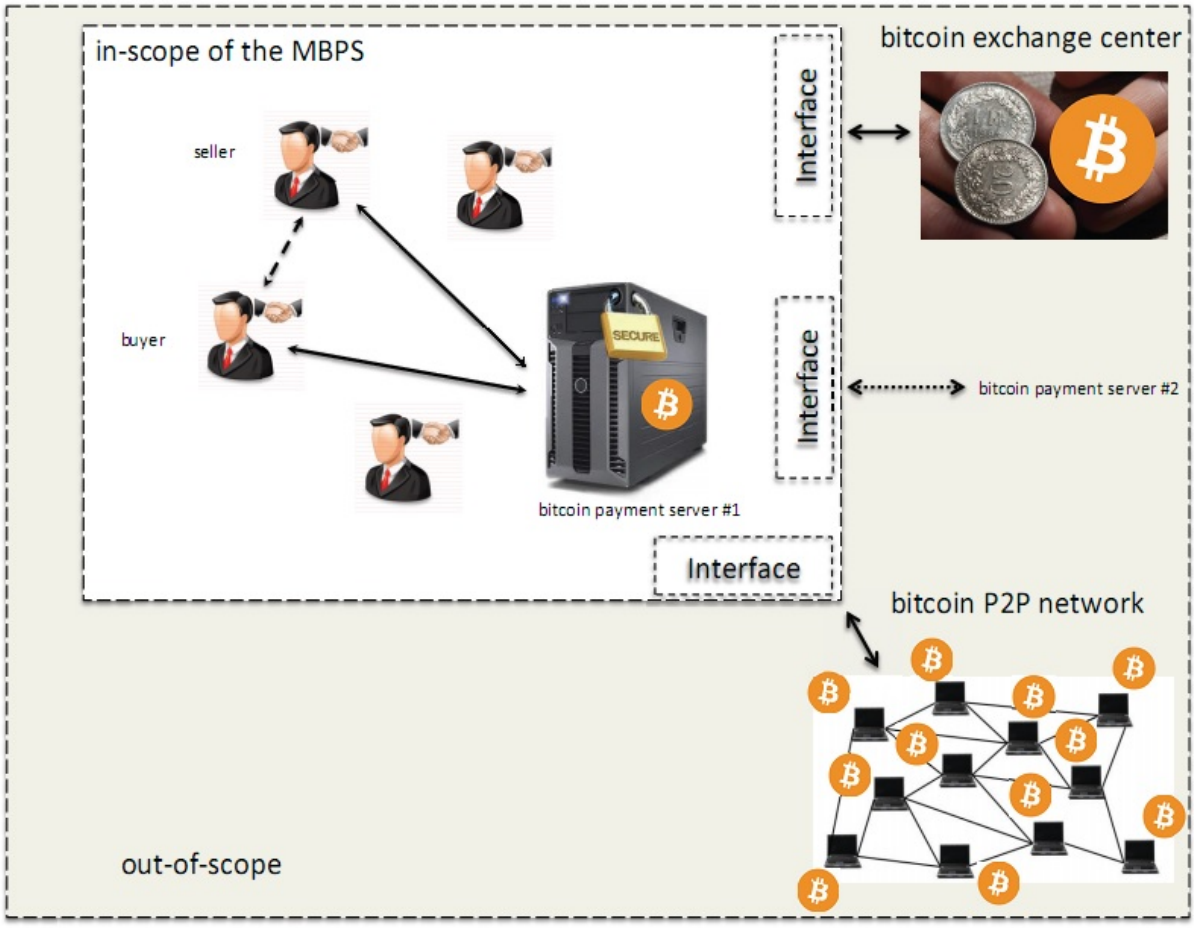


Figure 1.1: System Overview

external specifications of components (interfaces, message sequences etc.), and define the set of suited test cases (functionality evaluation is a must, technical performance evaluation good to have) for each step in the workflow. Ensure to have test cases for each component, for component interaction, and for the overall workflow covered.

- Implement your solution according to the requirements identified. Conduct the respective component-specific, inter-component, and workflow-oriented test cases. Document both the implementation as well as the evaluation based on conducted tests. The CSG test bed should be used for testing.

- Produce a distributable software package for the MBPS implemented and tested.

- An evaluation of MBPS should be carried out if possible with a seller. The evaluation needs to be carefully planned beforehand and coordinated with the seller. The planning includes to come up with a way how customers can exchange CHF to bitcoins as well as how the seller can exchange back bitcoins to CHF.

1.4 Goals

Driven by the description of work outlined, the following determine key goals for this Master Project:

- Design, implementation, and testing of the MBPS between a buyer and a seller performed, whereas the system design and its implementation shall be qualitatively assessed with respect to the applicable degree of fulfilling the requirements identified if possible in collaboration with a partner seller.
- Evaluation of the MBPS done by means of an appropriately designed, conducted, and reported study, assessing if the use of the designed and implemented MBPS is secure, fault tolerant and scalable in terms of the user base and the transactions.

1.5 Outline

The next chapters address the aforementioned goals by showing the design choices and decisions, the test run and the survey results and finally the lessons learned. Chapter 2 presents similar solutions or systems to MBPS as already mentioned in Section 1.1 and depicts the most important differences. Chapter 3 describes the design of the developed payment system by explaining each important component of the system in detail and showing also implementation details. The evaluation consisting of a test run followed by a survey are covered in Chapter 4. Chapter 5 presents the lessons learned concerning bitcoins and a mobile Bitcoin payment system in general. Finally, Chapter 6 summarizes the key findings and draws a conclusion. It also shows open issues concerning future work.

Chapter 2

Related Work

This Chapter covers related applications and systems. The goal is to avoid reinventing the wheel, i.e., building a system which already exists. Furthermore, related systems should be analysed in order to derive requirements for this project.
This project would be needless, if there were an application which meets the following requirements: (i) is an Android application, (ii) allows exchanging bitcoins instantly, and (iii) uses Near Field Communication. Section 2.1 analyses the Bitcoin Wallet with respect to these requirements. The payment service provider BIPS is covered in Section 2.2. Section 2.3 deals with BitcoinPAYFLOW, another payment service provider.

2.1 Bitcoin Wallet

Bitcoin Wallet [11, 12, 13] is a standalone Bitcoin payment application for Android devices. In contrast to desktop wallets like Bitcoin-Qt [14] it does not download the complete block chain. Instead, it uses the bitcoinj [15] library, which is a Java implementation of the Bitcoin protocol. This library allows it to maintain a wallet and to initiate transactions (send or receive bitcoins) without having a local copy of Bitcoin core. In the scope of mobile applications, it is hardly possible to store the complete block chain on the device due to memory limitations. At the time this report was written, the block chain had exceeded the size of 14 GB [16], but most devices have only 16 GB storage less the space assigned to the operating system.
Since the Bitcoin Wallet is decentralized, the wallet and the private keys are stored on the device itself. This requires that a user has to backup this sensitive information. Otherwise, a damaged device would result in losing all the bitcoins. A backup might nonetheless not prevent you from losing all your bitcoins, because Bitcoin Wallet does not require user authentication. So if someone finds your device, he can transfer the bitcoins from the device to his own wallet.
According to the application website [11] it is possible to send or receive bitcoins via NFC or QR-Codes. However, these technologies are only used to transmit the public key, i.e., the Bitcoin address. Another feature is that one can initiate a transaction even while being offline. If the counterpart is online, than a Bluetooth connection is established and

used to complete the transaction.
Bitcoin Wallet meets two out of the three requirements described above, namely (i) and (iii). However, since it is based on bitcoinj and therefore decentralized, it fully relies on the Bitcoin core protocol. Instant transactions as defined in requirement (ii) above are not possible with this approach.

2.2 BIPS

BIPS stands for Bitcoin Internet Payment System and operates as a payment service provider [6]. Primarily, BIPS is meant for merchants who sell goods or services for bitcoins. When a merchant sells something, he receives bitcoins on his BIPS online wallet. What BIPS does is then automatically transferring the bitcoins from a user's online wallet to his indicated bank account. This bank transfer is conducted once a day. However, before the bank transfer can be conducted, BIPS needs to trade the bitcoins in the currency the user has indicated. Thereto BIPS is using Bitstamp [17], one of the largest Bitcoin exchanges.
But BIPS is not only meant for merchants – normal users can use it too. A user can for example transfer money to BIPS' business bank account. When BIPS receives the money, the amount is credited to the user's BIPS account. He can then trade that money for bitcoins.
Since BIPS is using Bitstamp to trade bitcoins, exchanging bitcoins between peers happens instantly. However, this requires that the buyer and the seller have both a BIPS account. Doing a transaction outside the system, i.e., over the Bitcoin core network, takes the usual time, which means around one hour.
BIPS offers two Android applications. One is called BIPS POS [18] and is meant for merchants as a point of sales. It offers the functionality to receive bitcoins from other users. Therefore, the buyer scans the seller's address by means of scanning a QR-Code. The second application is called BIPS Market [19]. After authentication a user can have a detailed look on his account information. He can as well buy or sell bitcoins on the market.
To summarize, BIPS meets requirements (i) and (ii) described above, but neglects requirement (iii).

2.3 BitcoinPAYFLOW

BitcoinPAYFLOW [7] is similar to BIPS. It encourages online merchants to accept bitcoins as means of payment. After a customer has placed an order on the merchants website, the customer receives a Bitcoin address where he has to send the payment to. The merchant is then notified when the customer's payment arrives. This is done when the first network confirmation arrives. Furthermore, BitcoinPAYFLOW forwards the payments converted from Bitcoin to real currency to the merchant. This is only done after 6 network confirmations.
Unfortunately, the website does not explain anything in detail. An important question is

how do they prevent double spending, i.e., what do they do if a block is rejected after the first network confirmation. They do not state if they inform the buyer that the payment did not arrive, if the merchant has to cover the loss, or if they cover the loss on their own. In contrast to the two applications described above, BitcoinPAYFLOW does not meet any of the three requirements defined in the beginning of this Chapter. However, the main goal of BitcoinPAYFLOW is how merchants can trade their bitcoins back to a real currency, making that their target market.

Chapter 3

Approach, Design, and Implementation

After the illustration of related applications and systems, this Chapter focuses on the approach, design, and implementation of the Mobile Bitcoin Payment Solution. Section 3.1 describes the main idea of the MBPS, the key requirements or features, and the approach. Section 3.2 presents the overall architecture or design of the MBPS and gives an overview of the key components. The implementation details, consisting of use cases and requirements among others, are covered in Section 3.3.

3.1 Approach

As already mentioned in Chapter 1, the confirmation of transactions in the core Bitcoin network takes at this stage more than 40 minutes. In order to overcome this drawback, the MBPS runs on a centralized system which keeps the accounts of all its users. Furthermore, the MBPS is based on a prepaid approach and therefore allows instant transactions. Users have to wait that given time for a transaction to be confirmed only in two cases, i.e., transferring bitcoins to MBPS and transferring bitcoins out of MBPS. Transactions between two users within the system are performed immediately within seconds.
Another key feature of this approach is the implementation of a clearing center. For example there are two users, A and B, and A pays three times within a day a given amount to user B. To keep the number of core Bitcoin transactions low, instead of heaving three separate transactions the system could initiate a core Bitcoin transaction at the end of the day. Or user B could define a given threshold, where the MBPS initiates a core Bitcoin transaction only when B's balance exceeds that threshold. In this way, transactions could be accumulated over a longer period. Since transaction fees will play a bigger role in the future [2, 20], decreasing the number of transactions which leave the MBPS system is desirable because it helps keeping the transaction fees low.
The only manner in which users can interact with the MBPS or conduct transactions between them is by means of an Android application. Using Android as operating system of the mobile devices has been chosen because currently Android makes 81 percent of the

market share (according to the data of IDC for Q3 of 2013) [21]. Furthermore, in contrast to Apple products, the recent generation of Android mobile devices support NFC.

Summarized, these are the key points of this approach:

- By applying a centralized instance in combination with a prepaid approach, transactions between two users are performed immediately. This holds only for transactions within the system.
- In order to decrease the transaction fees, a clearing center is implemented which minimizes the number of transactions leaving the MBPS system.
- Users can interact with the MBPS or conduct transactions between them by means of an Android application. For transactions between two MBPS users NFC is used.

3.2 Design

Before going into implementation details, it is good to have an overview over the whole system first. Figure 3.1 shows the architecture of the MPBS. It outlines the key components and shows how they interact.

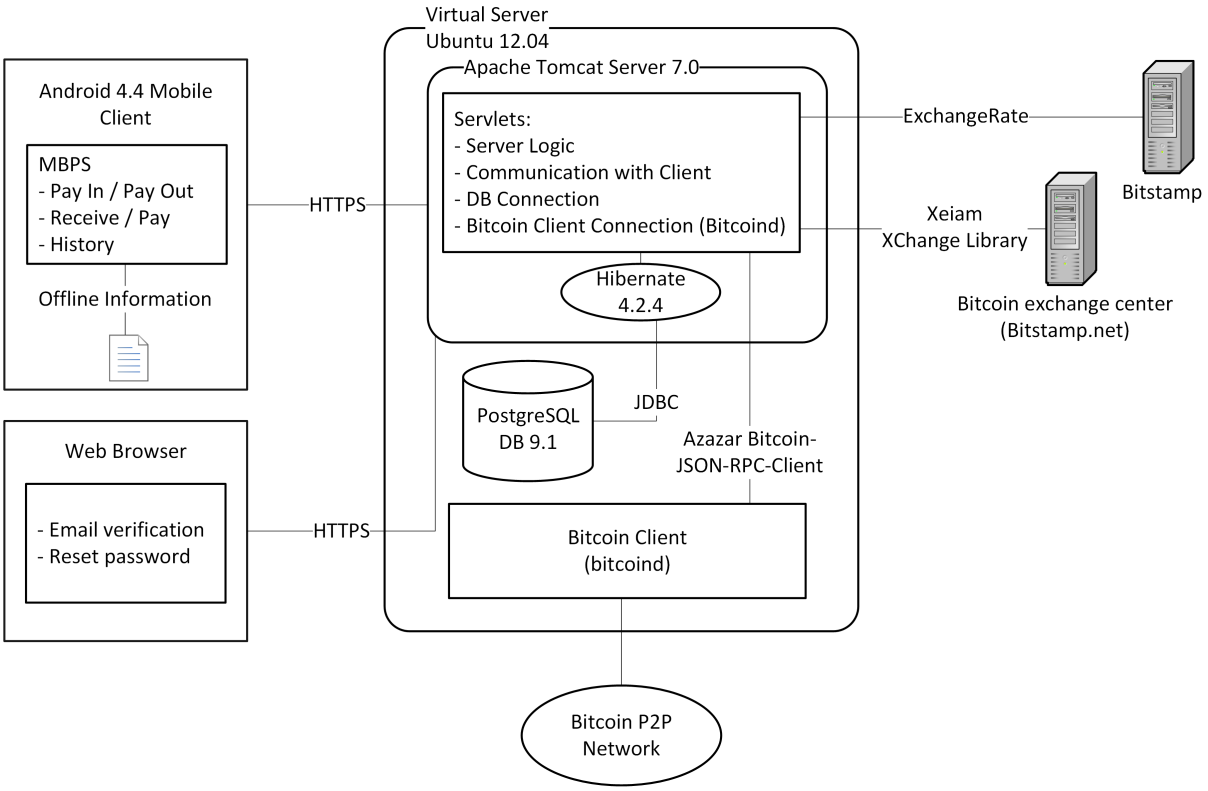


Figure 3.1: MBPS Architecture

The middle part of Figure 3.1 shows the most important component, i.e., the Virtual Server. This is the machine where the MBPS server code is running, allowing users to

register, authenticate, conduct transactions, etc. To persist this and other important information, the Virtual Server hosts as well a database service. The third service which is needed on the server side is a Bitcoin client. This is used to initiate transactions from the MBPS to any other Bitcoin address, or to listen for incoming transactions into the MBPS system. All transactions which are going through this Bitcoin client have to wait the usual 6 verifications before a transaction is confirmed.
The right hand side of Figure 3.1 shows two external servers. The upper server labelled *Bitstamp* is used to retrieve the current exchange rate USD/BTC from one of the leading Bitcoin exchanges. The lower server labelled *Bitcoin exchange center* is used to exchange bitcoins for USD or vice versa. However, this was only used for the test run (see Chapter 4).
On the left side of Figure 3.1 one can see the client which is meant to run on the user's Android mobile device. Using this client, a user can manage his user account, initiate transactions (pay with or receive bitcoins), pay out his bitcoins to another wallet, etc. For these important functions this is the only interface to the MBPS server.
For two specific cases a web browser has to be used to communicate with the MBPS server. This concerns verifying one's email address and resetting the password. Because a hyperlink containing a token as parameter is sent to the users dedicated email address, it is easier for the user to click on the link instead of copying the token to the application.

3.3 Implementation

After having seen the chosen approach in Section 3.1 and the high-level architecture in Section 3.2, this section dives now into the implementation details. First, Subsection 3.3.1 will describe the technology of the main components as seen in Figure 3.1. Subsection 3.3.2 will present an excerpt of the most important requirements of the MBPS. The functionality of the MBPS will be explained in Subsection 3.3.3 by means of use cases. Finally, the biggest and most interesting challenges faced during the implementation of the MBPS are presented in Susection 3.3.4.

3.3.1 Technology of Main Components

The component on the left side of Figure 3.1 is the mobile client. As mentioned before, the fact that most smartphone users own an Android device led to the decision to build the mobile client for the Android operating system. The Android platform version was not determined beforehand and should be chosen so that most Android users can install and use the mobile client. However, Figure 3.1 shows that the platform version 4.4 has been chosen. Even if this decision excludes the majority of Android mobile devices, it had to be taken. The reason for this is that Android versions below 4.4 do not support true two-way NFC communication – a feature which is crucial for the MBPS as shown later.
In order to communicate with the MBPS server, the Spring for Android framework [22] is included into the mobile client. It is used to conduct the HTTP requests and handle the server responses accordingly.

To prevent that the information sent from the mobile client to the server and vice versa is not eavesdropped or even manipulated, a secure connection between client and server is established on each HTTP request. This is accomplished by using the HTTPS protocol. The server's SSL certificate is signed by the certificate authority QuoVadis [23].

The machine which hosts the MBPS server is located at the CSG testbed [24]. Instead of a physical server, the MBPS server application runs on a dedicated Ubuntu 12.04 Virtual Machine. There are three applications or services running on the VM which make up the MBPS server: the Apache Tomcat server, the PostgreSQL database, and the bitcoind Bitcoin client.
The Apache Tomcat 7.0 server contains the Java application which is responsible for processing client requests, handling the interactions with the database as well as the Bitcoin client, and communicating with external servers such as Bitstamp.net for getting the BTC/USD exchange rate for example. The web application has been developed using Spring Framework (Release 3.2.4) [25]. In addition to Spring core, the Spring Security framework (Release 3.1.4) [26] is integrated which offers authentication and access-control. To facilitate the data persistence, the Hibernate framework [27] is used. The communication with the external servers to retrieve the exchange rate is performed through simple JSON requests. A special case is the communication with Bitstamp.net for the test run. In order to trade bitcoins easily, the library called Xeiam XChange [28] has been used. This is a Java library which encapsulates the JSON requests and offers a simple Java API. As mentioned above, the data which needs to be stored and retrieved or altered later is kept in a PostgreSQL (Release 9.1) [29] database. Hibernate handles the JDBC communication with the database.
The third application which runs on the VM is the Bitcoin client bitcoind [30]. It implements the Bitcoin protocol and provides access to the Bitcoin core network through remote procedure calls (RPC). The communication between the Java server application and bitcoind is facilitated by means of an external library – called Bitcoin-JSON-RPC-Client [31].

3.3.2 Requirements

Based on the work description of this Master Project (see Section 1.2) the requirements specification is part of the project. This means that the requirements were not given beforehand but had to be elicited, analysed, and documented by the team members.
The first step in the requirements specification, i.e., the elicitation, was conducted by applying different methods. First, the supervisors – which can be considered as the product owner – were interviewed in order to elicit a first set of requirements and to further define and narrow the system context. Second, brainstorming sessions in workshops where the project team participated led to even more and detailed requirements. Finally, other mobile applications and especially mobile payment applications, e.g., Bitcoin Wallet [11], were analysed.
The resulting set of requirements was then documented and analysed with the supervisors. In the analysis phase the requirements were also prioritized. Each requirement is assigned one of the three possible priorities, namely *must have*, *should have*, and *nice to have*. The *must have* priority represents essential requirements that absolutely have to be met

in order to accomplish the project successful. The *should have* priority requirements are important and improve the outcome significantly if they are met. The *nice to have* priority represents requirements which might be dropped if there is not enough time left. Furthermore, *nice to have* requirements should only be realized after the other two categories are finished completely.
The complete list of requirements can be found in Appendix A.

3.3.3 Use Cases

In parallel to the requirements specification, the use cases of the MBPS were elaborated. Each use case can be linked to one or more requirements, as shown in Appendix A. Figure 3.2 shows the use case diagram. There are two main actors which interact with the system, namely the *Buyer* and *Seller*. These actors have all use cases in common except of one – the *Buyer* pays for a service or good and the *Seller* receives the payment. Since the MBPS is a centralized system, all the use cases belonging to the *Buyer* or *Seller* also involve the third represented actor, the *Server*. There are also use cases which belong only to the *Server* and do not involve the other actors. Below each use case is explained in detail, beginning with use cases the *Buyer* and *Seller* actors have in common. The use cases belonging to the *Server* actor are explained at this end of this section.

Create Account

Since the MBPS server needs to keep track of transactions and other data, each user needs an individual user account on the server. Therefore, the MBPS offers a complete user management system which allows users to create, update and delete their account.
When creating a new account on the mobile client, the user has to provide some information. For instance, this is a unique username, a password, and a valid email address. The username must be unique since it serves as an identifier for the given user. There are some constraints for the password which assure that the password is not too short and therefore a security risk for the user. To avoid type errors when entering the password, the user has to enter it twice. For the sake of security the user passwords are not stored in plain text in the MBPS database. Instead, the password's hash (including a salt) is stored. MBPS uses the *bcrypt* hash function, which is included in the Spring Security framework [32]. This is one of the reasons why a user needs to provide his email address. If he forgets his password, he can ask to set a new one. Therefore, a link is sent to his indicated email address, which allows resetting the password. The user's email address is also used for other purposes, he can for example request some account information to be sent by email. To assure that the user has provided a valid email address and an address which belongs to him, the registration is not finished after clicking on the *sign up* button. The user receives a registration link on his provided email address where he has to click on that link and verify his email address. This assures that the provided email address belongs to the given user.
Besides the information which the user provides, there are additional data linked to each account. Every time when an account is created, the MBPS server generates a key pair (cf. public-key cryptography) and stores it in the internal database. The user needs the

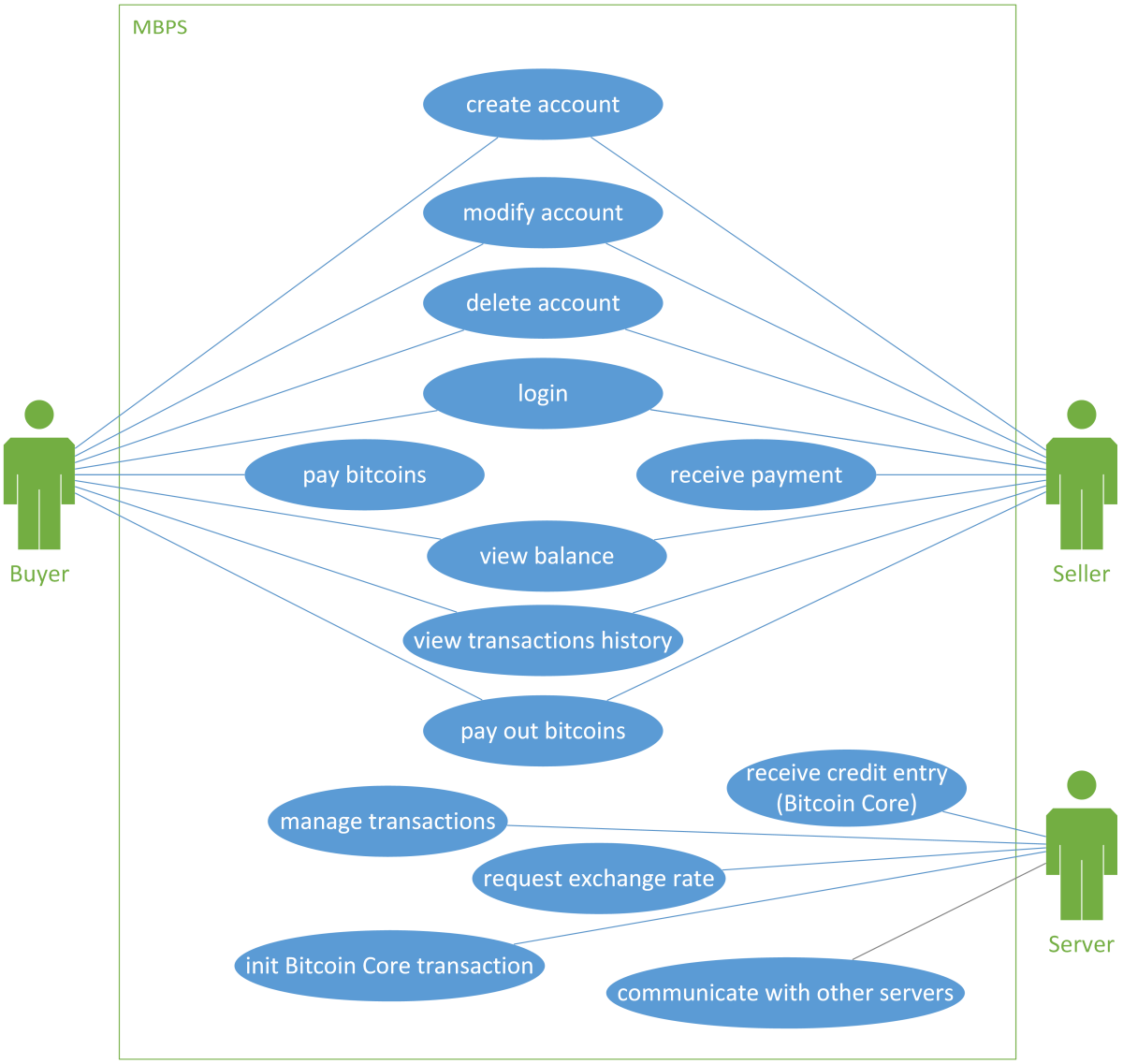


Figure 3.2: Use Cases

private key to sign transactions. The server can than verify that signature with the user's public key. Further, each user account receives on creation a new Bitcoin address. This address can be used when a user wants to pay in bitcoins into the MBPS system.
The sequence diagram B.1 in Appendix B illustrates how the client and the server interact in order to create a new user account. Figure C.1 in Appendix C shows the registration view of the mobile client user interface.

Modify Account

Once an account is created and verified, a user has the possibility to edit his account information. For instance, it is only possible to change the password or the email address. Figure C.2 shows the profile view of the user interface. Clicking on the pencil symbol allows changing the given data.

Delete Account

If a user wants to uninstall the application and quit the system, he has the possibility to delete his account. Similar to the other use cases, this can only be achieved by means of the Android mobile client. Figure C.2 shows the user interface and the button which has to be pressed in order to delete one's account. To avoid that a user deletes his account accidentally, the user is prompted to confirm the deletion of his account.
Another case which has to be covered by the MBPS is when a user wants to delete his account but still has bitcoins on his balance. To assure that these bitcoins are not locked in the system and are no more retrievable, a user can only delete his account when his balance is void. In the case that the user's balance is greater than zero, he has to pay out his bitcoins to another wallet first before being able to delete the MBPS account. Figure B.3 shows the sequence diagram for the *Delete Account* use case.
When a user successfully deletes his account, the user account is not deleted from the system's database. Instead, a *deleted* flag is set. This assures that all transactions concerning the deleted user are traceable at any time (e.g., a user can see the username of a deleted user in his transaction history). However, the deleted user can then no longer sing in or use any of the system's functionality.

Login

In order to protect the user's bitcoins, only the account owner should have access to his account. Hence, the MBPS requires user authentication in order to access its services. After the user has verified his email address and therefore finished the registration process, he can sign in. Except of the *Create Account* use case, the user has to authenticate against the server for all the other use cases. To do so, the user has to provide his username and password. Figure C.3 illustrates the sign in view of the user interface.
As mentioned in the *Create Account* use case above, all user information – including the keypair – is created on the server. However, the user needs for example his private key in order to sign transactions. This requires that the user information has to be replicated on the user's mobile device. The private key mentioned is not the Bitcoin private key. It belongs to a keypair which is used to authenticate the user against the server and for data integrity, as explained in Section 3.3.4.
Whenever a user sings in, the mobile client receives the current account information and stores it locally on the device for further purposes. The sequence diagram in Figure B.2 explains this procedure. The approach of having all the account information on the server has also the advantage that a user can use different mobile devices, e.g., once his mobile phone and some other time his tablet. It is important to mention that the account information is not stored in plain text on the mobile device. The file containing this sensitive information is encrypted with the user's password.

Whenever a password is used, it might happen that a user forgets his one. Since the MBPS does not store the password in plain text, it is not possible to send the password to the given user. However, the MBPS allows to restore a password or rather to set a new one. By providing his email address, a user receives a hyperlink where he can provide a new password on a simple web page. Figure C.4 shows the user interface where the

user can request the reset password hyperlink. This approach implies that a compromised email account might also result in a compromised MBPS account and a lost of all bitcoins. On the other side, forgetting the password does not mean that the bitcoins are lost. This is also due to the fact that the bitcoins and the complete user information are stored on the MBPS server.

Another issue with this approach is that it always requires a communication between the MBPS client and the server. However, mobile devices are not always connected to the internet. To increase the usability of the system, the mobile client offers an offline mode. This feature is similar to the Bitcoin Wallet shown in Section 2.1.
Since the account information is centralized on the server, the offline mode can only offer limited functionalities. It is for example not possible to change the account password. However, two use cases – i.e., *View Balance* and *Pay Bitcoins* – are possible. The information the user sees has the status of the last login with that specific device. If he has never used that device, than it is clear that he cannot use the application in offline mode. Having accessed the user account with another device in the mean time and undertaken an action which effects the balance clearly results in outdated information.
Given the case that a user has used the mobile client at least once before and has successfully authenticated against the server, it is possible to use the mobile client in offline mode. In this case the user has to authenticate against the mobile device. The file containing the user information, which is stored in the mobile device's internal storage, is only decrypted successfully if the correct username and password are provided. This assures that no unauthorized person can use the application.

Yet another security feature is provided by limiting the session lifetime. Being inactive for ten minutes or longer invalidates the session and automatically signs out the user. The user has than to re-authenticate if he wants to continue using the application.

View Balance

Besides of the use cases mentioned above, the MBPS allows its users to easily view their balance. If the user is online, he receives his current balance from the server. Otherwise, the last most current balance is read from the account information on the device's internal storage. In both cases a successful authentication is mandatory. Once the user has successfully signed in on the mobile client, the so called *Main Activity* is shown. In Android, this is the central view and the entry point of the application's user interface. Figure C.5 illustrates this view of the authenticated user.
Since bitcoins are subject to massive exchange rate fluctuations, it is helpful for a user to see the current balance in another currency. At this stage, the MBPS client offers the conversion from Bitcoin to Swiss Francs (CHF). The balance converted into CHF is shown below the Bitcoin balance in the *Main Activity*.

View Transactions History

What is also of interest for a user is to have an overview over all transactions he has made. A user might want to check for example if a pay in – which means a transaction

from any other Bitcoin wallet to his MBPS account – has been confirmed. Or he might want to see where his bitcoins have gone. This feature also serves for personal accounting. The transaction history is implemented on the mobile client in the *History View*, which is shown on Figure C.6.
To not overload the view, a pagination approach is applied. This means that the client receives only a batch of history items which are ordered by their timestamp beginning with the newest one. If he wants to see older transactions, he has to load the next batch or page by clicking on the appropriate button. Furthermore, a filter is also applied which divides the transactions into three categories. The first category is about transactions between two MBPS users. The second category covers pay in transactions (or deposits). Payout transactions are comprised in the the third category. The user has to chose between one of these categories in order to see the according history.
To facilitate the personal accounting, the MBPS allows exporting the complete list of transactions. Therefore, a user can request his complete history of transactions to be sent to him by email. All he has to do is to click on the according button (see Figure C.6).

Pay Out Bitcoins

A user might decide at any time to transfer his bitcoins to another Bitcoin address and therefore conduct a payout out of the MBPS system. Therefore, the MBPS offers this functionality. The only thing the user has to do is to switch to the given view, enter the amount which should be paid out, provide a valid Bitcoin address, and click on the according button. Figure C.7 shows how this described procedure looks like on the user interface.
Since this kind of transactions is processed in the Bitcoin core network, the user has to wait the usual amount of time before the transaction is confirmed and he can dispose of his bitcoins on the receiving side. However, once he has agreed to pay out an amount of bitcoins, it is charged against his MBPS balance. The sequence diagram in Figure B.4 explains this procedure in detail.
To facilitate the payout procedure, it is possible to define payout rules and conduct payouts automatically. This feature is especially of interest for users who mainly act as sellers in the MBPS and have a high volume of sales. The payout rules can be defined on a time/day or on a balance threshold basis. If a user for example wants to have at maximum a given amount of bitcoins on the MBPS system, he can define a balance threshold rule which automatically pays out his balance to a predefined Bitcoin address when his balance exceeds the defined threshold. The second option is to define temporal payout rules. They allow a user to assign up to four payout times per day. At these times the complete balance is paid out to the specified Bitcoin address. Figure C.8 and C.9 show the user interface of the balance limit and the temporal payout rule definition respectively.
The opposite of this use case is to pay in bitcoins into the MBPS. However, since a user has to initiate such a transaction outside of this system, it is not a use case. Nevertheless, the server has to listen for incoming transactions and map them to a given user. This is explained in the *Receive Credit Entry* use case below.

Pay Bitcoins

This describes the use case where a user sends bitcoins to another MBPS user. Due to the centralized server approach and the implement NFC, this payment is confirmed instantly and can be executed by holding two devices together for a short period of time. Similar to the payment procedure in a store where the cashier enters the amount and the customer pays in cash or with a card, the seller – i.e., the party which receives the bitcoins – enters the desired amount in the mobile client. In order to pay or to send bitcoins, the buyer has to be authenticated and in the respective application view. After establishing a NFC contact with the seller's device, the buyer sees on the user interface on his device the seller's username and the amount the seller wants to receive. The mobile client does also convert the Bitcoin amount to CHF and displays it. The buyer has then the possibility to check the information and to accept the payment if everything is fine or to reject it if he does not agree with the payment. Clicking on the *Reject* button terminates the payment and sends an appropriate message to the seller. Figure C.10 illustrates the *Pay Bitcoins* user interface.
When the user clicks on the *Accept* button, the payment protocol proceeds. The buyer signs therefore a message with his private key. This message contains amongst others the seller's username, the buyer's username, and the Bitcoin amount. Singing the message assures that the seller cannot alter for example the amount without the signature becoming invalid. This is particularly important because during the payment all communication goes over the seller, i.e., the buyer does not directly communicate with the server. Then, the signed message is sent to the seller via NFC. The seller forwards it to the server, together with a similar message which he creates and signs with his private key. Once these two signed messages arrive on the server, they are checked if they match. The server accepts and processes the transaction only if both messages contain the same values, the signatures are valid and the buyer has sufficient funds. If the server refuses the transaction, an appropriate error message is returned to the seller. Otherwise, an acknowledgement message is returned. In both cases, the seller forwards the message to the buyer via NFC. Independently, both clients show a message on the mobile screen indicating the transaction success or failure and the payment procedure terminates. A more detailed look on the protocol and the messages exchanged is given in the sequence diagram on Figure B.5.
To speed up the payment process, especially when the buyer and seller know and trust each other, the MBPS client offers the functionality to accept payments automatically, without waiting for the user to press a button. This is achieved with the help of a switch button, as shown in Figure C.10.
As mentioned above, during the payment process the buyer does not communicate with the server at all. The intention behind this approach was to allow users to pay even when being in offline mode.

Receive Payment

This use case is similar to the one above, but from the viewpoint of the seller. In order to conduct a transaction, the user acting as a seller must follow a different procedure. As mentioned above, the seller has to enter the amount he wants to receive. The mobile

client allows to enter the desired amount in Bitcoin or in CHF. Therefore, the user has to chose the currency from the drop-down menu as shown in Figure C.11. If the user enters the amount in CHF the user interface shows also the converted Bitcoin value and vice versa.
Once the seller has entered the amount in the chosen currency, he only has to establish a NFC connection. This is done when the seller and the buyer hold their mobile devices together. The seller starts the protocol by sending his unsername and transaction number (what this is and what it serves is explained in Section 3.3.4 below) and the amount in BTC. The buyer than builds his message using this information and signs it as described above before returning it to the seller. Once the seller receives the response, he can read out the details he needs – i.e., the buyer's username and his transaction number – in order to create such a signed message as well. Both messages are then sent from the seller to the server. That means that the server is asked to confirm a transaction as described in both messages. The server response states if the transaction has been successful or not. This information is forwarded via NFC to the buyer and shown on the display of the mobile device.
To improve the handling of the application – especially for point of sales as the Mensa in the test run (see Section 4.1) – the *Receive Payment* view offers a calculator functionality. Cashiers can then sum up given prices and do not need to execute mental arithmetic.

Manage Transactions

This and the following use cases concern the MBPS server. Based on the centralized approach chosen due to the need to allow instant transactions, the server is the instance which manages the transactions and keeps the books of the users. Therefore, the server is responsible for accepting or refusing a transaction request. The server refuses a transaction request if one of the following conditions is met:

- A signature attached to the message is not valid. (The server can verify the signatures since it has the public keys of all users.)
- The buyer's and the seller's messages are not identical apart from the signature, e.g., the seller wants to cheat and requests more bitcoins than the buyer has agreed on to pay.
- The buyer's balance is too low for this transaction.
- The amount contained in the messages is negative or equals to zero.

If none of these conditions is met, then the server writes the transaction to the database, updates the two user balances accordingly and returns the confirmation back to the seller.

Request Exchange Rate

The mobile client shows on different views the exchange rate or converts an amount from BTC to CHF and vice versa. Whenever the client needs the exchange rate, it has to

request it from the MBPS server. The server itself gets the current exchange rate from Bitstamp [9]. The response is however the USD/BTC exchange rate. This needs to be converted to CHF/BTC. Therefore, the USD/CHF exchange rate is needed. The server gets the latter from Yahoo Finance [33]. The USD/CHF exchange rate is updated every hour since the volatility is very low. The USD/BTC exchange rate however is updated every 5 seconds. The most up to date CHF/BTC exchange rate is cached on the server and returned to a client whenever requested.

Init Bitcoin Core Transaction

Whenever a user initiates a payout – manually or automatically by means of payout rules – the server has to initiate a Bitcoin core transaction. This kind of transaction is processed with the aid of bitcoind. Therefore, the Bitcoin address stored in the user account which has been set by the user is used. Even if it takes about one hour for a Bitcoin core transaction to be confirmed, the given amount is debited from the user's account instantly. This prevents that a user can spend his bitcoins again in the meantime, e.g., initiate another payout or transfer bitcoins to another MBPS user.

Receive Credit Entry

This use case is the opposite of the *Pay Out Bitcoins* use case described above. Since the MBPS works as a prepaid system it has to offer the functionality to pay in bitcoins into the system. A user has the possibility to transfer bitcoins from any other Bitcoin address to his MBPS account. To be able to map an incoming Bitcoin core transaction to a user and to credit his account, each user has a distinct Bitcoin address where he must transfer bitcoins to. The *Pay In* view on the user interface, as illustrated in Figure C.12, displays this Bitcoin address. By using this address, the user has to initiate a Bitcoin core transaction outside of the MBPS system. To avoid errors while typewriting this address, the user has the possibility to copy it to the clipboard or to send it to his email address. As is usually the case with Bitcoin core transactions, the user has to wait a given period of time before he can dispose of his bitcoins on the MBPS. To protect against double spending and revoked blocks, the MBPS only credits incoming transactions after a given number of transaction verifications. Based on requirements 11.1 and 11.2 in Table A.1 Appendix A the required number of verifications is 6 for amounts less than 0.5 BTC and 12 for amounts greater than or equals to 0.5 BTC, respectively.
In order to check if a transaction has been confirmed or not, the MBPS server has to poll the Bitcoin core network permanently. The used Bitcoin-JSON-RPC-Client library does this polling in a 30 seconds interval. Once the required number of confirmations have arrived, the server accepts the given transaction and updates the user's balance. The sequence diagram in Figure B.6 points up this procedure.

Communicate With Other Servers

The last use case from the use case diagram on Figure 3.2 deals with the communication between servers, i.e., multiple MBPS servers. Due to time constraints the communication between MBPS servers could not have been tested. For the main goal of the project and the test run only one MBPS server was required.
However, the communication between MBPS servers is possible with the approach and design chosen. Similar to the clients which communicate with the server, other servers would also communicate over HTTP requests.

3.3.4 Challenges

The design and implementation of the aforementioned system posed different challenges. This Section describes the two most important categories of challenges – i.e., security and technological challenges – including their solutions.

Security Challenges

The first category of challenges concerns the security of the system. Since security is a very important topic in a payment system, different security mechanisms and services are applied to secure the user's fortune. As mentioned above, the MBPS requires user authentication for the online and offline usage of the application and thereby offers data confidentiality as well as access control. Furthermore, the communication between client and server takes place over an HTTPS connection and is therefore encrypted. The third aspect concerns the NFC data exchange between two clients in order to pay or receive bitcoins. In contrast to technologies like Bluetooth or WiFi, the range of NFC is much smaller – i.e., about 4 cm [34, 35]. Therefore, a man-in-the-middle attack cannot be executed without the two communicating parties to detect it. This is due to the physical closeness of the two clients. Hence, there is no need to encrypt the NFC communication and to offer confidentiality. Nevertheless, data origin authentication and data integrity is needed because the server has to verify that both parties, seller and buyer, agree on the transaction details and are willing to do the business. This is achieved by signing a message with the private key and attaching the authentication header to the given message.

Another security aspect concerns the payment protocol. Even if double spending bitcoins is not possible due to the centralized approach, there are other threats. Hence, the protocol should be by design resilient against, e.g., malicious users or clients, or errors due to connection disruptions and malfunctions. Since the MBPS is open source, one could even alter the client in order to cheat on other users. This resilience is achieved through the message signatures and the introduction of a transaction number belonging to each user account. After each successful transaction, the transaction number of the corresponding user is incremented automatically on the MBPS server.
If a buyer has a malicious client he has no possibility to cheat on the seller. Changing the amount of the transaction, the recipient or any other data is recognized and refused by

the server since the two requests do not match. A malicious seller has more possibilities to cheat. Since he is forwarding the message coming from the buyer, he could alter that message. However, changing the content of the message results in an invalid signature and would be noticed by the server. He could also send the same message to the server twice, pretending that there were two transactions. This kind of replay attack does not work neither, since the second time the server receives the (same) transaction request, the buyer's transaction number is out-of-date. The same holds if the seller does not forward the server confirmation to the buyer, but pretends that the transaction was not successful hoping that the buyer will retry to pay.
Transaction numbers do not only prevent from malicious behaviour. If for example the server confirmation does not arrive to the seller, the seller and the buyer might retry the transaction. Hence, the server would reject the second payment and prevent paying twice.

Technological Challenges

The second category of challenges concerns the technology used, particularly the NFC feature in Android devices. From API level 9 (Android 2.3) on – where NFC was supported for the first time – up to API level 18 (Android 4.3) it is not possible to establish two-way NFC connections between two mobile devices. The only possibility which is offered by the API is to use Android Beam [36]. However, this feature does allow to send only one message from the sender to the receiver. Furthermore, it requires user input to start sending that one message. For any further message, one has to establish a new connection and click again on the device to send the data.
The Android Beam approach is not practical for the purpose of a mobile payment solution. Since the current payment protocol implementation is sending two messages from the seller to the buyer which also require a response (see Figure B.5), it means that four NFC connections must be established and confirmed by the user. Therefore, another approach has been elaborated first, including also Bluetooth. The following illustrates the procedure of that approach to establish a connection and to transmit all data:

1. Send the Bluetooth MAC address to the other peer by NFC (Android Beam). This requires to hold the devices together and for the sending peer to click on his mobile device.

2. The device receiving the MAC address establishes a Bluetooth connection with the other device.

3. Send all payment related data over Bluetooth, which offers a two-way communication.

4. After having exchanged all messages, close the Bluetooth connection.

However, this has also several drawbacks. First, it still requires to touch the device in order to start the payment. It would not be as easy as holding two devices together. Second, to avoid one further user input – which is required to allow pairing the devices via Bluetooth when using the encrypted channel – the insecure Bluetooth channel [37] has

been used. The insecure connection would have required to encrypt the communication between the clients, making the key exchange a challenge. Third, scenarios with selling points would also have been potential sources of attacks. A malicious user could store the seller's MAC address and flood the seller similar to a denial-of-service attack, making him unable to accept payments.

The release of the Android version 4.4 (KitKat) [38] on November 2013 smoothed the way for the current solution, which is also the fastest, most user-friendly, and most secure approach among the presented ones. Android 4.4 includes Host-based Card Emulation (HCE) [39] and by that way offers real two-way communication over NFC. There is no need to click on the device to initiate the communication – holding the devices together is all one has to do.
However, not all devices running on Android 4.4 implement the HCE. This holds for devices with a custom ROM (e.g., CyanogenMod [40]) which have the NXP NFC controller [41] built in, e.g., the Samsung Galaxy SIII [42]. Currently, for this controller the NFC implementation contains only stubs [43]. This means that even if your device has Android 4.4 and a NFC controller and therefore meets all requirements, you will not be able to pay bitcoins with the MBPS application. Though, all tested devices with a Broadcom controller [44] and Android 4.4 – e.g. Nexus 4, 5, 7, and 10 – worked reliably.

Chapter 4

Evaluation

This Chapter evaluates the approach of a mobile Bitcoin payment solution as well as the handling of the MBPS application itself. The evaluation, consisting of a test run and a survey, aims to disclose what users think about MBPS and a mobile Bitcoin payment solution in general. Furthermore, the evaluation should reveal possible drawbacks and room for improvement concerning the MBPS as well as the approach in general. Section 4.1 explains the set up and gained insights from the performed test run while Section 4.2 states the set up and results from the conducted survey.

4.1 Test Run

For testing MBPS a one week test run was scheduled together with the local Mensa Binzmühle in Zurich Oerlikon. In this test run MBPS was tested and evaluated. The following Subsections 4.1.1 and 4.2.2 describe the set up and explain the insights from this test run.

4.1.1 Set Up

From 10th to 14th of February 2014 it was possible to pay all consumptions in Mensa with bitcoins. This timeframe was selected because this was one week before the spring semester started. As the Mensa is very crowded during the semester and it is hard to try out a new system with technologies users are not familiar with in a frantic environment, it was decided to do the test run before the semester. As a result of that, less transactions had to be expected.
To make it easier for users to participate in the test run, the MBPS team provided a Bitcoin exchange point (see Figure 4.1). At this exchange and information point participants had the opportunity to get informed about bitcoins in general and MBPS and also to buy bitcoins on site to top up their MBPS accounts.

As the Bitcoin exchange rate is very unstable, an approach to overcome the exchange rate risk had to be figured out. To make sure that the Mensa received the same amount of

money as they would if they sell with cash, immediate trades on Bitstamp were done. When a user bought bitcoins at the exchange point, the exchange point also bought the same amount of bitcoins on Bitstamp. Did a user pay his meal in Mensa, a transaction was automatically executed on Bitstamp to sell the same amount of bitcoins. With this approach it could be guaranteed that only participants had to carry the exchange rate risk and not the Bitcoin exchange point and also not Mensa as a seller.
Further a fee of 10 percent was applied on top of the exchange rate to make sure no loss would result in very fast alternating exchange rate scenarios and to overcome the fees applied on Bitstamp transactions.

Figure 4.1: Bitcoin Exchange Point

4.1.2 Results

During the one week test phase, a lot of insights were gathered. This Section contains mainly results based on the MBPS team's personally made experiences and the directly received feedback by participants. Detailed user-feedback based on the performed survey is provided in Section 4.2.

General Findings And Positive Feedback

During the one week test run, 11 people took the opportunity to pay for their consumptions in Mensa with bitcoins. These 11 users did 42 transactions which resulted in sales of CHF 187.50 or BTC 0.32250311.
Participants generally liked the MBPS application and the ability for fast transmission of bitcoins without having to type in long Bitcoin addresses or to scan QR codes. Users also mentioned that they liked the simple user interface. Some potential participants of the test run without the proper hardware criticized the fact of a missing support for Apple iPhone devices or the compatibility with Android devices with OS version 4.3 and below. As the iPhone doesn't support NFC, it is not possible to support these devices so far. As only Android 4.4 allows it to make use of two way NFC communication, the restriction for Android devices had to be made.

Potential For Improvement

During the test run it became apparent, that users are not yet familiar with the handling of NFC. The range of NFC is very close, and therefore almost constant contact between the two interacting devices is necessary. In the approach of two-way NFC communication mentioned in this work, a buyer needs to wait for the server confirmation after accepting a transaction. Reality showed that a lot of users instantly took their devices away from the NFC receiver, after confirming a transaction. This resulted in a loss of the connection and the server confirmation could not be sent to the end-user. This potential for misbehaviour needs to be resolved in a future work through a better user interface with hints how long to keep the NFC contact. The test run showed, that MBPS is not yet reliable enough in these non optimal situations. Improved error handling as for example reliable retransmission and state resets need to be implemented.

4.2 Survey

For further evaluating MBPS, a survey was conducted by aid of a questionnaire. Subsections 4.2.1 and 4.2.2 explain the set up and illustrate the results from this survey.

4.2.1 Set Up

A survey was conducted in which two groups of people were questioned. Both groups consist mainly of people which have an IT background (professors, assistants, employees and students of the Department of Informatics at the University of Zurich). The questionnaire can be found in Appendix D.
One group were people, which have been invited per email to take part in the test run, but did not participate. These people were asked more general questions about what they think of a mobile payment solution of bitcoins and why they did not take part.

The second group consist of participants which actually took part in the test run and used MBPS. They were asked additional, more specific questions about their experiences with the use of MBPS.
The results are presented first in the following two chapters and afterwards shortly summarized and evaluated.

4.2.2 Results

35 people (5 female, 30 male) responded to the questionnaire whereof eight people actually used the application. 91% of these people were aware of the test run in the Mensa. The average age is 29 years (standard deviation 4.3). The people questioned are mostly familiar with new technologies and the use of smartphones as can be seen by the results of the media literacy related questions. The survey participants were asked if they read news in the internet, use internet banking, buy stuff in the internet, use social media, read news with their smartphone, and if they use social media with their smartphone. 71% answered these questions with frequently or very frequently, whereas 29% answered them with never, rarely or occasionally.
In the following paragraphs the results of the survey are briefly presented and discussed. For a better overview the questions are grouped by topic.

General Interest

Participants were asked to rate their general interest in paying with bitcoins and paying with bitcoins by mobile devices. 33% rated their general interest in both questions as very low or low. 67% indicated their interest in this topic as medium to very high. Figure 4.2 shows the detailed results.
This result shows that indeed a high interest for paying with bitcoins exists. However, this result has to be qualified based on the fact that the survey participants mainly have an IT educational background and therefore are more likely to be interested in this topic.

Interest In MBPS Information

Users could inform themselves about MBPS by reading the posters at the exchange point, surfing the homepage, speaking to the MBPS team at the exchange point or they could not inform themselves at all. Figure 4.3 shows the detailed allocation of the information channels. 71% caught up on the MBPS system whereas 29% were not interested at all and stayed uninformed. 34% used even more then one media channel to get information about MBPS.

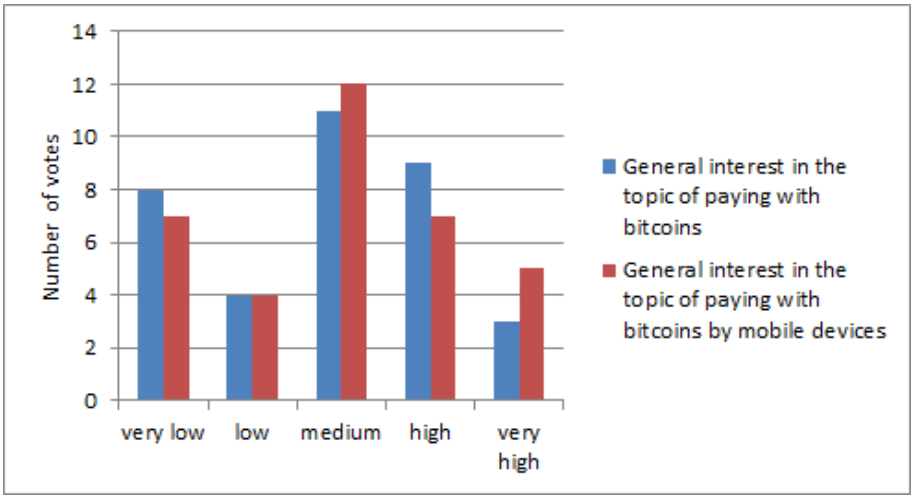


Figure 4.2: General Interest In Paying With Bitcoins

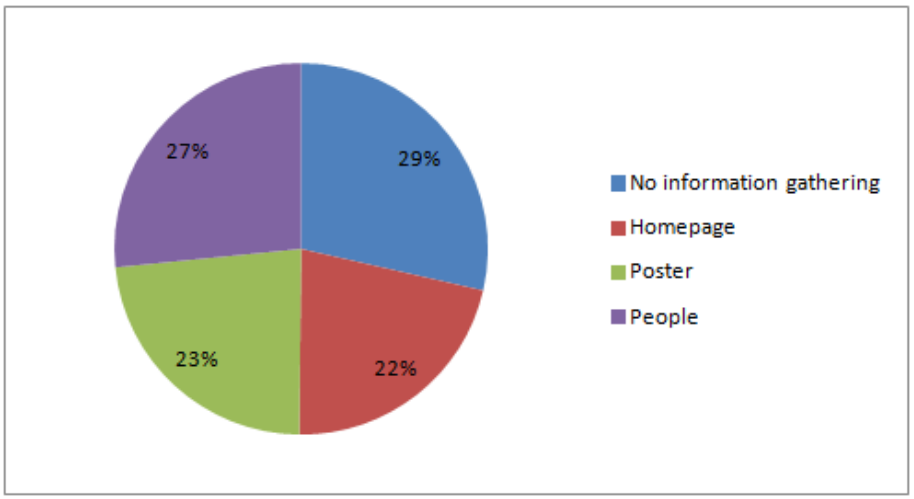


Figure 4.3: Use Of Different Information Channels

Reasons For Not Taking Part

People who filled out the questionnaire but did not participate at the test run were asked why they did not tried out MBPS. The most significant factor is here the hardware requirements as can be seen in Figure 4.4. 52% indicated that they could not take part because of the very strict hardware requirements but would eventually have been interested. The second most important factor is that survey participants were not on site where the test run took place. The reason for this is mainly because the test run took place during the semester break and a lot of students were not at the university during this time period.

MBPS Handling

Participants of the survey – who took part in the test run – had to rate the difficulty of every main functionality of the MBPS Android client. Figure 4.5 shows, that most rated the

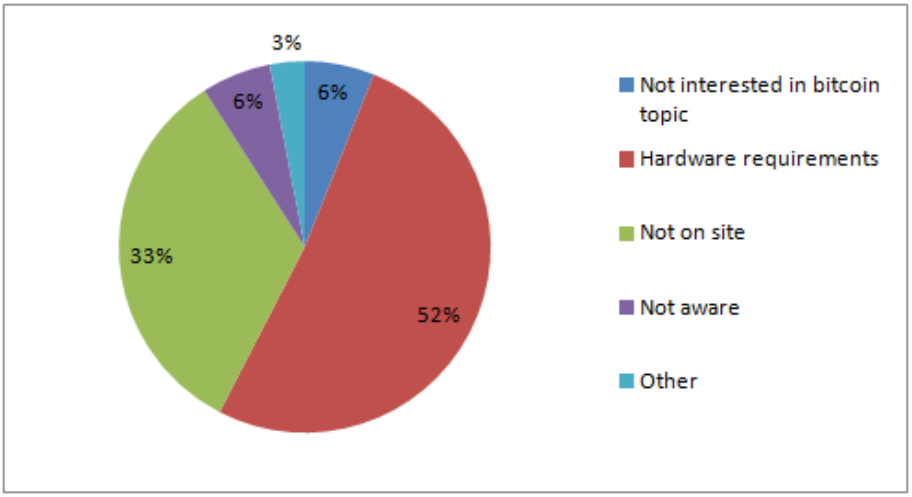


Figure 4.4: Reasons For Not Participating In The Test Run

handling of the mobile application between easy and average. 93% state that the actions in MBPS are between very easy and average, whereas 7% perceive the handling as difficult.

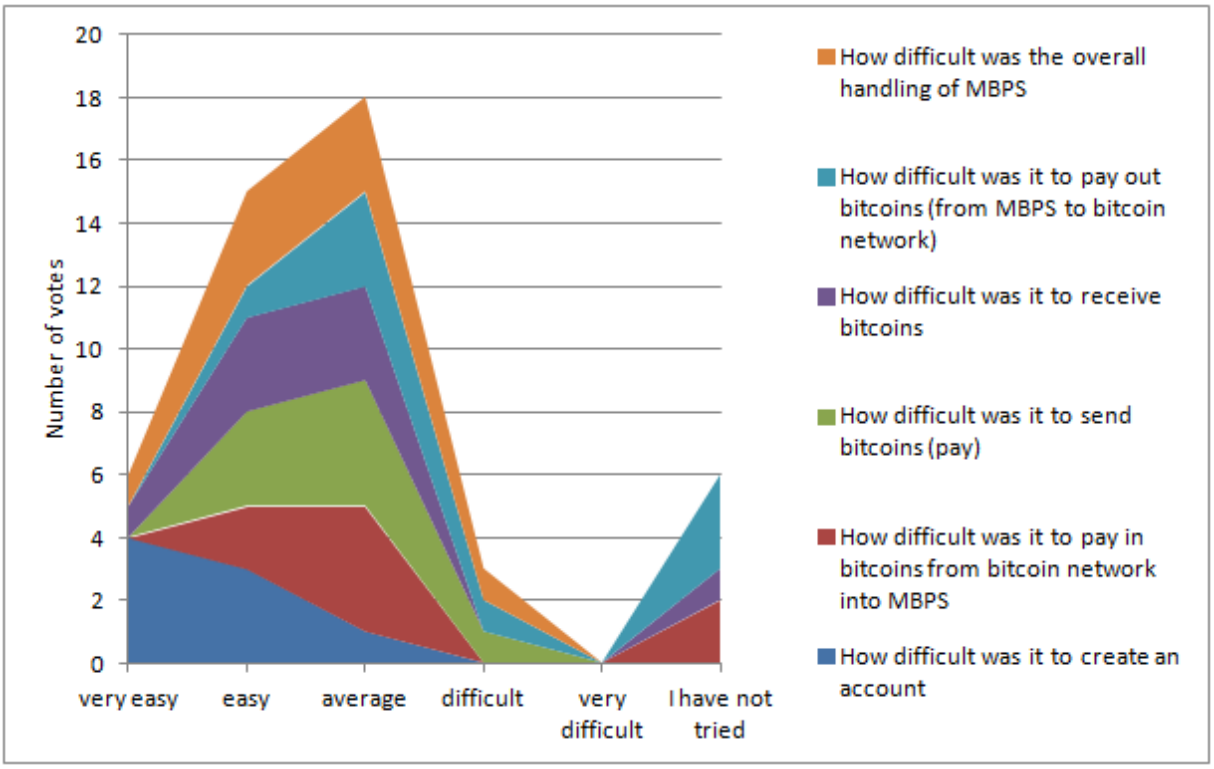


Figure 4.5: MBPS Handling

Incentives

Participants of the test run were asked about how important they perceive different incentives to use MBPS. Figure 4.6 shows that the use of Bitcoin and to be able to quickly

do a payment without cash are regarded as important incentives, whereas the usage of the newest technology is of rather average interest for users.

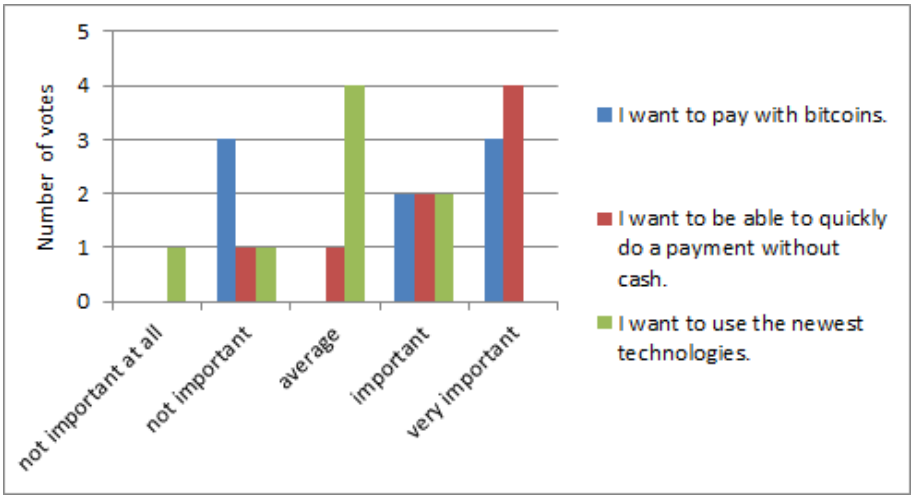


Figure 4.6: Incentives To Use MBPS

General Comments

Participants of the questionnaire had the opportunity to make further comments about MBPS, which are discussed below. One participant complained that it took much longer to pay with bitcoins than to pay with cash. The transaction itself would actually have been faster than the exchange of money. But as the cashier had to enter the amount not only in MBPS, but also in the Mensa's own cash system, it took longer. A user can also speed up the payment process if he opens the application and signs in before it is his turn to pay.
Further, it was stated that re-entering the password every time when opening the application is annoying. The session time-out of ten minutes and the necessary re-entering of the password if the MBPS application is reopened are safety features as seen in other banking applications such as the UBS mobile banking application [45] and therefore necessary even if not user-friendly.
One feedback praised the approach of a mobile payment system with bitcoins but also raised concerns regarding the required trust in the MBPS network in contrast to the Bitcoin network. This trust is truly necessary in the MBPS as not otherwise possible in such a prepaid system.
Feedback also commended the MBPS system, but criticized the restriction to Android 4.4 and NFC. This limitation is already discussed in Section 3.3.4.

Chapter 5

Lessons Learned

The aforementioned test run gave a deep insight into the matter of bitcoins, mobile payment solutions, and the used technologies. It also revealed some weaknesses of these areas and showed where the pitfalls are.

As shown in Figure 4.2, people are interested in paying with bitcoins by means of a mobile device. However, Figure 4.6 reveals that the factor to pay without cash is at least as important as to be able to pay with bitcoins. Based on the monitoring and feedback during the test run and the survey, the following points represent the lessons learned concerning the design and implementation of a mobile payment solution in consideration of the used technologies:

- If the application or system at hand shall go beyond the status of a prototype, as much as possible devices should be supported. The MBPS suffered from the hardware as well as software specifications it required. Only three users had a device which met these requirements and therefore could use their own ones to pay for their meals. All other users borrowed a device from the project team.
- When it comes to new technologies such as NFC, special instructions and guidance are required. If it is not possible that a developer or an experienced user instructs the new user, than this should happen at least by information shown on the application's user interface. As mentioned in Chapter 4, most users were inexperienced when it came to use the NFC feature to pay for a meal. They expected the transaction to be done immediately after having touched the seller's mobile device and therefore removed their device too fast. Instead, the connection should be kept alive for about two seconds in order for all protocol messages to be exchanged completely.
- When it comes to Android, it is inevitable to test the application with multiple devices, especially from different manufacturers. As shown in Section 3.3.4, there are two different NFC controllers assembled in nowadays Android smartphones. In order to work, these two controllers need to be treated differently.

Besides of these points concerning mostly the technological part of the application, there is also a lesson learned affecting the Bitcoin aspect. Right after having opened the Bitcoin

exchange point on the first day of the test run, the Bitcoin exchange rate experienced a huge drop of about 30%. Figure 5.1 illustrates that drop which reaches the bottom at 12:00. The x-axis shows the time, where the y-axis shows the exchange rate USD/mBTC. The reason for the drop was that MtGox stopped withdrawals and explained their current issue with transaction malleability.

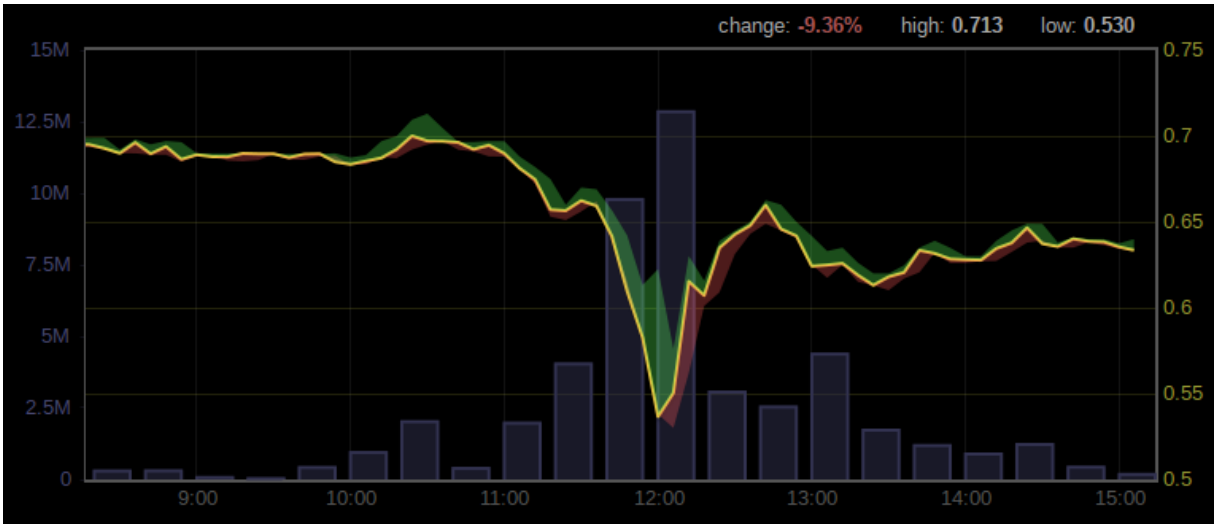


Figure 5.1: USD/mBTC Exchange Rate Drop – February 10, 2014 [46]

The test run participants who bought bitcoins at that time made a profit, since after one hour the exchange rate climbed back to almost the same value as before the press release. This is a prime example for the currently biggest drawback of bitcoins, i.e., the volatility. Even if this drop is an extreme case, Bitcoin is still volatile. Figure 5.2 shows the USD/mBTC exchange rate trend over one week, from March 3, 2014 until March 9, 2014. The high volatility makes it difficult to use bitcoins as means of payment.

Figure 5.2: USD/mBTC Exchange Rate Trend – March 3, 2014 to March 9, 2014 [46]

Chapter 6

Summary and Conclusions

A Bitcoin core transaction should only be considered as confirmed after 6 blocks verify that transaction. Based on the time it takes currently to generate one block, the payee can only be sure to have received his bitcoins after around 48 minutes. The motivation of this Master Project was to overcome this time constraint and to develop a mobile Bitcoin payment method which allows exchanging bitcoins instantly by means of a mobile device. The introduction of a clearing center which allows reducing the number of transactions and two-way Near Field Communication between the buyer and the seller represent additional goals to be achieved.
Although payment solutions such as Bitcoin Wallet, BIPS, and BitcoinPAYFLOW exist, there is no application offering the same functionality as MBPS. While MBPS is an Android mobile application which allows exchanging bitcoins instantly, uses NFC, and acts as a clearing center, the aforementioned solutions do not meet one or more of these requirements.
Based on the requirements given in the project description and the ones elaborated during the design phase, a centralized payment system has been designed, implemented, and evaluated. The system at hand is based on a prepaid approach and therefore allows instant transactions between users. Bitcoin core transactions and the involved confirmation time of about 50 minutes are only necessary when a user wants to pay bitcoins into or out of the system. In order to decrease the transaction fees, the MBPS acts as a clearing center and minimizes the number of transactions leaving the MBPS system. An important property of the MBPS is its resistance against network errors and malicious users or clients. This is required because the client is open source. It is not possible to betray other users by using a manipulated client.
The test run and the following survey showed that indeed a high interest for paying with bitcoins exists. A further incentive which supports this approach is to be able to pay cashless by means of a mobile device. However, the users were not comfortable with everything concerning the MBPS. One reported issue is the trust in the central authority. Unfortunately, this can not be bypassed in a prepaid approach.

The biggest hurdle which prevented users from participating in the test run were the strict hardware requirements. However, requiring at least version 4.4 was mandatory. The reason for this is that for Android versions prior to 4.4, the API does not offer two-way NFC, even if the built-in hardware would allow it. Furthermore, even three months

after the release of Android 4.4, only few devices are equipped with the newest Android version. Except of the Nexus series – i.e., Nexus 4, 5, 7, and 10 – only the Samsung Galaxy Note 3 is known to have been updated to Android 4.4. The problem here are the customizations by the manufacturers, which take to long to adopt to the newest version.

Future work could focus on different areas of the MBPS. First, the user experience could be improved. To overcome the users inexperience with NFC for example, the mobile client should show a progress dialog, bar, or circle indicating the progress of the transaction. This would help the users to know when they can conclude the contact. Furthermore, another use case could be supported. Instead of requesting bitcoins, users could decide and enter themselves how much they want to pay. This would for example correspond to the use case of a donation or selling bitcoins for cash. The support of other currencies (e.g., enter the amount in Euro or USD) or crypto-currencies (e.g., Litecoins) could also be implemented.
Second, the NFC protocol could be improved in order to be more resilient, e.g., retransmit packets which get lost due to a connection failure. Furthermore, the recognition of sessions could be improved. A user should for example be able to abort the contact with the seller in order to press the *Accept* button on the user interface. After re-establishing a NFC contact, the payment should proceed and terminate successfully. One could also focus on making the transactions between MBPS users faster. Therefore, the NFC protocol should be improved by different means. Instead of using Java serialization, the objects which are send via NFC could be serialized with a custom and more efficient protocol.
Third, a distributed MBPS could be developed. Different instances and organizations could operate a MBPS system, and the servers connect to each other, forming a super-peer network. The user A of the MBPS organization X could then go to a supermarket B which is a user of the organization Y. When A pays at the supermarket, the payment goes from X's server to Y's server, instantly. The outstanding amount is then sent via the Bitcoin system once or twice a day between the two organizations.

Bibliography

[1] "Bitcoin." http://bitcoin.org/en. Accessed: February 2014.

[2] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System." https://bitcoin.org/bitcoin.pdf. Accessed: February 2014.

[3] "Blockchain." http://blockchain.info/en/stats. Accessed: February 2014.

[4] "Bitcoin Wiki, Confirmation." https://en.bitcoin.it/wiki/Confirmation. Accessed: February 2014.

[5] T. Bamert, C. Decker, L. Elsen, R. Wattenhofer, and S. Welten, "Have a Snack, Pay with Bitcoins," *13th IEEE International Conference on Peer-to-Peer Computing (P2P), Trento, Italy.*, September 2013.

[6] "BIPS." http://bips.me. Accessed: February 2014.

[7] "Bitcoin Payflow." http://www.bitcoinpayflow.com. Accessed: February 2014.

[8] "NFC." http://www.nfc-forum.org. Accessed: March 2014.

[9] "Bitstamp." https://www.bitstamp.net/. Accessed: February 2014.

[10] "Mensa UZH Binzmühle." http://www.mensa.uzh.ch/standorte/mensa-uzh-binzmuehle.html. Accessed: February 2014.

[11] A. Schildbach, "Bitcoin Wallet." https://play.google.com/store/apps/details?id=de.schildbach.wallet. Accessed: February 2014.

[12] A. Schildbach, "Bitcoin Wallet." https://code.google.com/p/bitcoin-wallet. Accessed: February 2014.

[13] A. Schildbach, "Bitcoin Wallet." https://github.com/schildbach/bitcoin-wallet/tree/master/wallet. Accessed: February 2014.

[14] "Bitcoin-Qt." https://bitcoin.org/en/download. Accessed: February 2014.

[15] "bitcoinj." https://code.google.com/p/bitcoinj. Accessed: February 2014.

[16] "Blockchain." http://blockchain.info/charts/blocks-size. Accessed: February 2014.

[17] "Coindesk. Bitcoin payment processor BIPS abandons Mt. Gox for Bitstamp." http://www.coindesk.com/bitcoin-payment-processor-bips-abandons-mt-gox-for-bitstamp. Accessed: February 2014.

[18] "BIPS POS." https://play.google.com/store/apps/details?id=com.bips.pos. Accessed: February 2014.

[19] "BIPS Market." https://play.google.com/store/apps/details?id=com.bips.market. Accessed: February 2014.

[20] "Bitcoin Wiki, Transaction Fees." https://en.bitcoin.it/wiki/Transaction_fees. Accessed: February 2014.

[21] "IDC. Android Pushes Past 80% Market Share." http://www.idc.com/getdoc.jsp?containerId=prUS24442013. Accessed: February 2014.

[22] "Spring for Android." http://projects.spring.io/spring-android. Accessed: February 2014.

[23] "QuoVadis." http://www.quovadisglobal.ch/de.aspx. Accessed: February 2014.

[24] "CSG Testbed." http://www.csg.uzh.ch/services/testbed.html. Accessed: February 2014.

[25] "Spring Framework." http://projects.spring.io/spring-framework. Accessed: February 2014.

[26] "Spring Security." http://projects.spring.io/spring-security. Accessed: February 2014.

[27] "Hibernate." http://hibernate.org. Accessed: February 2014.

[28] "Xeiam XChange." http://xeiam.com/xchange. Accessed: February 2014.

[29] "PostgreSQL." http://www.postgresql.org. Accessed: February 2014.

[30] "bitcoind." https://en.bitcoin.it/wiki/Bitcoind. Accessed: February 2014.

[31] "Bitcoin-JSON-RPC-Client." https://bitbucket.org/azazar/bitcoin-json-rpc-client/wiki/Home. Accessed: February 2014.

[32] "Spring Security, BCryptPasswordEncoder." http://docs.spring.io/autorepo/docs/spring-security/3.1.x/apidocs/org/springframework/security/crypto/bcrypt/BCryptPasswordEncoder.html. Accessed: February 2014.

[33] "Yahoo Finance." http://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20yahoo.finance.xchange%20where%20pair%20in%20%28%22USDCHF%22%29&format=json&diagnostics=true&env=store%3A%2F%2Fdatatables.org%2Falltableswithkeys&. Accessed: February 2014.

[34] "NFC Forum." http://nfc-forum.org/what-is-nfc/about-the-technology/. Accessed: February 2014.

[35] "Android Developers, Near Field Communication." http://developer.android.com/guide/topics/connectivity/nfc/index.html. Accessed: February 2014.

[36] "Android Developers, Android Beam." http://developer.android.com/guide/topics/connectivity/nfc/nfc.html#p2p. Accessed: February 2014.

[37] "Android Developers, Insecure Bluetooth Connection." http://developer.android.com/reference/android/bluetooth/BluetoothDevice.html#createInsecureRfcommSocketToServiceRecord(java.util.UUID). Accessed: February 2014.

[38] "Android KitKat." http://www.android.com/versions/kit-kat-4-4/). Accessed: February 2014.

[39] "Android Developers, Host-based Card Emulation." http://developer.android.com/guide/topics/connectivity/nfc/hce.html). Accessed: February 2014.

[40] "CyanogenMod." http://www.cyanogenmod.org. Accessed: February 2014.

[41] "NXP's NFC Solution." http://www.nxp.com/campaigns/nfc/nxp-solution. Accessed: February 2014.

[42] "NXP Powers NFC in the Samsung GALAXY S III." http://www.nxp.com/news/press-releases/2012/05/nxp-powers-nfc-in-the-samsung-galaxy-s-iii.html). Accessed: February 2014.

[43] "Source Code NXP NFC Implementation." https://android.googlesource.com/platform/packages/apps/Nfc/+/master/nxp/src/com/android/nfc/dhimpl/NativeNfcManager.java. Accessed: February 2014.

[44] "Broadcom NFC." http://www.broadcom.com/products/NFC. Accessed: February 2014.

[45] "UBS Mobile Banking." https://play.google.com/store/apps/details?id=com.ubs.swidKXJ.android. Accessed: March 2014.

[46] "Bitcoinity." http://bitcoinity.org/markets. Accessed: February 2014.

List of Figures

List of Tables

Appendix A

Requirements

The table below shows the set of requirements for this project. The requirements with the priorities *must have* and *should have* have all been met. This holds as well for the *nice to have* requirements, but with two exceptions. Requirement number 16.1 could not be implemented due to time constraints. Requirement 13 has partially been met. While it is possible for servers to communicate with each other through HTTP requests, the logic which is required for the communication between payment servers is not implemented.

Number	Category	Use Case	Name	Description	Priority
1	Functional	Create account	Register Account	Allow user to register via the mobile client by providing a unique username, unique email address and secure password (more than 8 letters, special characters included).	Must have
2	Functional	Login	Authenticate	Allow user to login by providing username and password.	Must have
2.1	Functional	Login	Password Loss	User can order new password by email	Must have
3	Functional	Modify account	Edit	The user should be able to change his email address and password.	Should have
4	Non Functional	All	Login Security	Provide security mechanisms to secure connection by SSL.	Must have

Number	Category	Use Case	Name	Description	Priority
5	Functional	Delete Account	Delete Account	A user can delete his account via the mobile client. The account is not really deleted in the database, but rather set a 'delete' flag.	Must have
5.1	Functional	Delete Account	Delete Account	If the balance is greater than 0, the deletion process has to abort. The user must have the possibility to pay out his money to a Bitcoin address.	Must have
5.2	Functional	Delete Account	Delete Account	If the balance is 0, the user is prompted if he wants to delete his account. There is no possibility to revert this action. The history of transactions is kept in the DB.	Must have
6	Functional	Receive Payment	Charge	A user can initialize charging for good and enter the amount of BTC. Seller's client shows value in BTC, connects to Buyer's client and does transaction in BTC.	Must have
6.1	Functional	Receive Payment	Charge	The client should show the amount in BTC as well as CHF.	Should have
6.2	Functional	Receive Payment	Charge	The user should be able to switch between BTC and CHF for the entry.	Nice to have
7	Functional	Pay	Pay for Good / Services	User can receive charging request by NFC and approve/reject a payment. In the request view he sees his current balance and the price of the charging request in BTC.	Must have
7.1	Functional	Pay	Pay for Good / Services	A user should see requested charging amount also in CHF.	Nice to have
8	Functional	View Account Balance	View Account Balance	Allow user to see his current balance in BTC.	Must have
8.1	Non Functional	View Account Balance	View Account Balance	Allow user to see his current balance in CHF as well.	Nice to have

Number	Category	Use Case	Name	Description	Priority
9	Functional	View Account History	View Account History	A user can view the entire history of his own transactions.	Must have
9.1	Functional	View Account History	View Account History	A user can navigate through the history by loading a batch of transactions called pages - the entire history is not loaded at once. The user should have the possibility to load the previous or next page.	Nice to have
9.2	Functional	View Account History	View Account History	A user can request his entire history of transactions to be send to his registered email address in the csv (comma separated value) format.	Nice to have
10	Non functional	Pay / Receive Payment	Transaction Speed	The transaction between buyer/seller has to be finished in maximum 10 seconds after both agreeing on the transaction details.	Must have
10.1	Functional	Pay / Receive Payment	Transaction Feedback	Display clear notifications about successful/unsuccessful transactions on the mobile client (seller and buyer).	Should have
11	Functional	Receive Credit Entry	Receive Credit Entry	The server has to offer the possibility to the registered users to transfer BTC from the Bitcoin network to the server.	Must have
11.1	Non Functional	Receive Credit Entry	Credit Entry Verification	A transfer as mentioned in 11 is accepted by the MBPS server after at least 6 verifications in the Bitcoin network for payments below 0.5 BTC.	Must have
11.2	Non Functional	Receive Credit Entry	Credit Entry Verification	A transfer as mentioned in 11 is accepted by the MBPS server after at least 12 verifications in the Bitcoin network for payments equals or above 0.5 BTC.	Must have
12	Functional	Pay Out	Pay Out Seller	Allow user to initialize pay out.	Must have
12.1	Functional	Pay Out	Pay Out Seller	Allow user to define payouts by reaching a certain balance or by defining up to 4 daily pay out times.	Should have

Number	Category	Use Case	Name	Description	Priority
*13**	*Functional*	*Communicate with other Servers*	*API for other Bitcoin Payment Servers*	*Other Bitcoin payment servers must have a common interface in order to communicate with each other.*	*Nice to have*
14	Functional	Manage Transactions	Handling Transactions	Server is responsible for handling transactions and checks if transactions can be executed.	Must have
14.1	Non Functional	Manage Transactions	Prevent Double Payment	Make sure transactions will be accounted only once even if connection problems occur.	Must have
14.2	Non Functional	Manage Transactions	Prevent Double Spending	The MBPS has to assure that double spending is not possible.	Must have
14.3	Non Functional	Manage Transactions	Prevent Negative Balance	Make sure no transactions will be processed which result in negative balance for the buyer.	Must have
15	Non Functional	All	Internet Connection	Seller needs to have internet connection.	Must have
16	Functional	Pay	Apply Security Fee	To compensate for exchange rate drops (BTC to CHF), a seller may apply a fee to the real amount in percentage, called security fee.	Should have
*16.1**	*Functional*	*Pay*	*Security Fee Payback*	*Large sellers, or sellers who have defined daily batch payouts (see. Requirement 12.1), payback the applied fee, if the exchange rate has not changed between the time the good/service was payed and the payout is initiated. The buyer has to see the payback in an appropriate way.*	*Nice to have*

Table A.1: Requirements

Appendix B

Sequence Diagrams

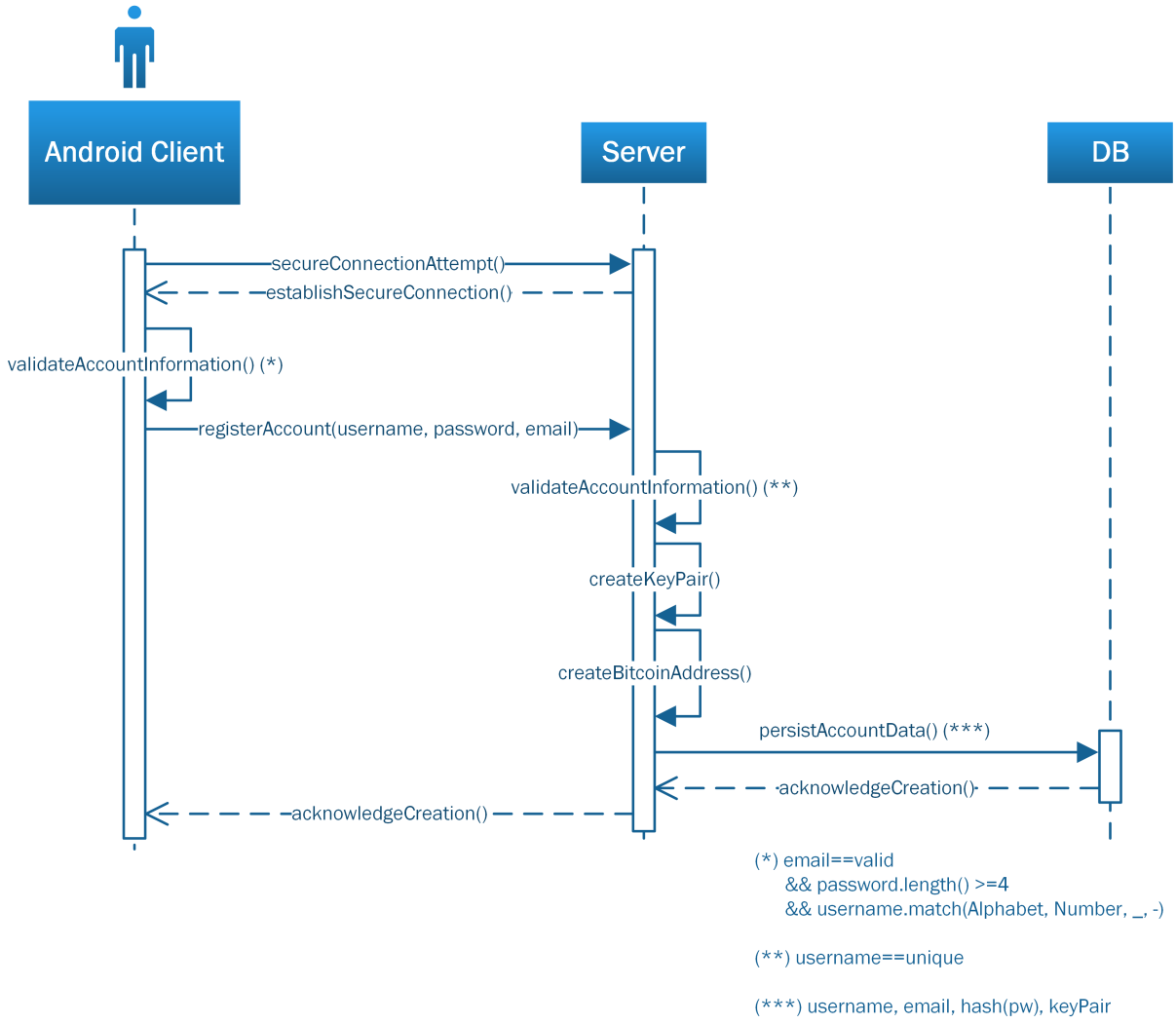


Figure B.1: Sequence Diagram - Create Account

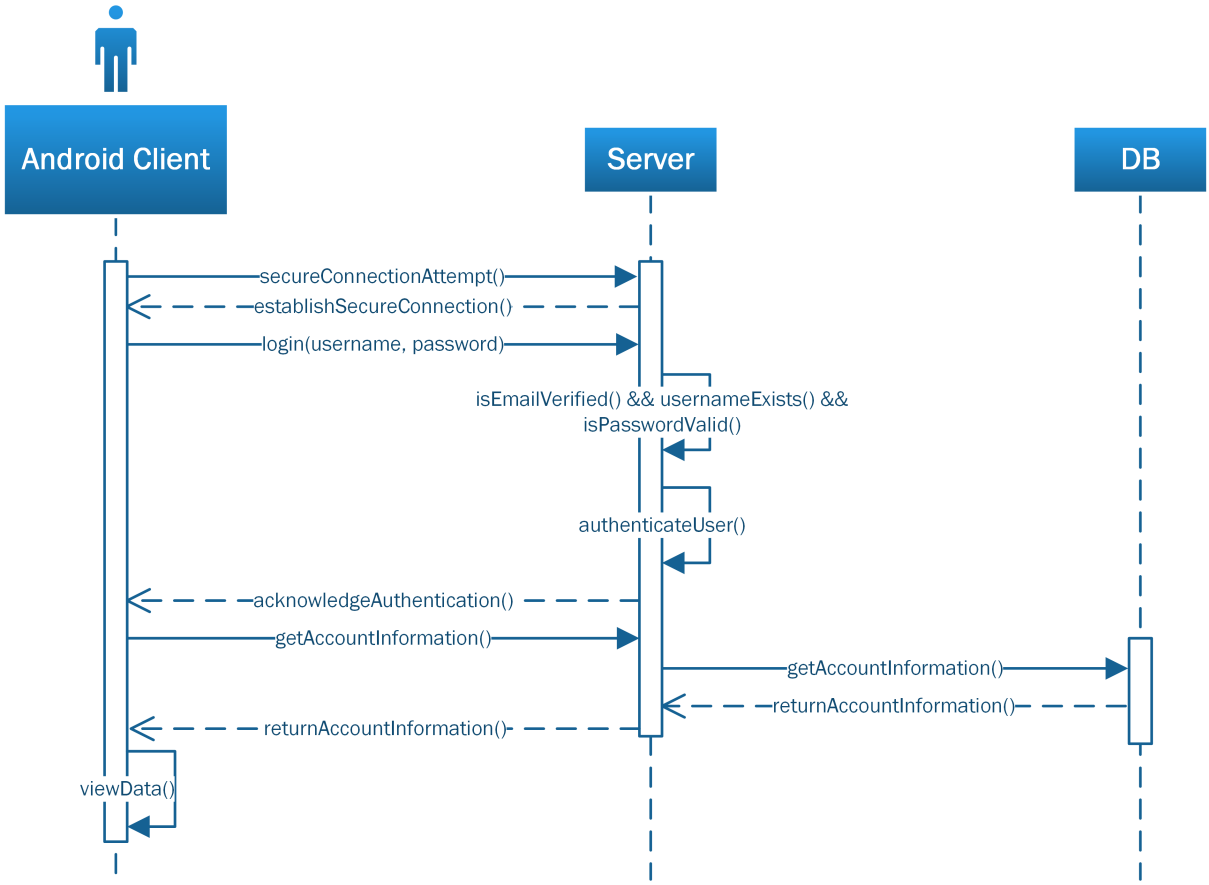


Figure B.2: Sequence Diagram - Sign In

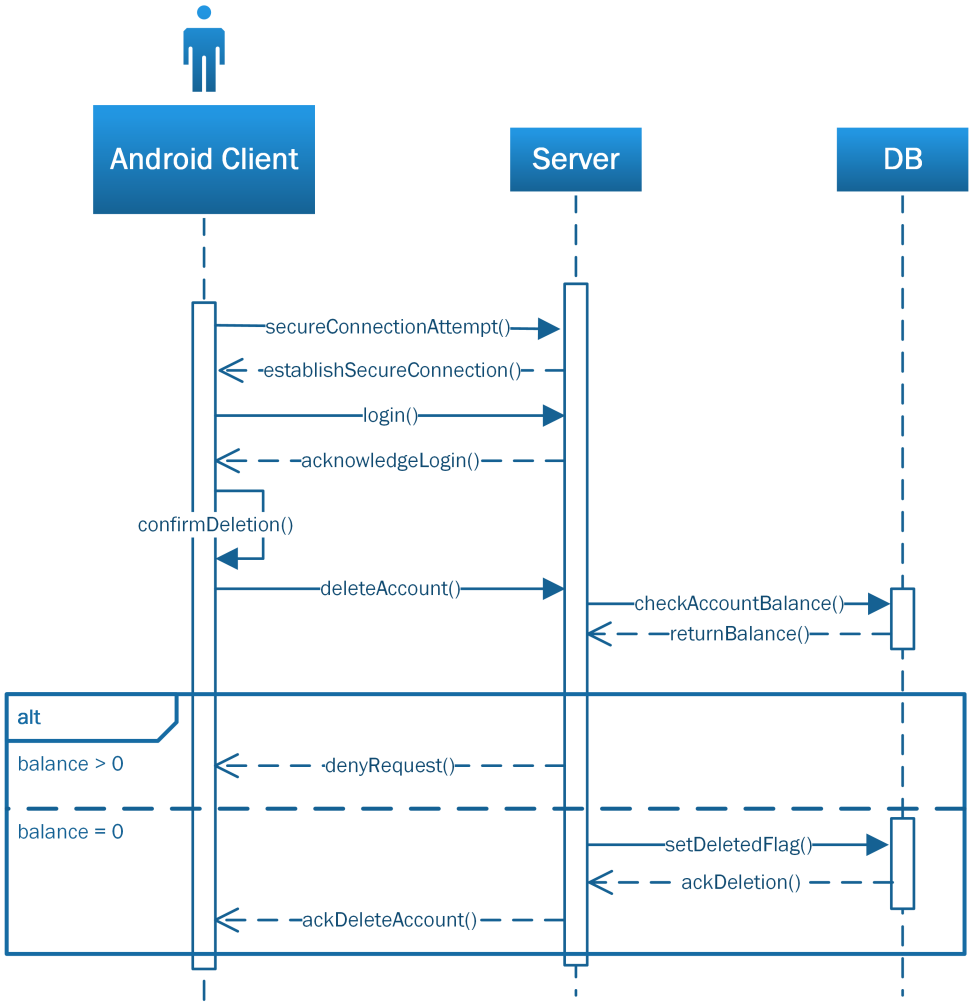


Figure B.3: Sequence Diagram - Delete Account

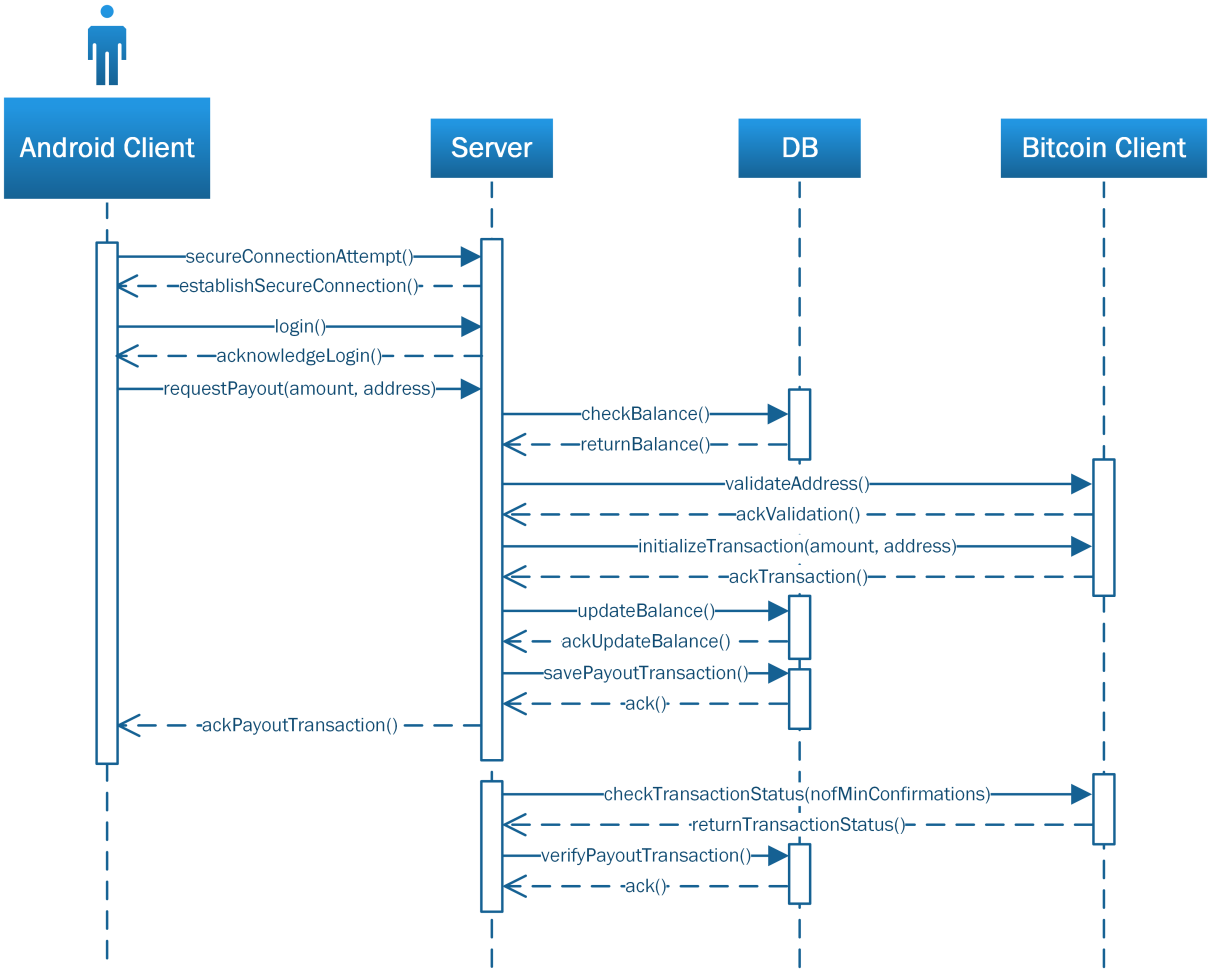


Figure B.4: Sequence Diagram - Payout

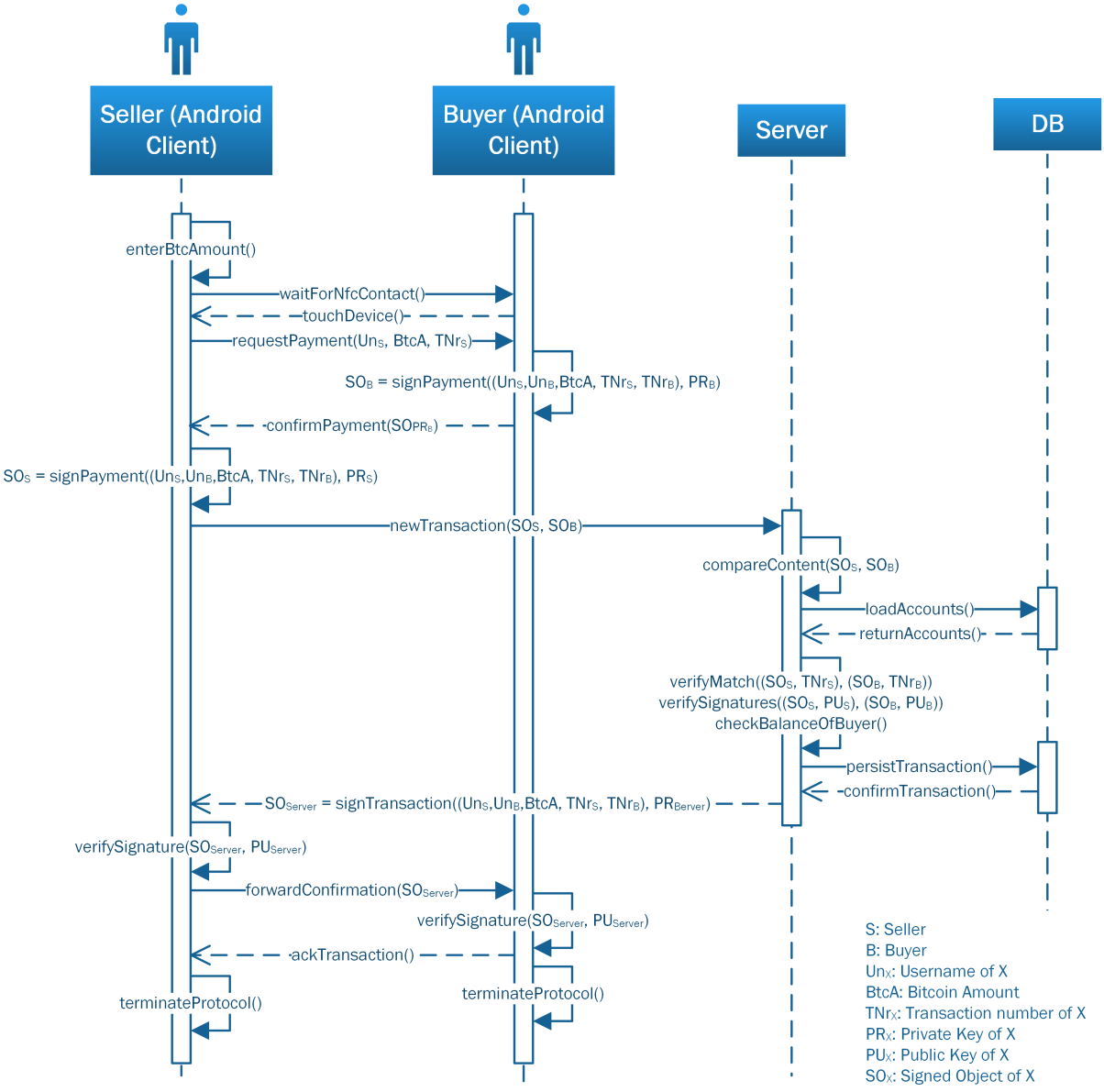


Figure B.5: Sequence Diagram - P2P Payment

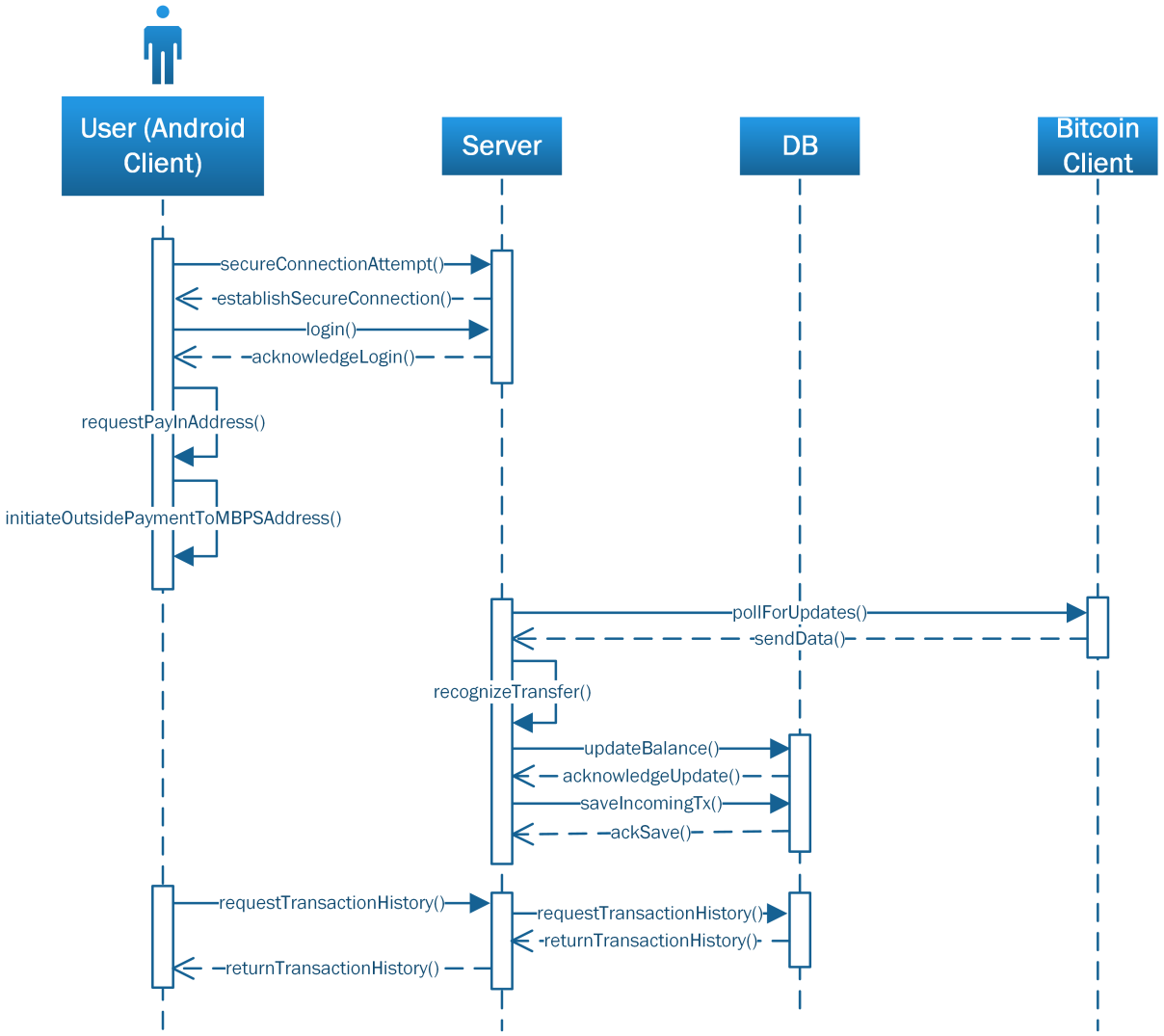


Figure B.6: Sequence Diagram - Receive Credit Entry

Appendix C

Mobile Client Screenshots

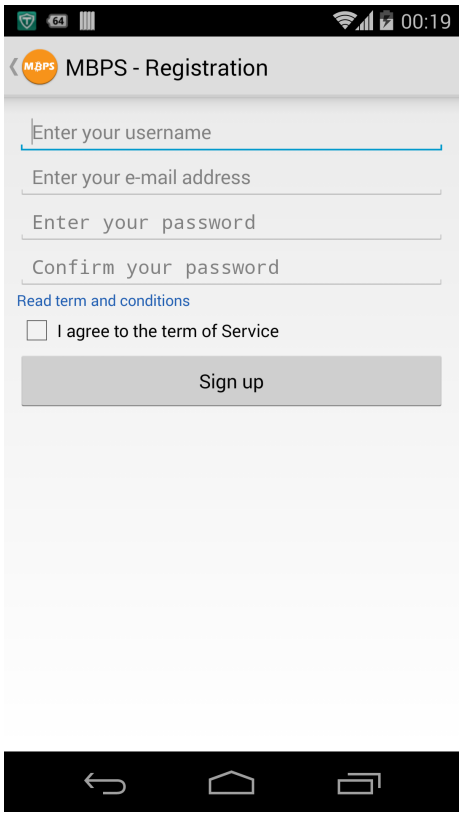


Figure C.1: Registration

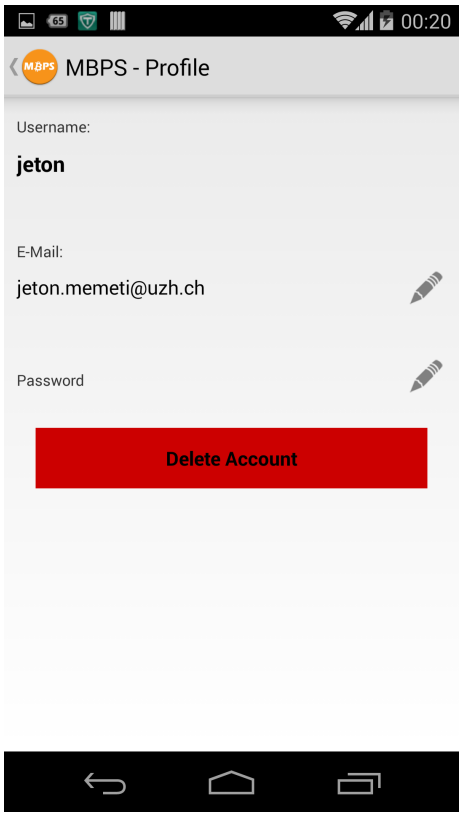


Figure C.2: Show, Edit, Delete Useraccount

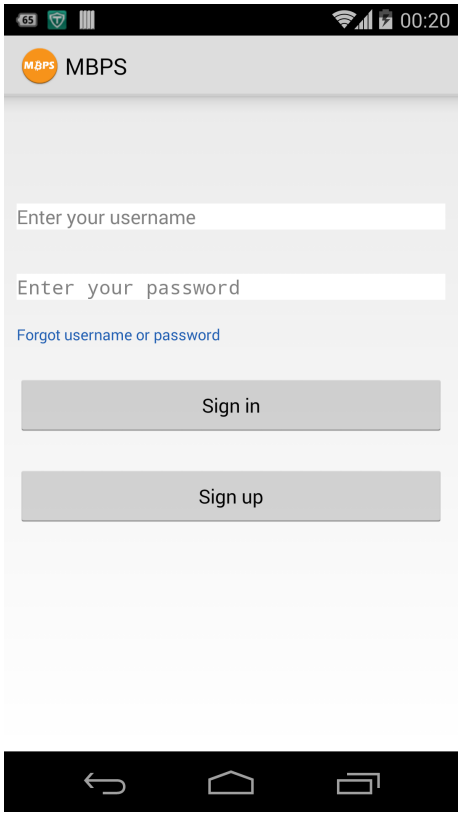


Figure C.3: Sign In

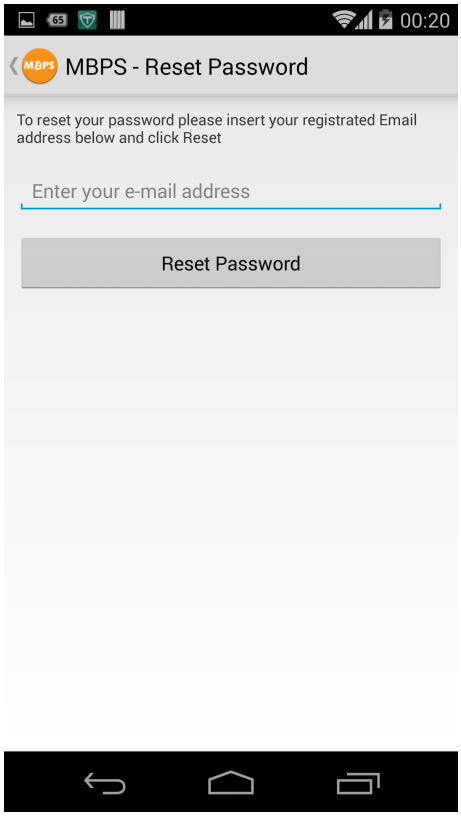


Figure C.4: Reset Password

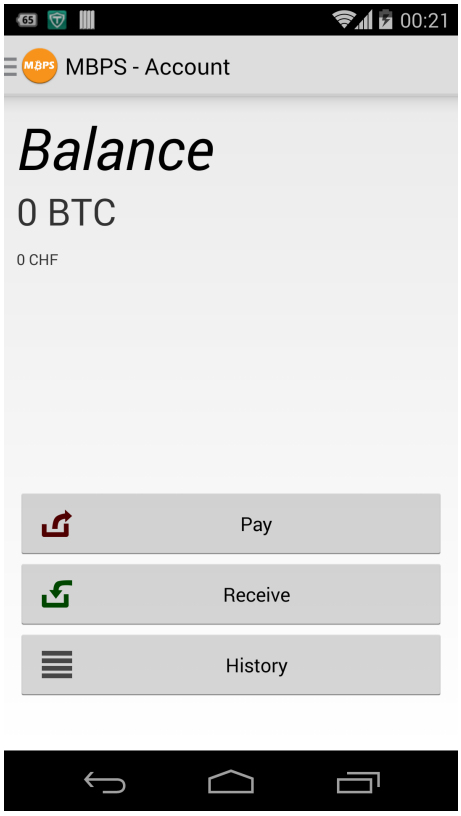


Figure C.5: Main Activity

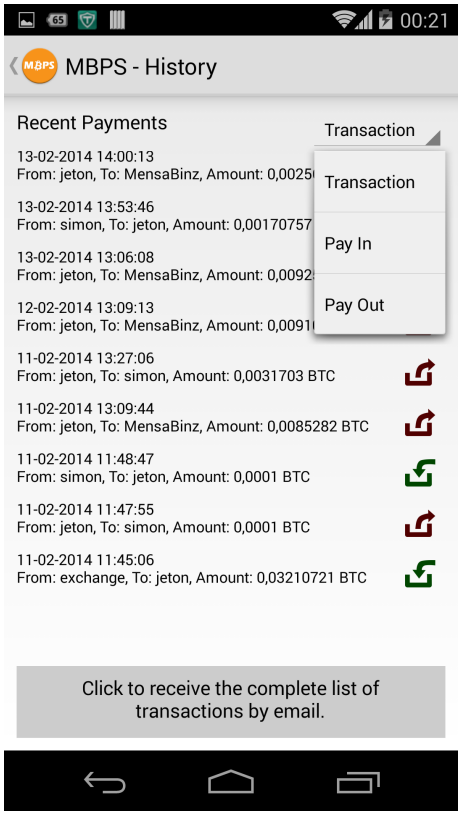


Figure C.6: History View

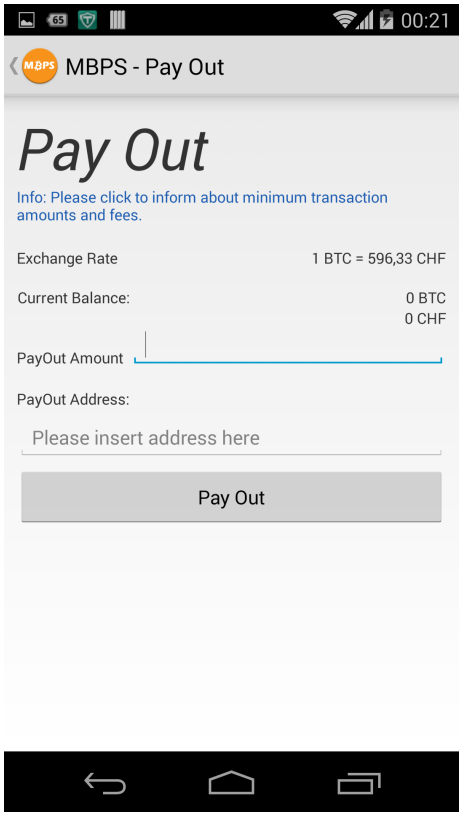


Figure C.7: Payout

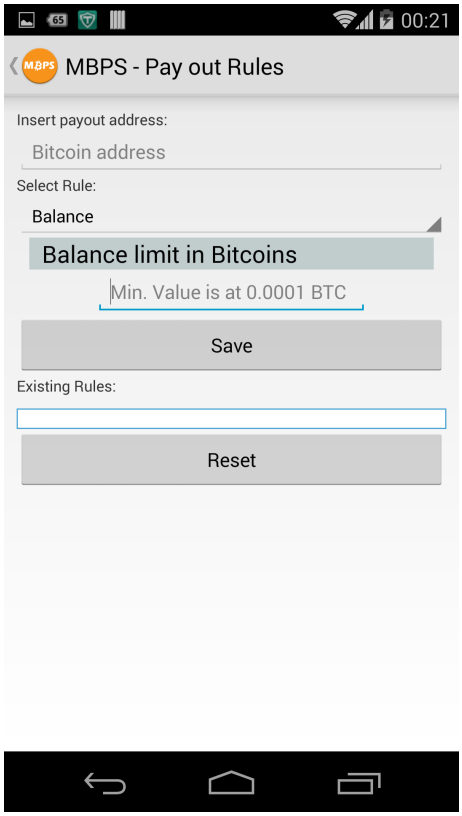


Figure C.8: Payout Rule - Balance

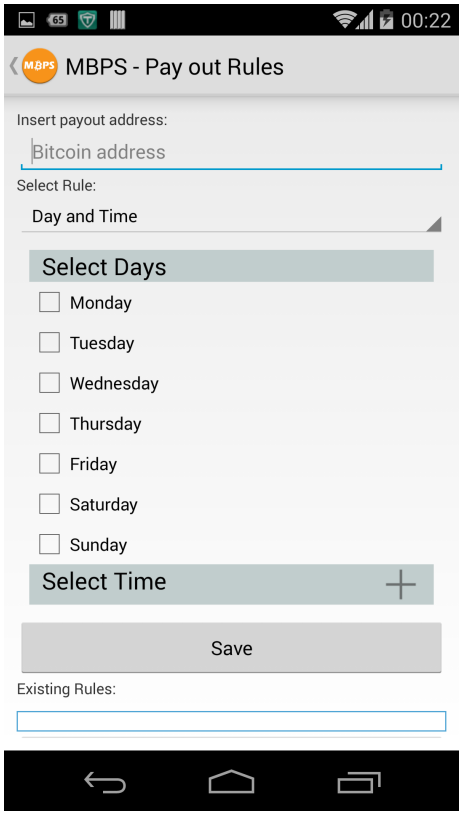


Figure C.9: Payout Rule - Time

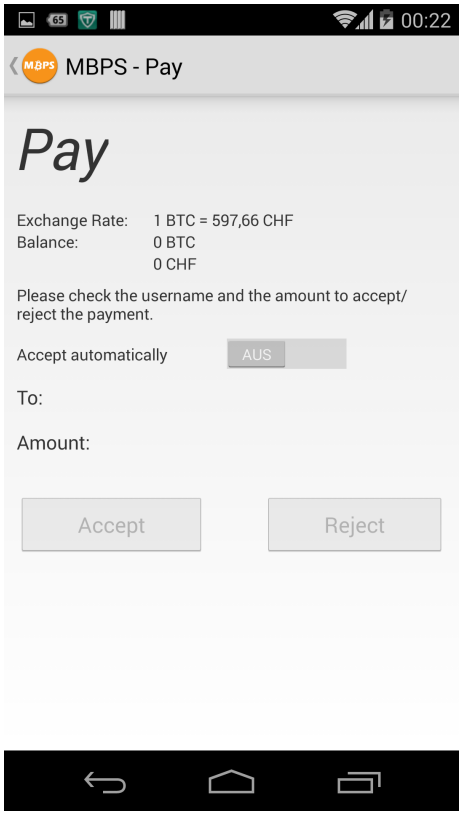


Figure C.10: Pay Bitcoins

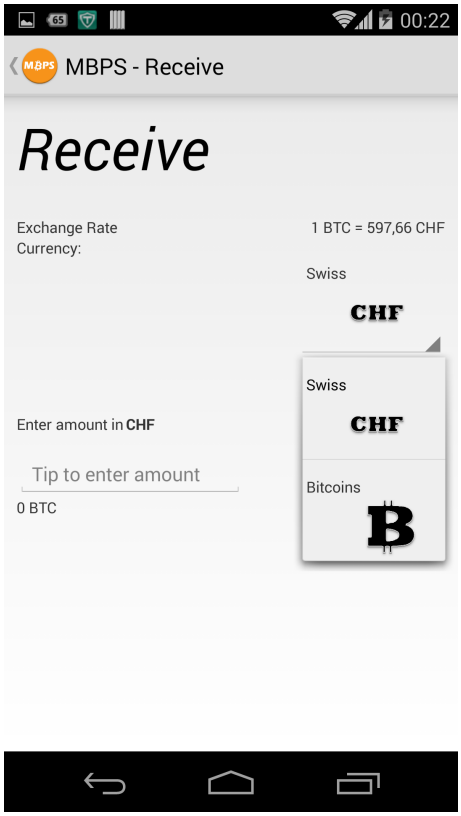


Figure C.11: Receive Payment

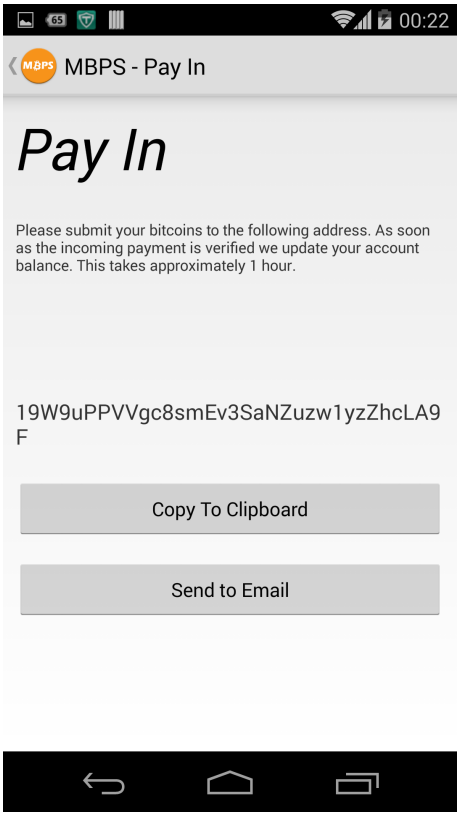


Figure C.12: Pay In

Appendix D

Questionnaire

This are the questions used in the questionnaire as mentioned in Chapter 4.

Bitcoin Survey

This questionnaire helps us gather data for our Master Project called Mobile Bitcoin Payment Solution (MBPS). MBPS is a mobile application for android devices which allows users to do bitcoin transactions immediately. We would be very thankful if you can find 4 minutes to provide answers to our following questions. Your answers will help us to improve our system. The data is collected and processed anonymously. Privacy is of outmost importance for us. Thus, we guarantee you the following: No personal information will be disclosed to any third-part entities by any means. For scientific publications the ethics and code of conducts for studies will be followed, and only fully anonymized and aggregated results will be used.

1. What is your gender? *

Female

2. What is your age? *

3. I read news in the internet *

- very frequently
- frequently
- occasionally
- rarely
- never

4. Literacy with digital media: I use internet banking *

- very frequently
- frequently
- occasionally
- rarely
- never

5. I buy stuff in the internet *

- very frequently
- frequently
- occasionally
- rarely
- never

6. I use social media (facebook, twitter, etc.) *

- very frequently
- frequently
- occasionally
- rarely
- never

7. I read news via my smart phone *

- very frequently
- frequently
- occasionally
- rarely
- never

8. I use social media via my smart phone *

- very frequently
- frequently
- occasionally
- rarely
- never

9. Interest in the idea: How would you generally rate your interest in the topic of paying with bitcoins. *

- ○ very low
- ○ low
- ○ medium
- ○ high
- ○ very high

10. Interest in the idea: How would you generally rate your interest in the topic of paying with bitcoins and mobile devices. *

- ○ very low
- ○ low
- ○ medium
- ○ high
- ○ very high

11. Did you notice that there was a Bitcoin testrun in the Mensa Binz in the week from 10th to 14th February of 2014? *

- ○ Yes
- ○ No

12. Did you inform yourself further about the testrun by asking involved people, by visiting the homepage or reading the poster in the entry hall. *

- ☐ Yes, Poster
- ☐ Yes, Homepage
- ☐ Yes, People
- ☐ No

13. Did you take part in this testrun? (E.g. payed in some bitcoins, bought some bitcoins, did transactions) *

- ○ Yes
- ○ No

14. How difficult was it to create an account *

- ○ very easy
- ○ easy
- ○ average
- ○ difficult
- ○ very difficult
- ○ I have not tried

15. How difficult was it to use MBPS: ...to pay in bitcoins from bitcoin network into MBPS *

- very easy
- easy
- average
- difficult
- very difficult
- I have not tried

16. How difficult was it to use MBPS: ...to send bitcoins (pay) *

- very easy
- easy
- average
- difficult
- very difficult
- I have not tried

17. How difficult was it to use MBPS: ...to receive bitcoins *

- very easy
- easy
- average
- difficult
- very difficult
- I have not tried

18. How difficult was it to use MBPS: ...to pay out bitcoins (from mbps to bitcoin network) *

- very easy
- easy
- average
- difficult
- very difficult
- I have not tried

19. How difficult was it to use MBPS: ...overall handling is easy and intuitive to use *

- very easy
- easy
- average
- difficult
- very difficult
- I have not tried

20. Which of the following incentives would make you use MBPS: I want to pay with bitcoins. *

- very important
- important
- average
- not important
- not important at all

21. Which of the following incentives would make you use MBPS: I want to be able to quickly do a payment without cash. *

- ○ very important
- ○ important
- ○ average
- ○ not important
- ○ not important at all

22. Which of the following incentives would make you use MBPS: I want to use the newest technologies. *

- ○ very important
- ○ important
- ○ average
- ○ not important
- ○ not important at all

23. Other comments (praise, suggestions for improvement, criticism)?

24. If you did not take part in the test run, what was the reason?

- ☐ I am not interested in Bitcoins at all.
- ☐ I do not own a mobile device which fulfills the requirements.
- ☐ I wasn't at Binzmühle at the time of the testrun.
- ☐ I was not aware of the testrun.
- ☐ Other.

Figure D.1: Questionnaire

Appendix E

News Roundup

The test run and especially the idea to pay with bitcoins in the University Mensa Binzmühle was picked up by some news web sites and printed media. The Faculty of Economics, Business Administration and Information Technology as well as the Institute of Informatics published details about the test run on their web sites, too. These stories and news are shown below.

Figure E.1: UZH - Business Faculty - February 10, 2014 - ”Testlauf: Mensa Binzmühle akzeptiert Bitcoins”

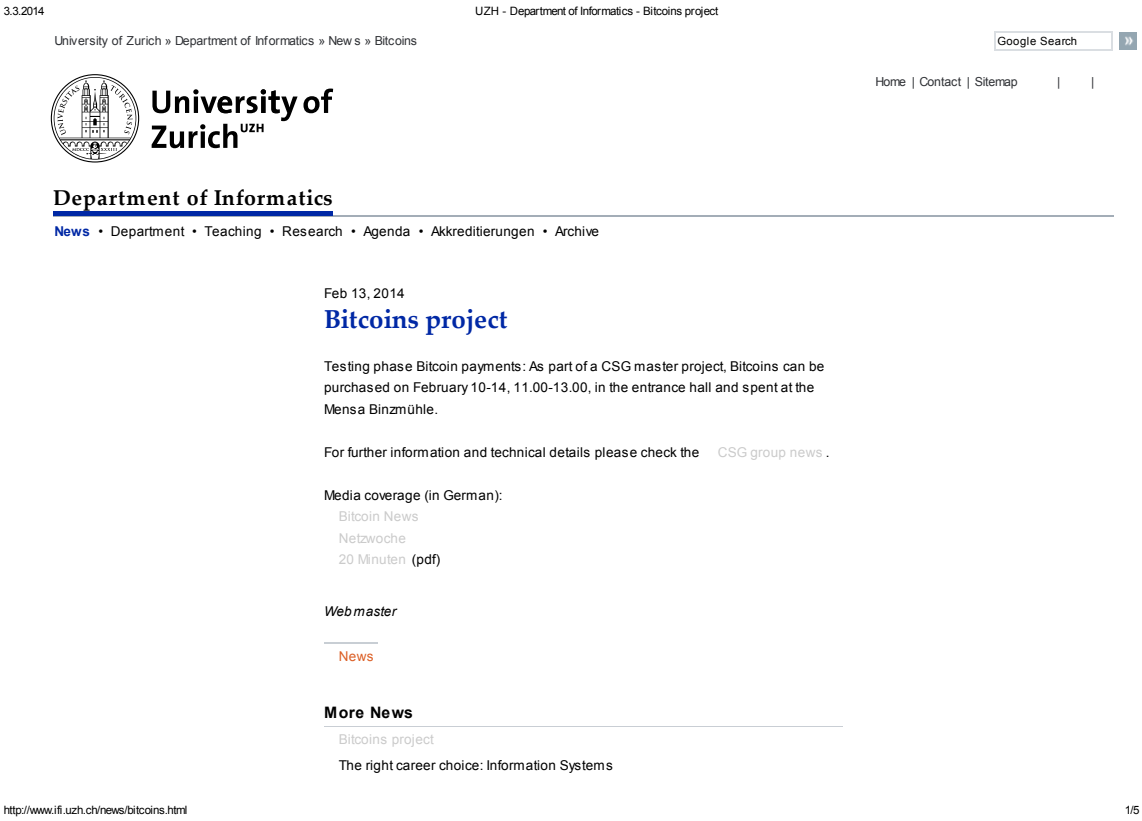


Figure E.2: UZH - Department of Informatics - February 13, 2014 - ”Bitcoins project”

Räuber fräsen Bancomat auf – über 100 000 Fr Beute

MELLINGEN. Unbekannte haben aus einem frei stehenden Bancomaten viel Geld gestohlen. Es ist innert Kürze bereits der zweite Vorfall dieser Art im Aargau.

Mindestens 100 000 Franken haben Räuber aus einem frei stehenden Bancomaten beim S-Bahnhof Heitersberg ausserhalb von Mellingen AG erbeutet. Sie frästen mit einem Winkelschleifgerät ein Loch in die Türe und drangen ins Innere der runden Kabine ein. Dort demolierten die Täter die Alarmanlage und brachen den Geldbehälter auf. Die Höhe des Sachschadens kann Bernhard Graser, Sprecher der Kantonspolizei Aargau, noch nicht beziffern.

Bereits vor zwei Wochen hatten Unbekannte einen frei stehenden Bancomaten der Aargauischen Kantonalbank (AKB) in Oberentfelden AG auf die gleiche Weise geknackt und viel Geld erbeutet. Ob es dieselben Täter waren, kann Graser nicht sagen: «Klar ist, dass Profis bei den videoüberwachten Automaten am Werk waren.» Weitere ähnliche Fälle sind ihm nicht bekannt. Im Kanton Zürich wurden laut der Zürcher Kapo bisher noch keine frei stehenden Bancomaten aufgefräst.

Die neue Masche beunruhige die Banken, sagt AKB-Sprecherin Ursula Diebold: «Wir werden die Sicherheit an den frei stehenden Automaten überprüfen, die vereinzelt im Kanton stehen.» Bei der Raiffeisenbank hat man dies laut Sprecher Franz Würth bereits getan: «Sie sind technisch auf einem guten Stand.» Trotzdem werde man nochmals über die Bücher gehen – geplante Massnahmen gibt er aber nicht bekannt. SOM

Die Täter knackten diesen Bancomaten bei Mellingen. KAPO AG

Eindringling attackiert Mieter mit Messer

ZÜRICH. Beim Verlassen des Hauses an der Limmatstrasse 210 ist ein Mieter von einem unbekannten Mann mit einem Messer angegriffen und verletzt worden. Passiert ist es in der Nacht auf gestern kurz nach 24 Uhr: Als das Opfer die Haustüre öffnete, wollte der Unbekannte ins Gebäude dringen. Als der Mieter ihn aufzuhalten versuchte, zückte der Täter unvermittelt ein Messer und stach zu. Das Opfer erlitt laut Mitteilung der Stadtpolizei Zürich mittelschwere Schnittverletzungen an der Hand – es befindet sich im Spital. Nachbarn sind schockiert, aber nicht verwundert: «Seit klar ist, dass das Haus bald renoviert wird, trifft man im und vor dem Treppenhaus öfter auf komische Gestalten», sagt eine Bewohnerin. Die Polizei sucht Zeugen. ROM

Mädchen (12) wird vermisst

TURBENTHAL. Die 12-jährige Ambar Diaz wird seit gestern Vormittag in Turbenthal vermisst. Das Mädchen wollte am Morgen kurz nach acht Uhr von der Schule Hohmatt nach Hause, kam dort aber nie an, wie die Zürcher Kantonspolizei mitteilte. Die Schülerin sei nach Hause geschickt worden, sagte eine Kapo-Sprecherin auf Anfrage. Über die Gründe machte sie keine Angaben. Das dunkelhäutige, 151 Zentimeter grosse Mädchen, das Zürichdeutsch spricht, trägt laut Mitteilung eine schwarz-weiss-karierte Jacke, dunkle Jeans und eine schwarze Mütze. SDA/ROM

Mensa nimmt Bitcoins an

ZÜRICH. In der Uni-Mensa Binzmühle in Oerlikon kann man während einer Testphase bis zum 14. Februar mit der Internetwährung Bitcoins bezahlen. In einem Masterprojekt an der Universität Zürich haben Studenten ein mobiles Bezahlsystem MBPS (Mobile Bitcoin Payment Solution) entwickelt. Dieses ermöglicht Transaktionen mittels NFC (Near Field Communication). Die Applikation wird mit Bitcoins aufgeladen, die unter anderem im Gebäude vor der Mensa bezogen werden können. BLU

Figure E.3: 20 Minuten - February 12, 2014 - ”Mensa nimmt Bitcoins an”

netzwoche

12.02.2014 12:29 (MARCEL MAURICE URECH)

Testlauf

Mensa an Uni Züri akzeptiert Bitcoins

Die Mensa Binzmühle der Universität Zürich (Quelle: Screenshot von mensa.uzh.ch)

Studenten der Universität Zürich können ihr Mittagessen in der Mensa Binzmühle mit der virtuellen Währung Bitcoin bezahlen. Um dies zu ermöglichen, haben Studenten ein mobiles Bitcoin-Bezahlsystem aufgebaut.

Die Mensa Binzmühle der Universität Zürich in Oerlikon wird noch bis nächsten Freitag die virtuelle Währung Bitcoin akzeptieren. Möglich gemacht haben dies drei Studenten des Instituts für Informatik. Sie haben im Rahmen eines Masterprojekts ein mobiles MBPS (Mobile Bitcoin Payment Solution) aufgebaut. Das Bezahlsystem basiert auf der drahtlosen Funktechnologie NFC (Near Field Communication) und setzt ein Smartphone mit Android ab der Version 4.4 voraus.

Am Projekt beteiligt sind die Studenten Jeton Memeti, Mehment Ali Bekooglu und Simon Kaeser unter der Leitung der Dozenten Thomas Bocek, Christos Tsiaras und Burkhard Stiller.

Webcode :: http://www.netzwoche.ch/000-000-246

Figure E.4: Netzwoche - February 12, 2014 - ”Mensa an Uni Züri akzeptiert Bitcoins”

IT-MARKT

Marcel Maurice Urech
TEAM

12.Februar 2014 - 12:29 | Updated 12.Februar 2014 - 12:31
Testlauf

Mensa an Uni Züri akzeptiert Bitcoins

Die Mensa Binzmühle der Universität Zürich (Quelle: Screenshot von mensa.uzh.ch)

Studenten der Universität Zürich können ihr Mittagessen in der Mensa Binzmühle mit der virtuellen Währung Bitcoin bezahlen. Um dies zu ermöglichen, haben Studenten ein mobiles Bitcoin-Bezahlsystem aufgebaut.

Die Mensa Binzmühle der Universität Zürich in Oerlikon wird noch bis nächsten Freitag die virtuelle Währung Bitcoin akzeptieren. Möglich gemacht haben dies drei Studenten des Instituts für Informatik. Sie haben im Rahmen eines Masterprojekts ein mobiles MBPS (Mobile Bitcoin Payment Solution) aufgebaut. Das Bezahlsystem basiert auf der drahtlosen Funktechnologie NFC (Near Field Communication) und setzt ein Smartphone mit Android ab der Version 4.4 voraus.

Am Projekt beteiligt sind die Studenten Jeton Memeti, Mehment Ali Bekooglu und Simon Kaeser unter der Leitung der Dozenten Thomas Bocek, Christos Tsiaras und Burkhard Stiller.

Figure E.5: IT-Markt - February 12, 2014 - ”Mensa an Uni Züri akzeptiert Bitcoins”

News.pdf

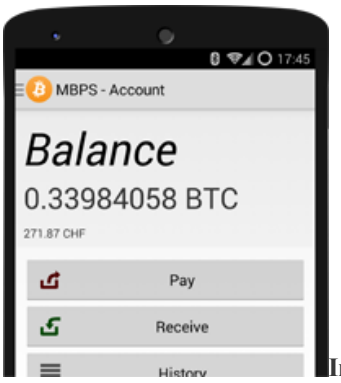
Feb
12

By Christian Maeder

Tweet

Universität Zürich startet Bitcoin Testphase

Category: Bitcoin | Tags: Bitcoin, MBPS, Mensa, NFC, Universität Zürich, UZH | 1 Comment



In der Uni-Mensa Binzmühle in Oerlikon kann in einer Tesphase bis zum 14. Februar in Bitcoin bezahlt werden. Dies ermöglichen Studenten im Rahmen eines Masterprojekts.

Für dieses Projekt wurde das mobile Bezahlsystem MBPS (Mobile Bitcoin Payment Soloution) entwickelt. Dabei werden die Bitcoins mittels NFC übermittelt. Die Bitcoins können im Gebäude der Mensa mittels einer Aplikation aufs Handy geladen werden.

>> Download der App für Android hier

Figure E.6: Bitcoin News - February 12, 2014 - ”Universität Zürich startet Bitcoin Testphase”

BITCOIN

Universität Zürich startet Bitcoin-Testphase

Bild: Bitcoin News

«In der Uni-Mensa Binzmühle in Oerlikon kann in einer Testphase bis zum 14. Februar in Bitcoin bezahlt werden», schreibt bitcoinnews.ch. Der Bezahlvorgang geschieht über eine Android-App.

Die Bitcoins können an der Uni mit der App auf das Smartphone geladen werden. Konkret wird die virtuelle Währung über NFC auf das Smartphone transferiert. Studenten der Uni Zürich haben hierzu das mobile Bezahlsystem MBPS (Mobile Bitcoin Payment Soloution) entwickelt. (oli)

Figure E.7: Watson - February 12, 2014 - ”Universität Zürich startet Bitcoin Testphase”

Appendix F

MBPS Installation Guidelines

F.1 Client Installation

1. Consider hardware requirements: Android 4.4 device with NFC
2. Download and install MBPS application from:
   - Google Play Store: `https://play.google.com/store/apps/details?id=ch.uzh.csg.mbps.client`
   - MBPS Homepage: `http://bitcoin.csg.uzh.ch/downloads`
   - Source: `https://github.com/MBPS-Project/mbps_client`
3. Register your MBPS account
4. Verify your email address
5. Login to MBPS with your username and password

F.2 Server Installation

MBPS is developed and tested for the software versions mentioned in the listing. MBPS should also run with newer versions, but there is no guarantee as not tested. Only up to date and stable software versions should be used to run MBPS.

1. Set up a Linux server (tested with Ubuntu 12.04) with Java 1.7 (tested with version 1.7.0_51)
2. Install and set up Apache Tomcat 7 on your server (tested with version 7.0.26)
3. Install Postgresql 9 (tested with version 9.1.12)
4. Install official Bitcoin Client *bitcoind* (tested with version 0.8.6)

5. Configure bitcoind configuration file (default: `/home/username/.bitcoin/bitcoin.conf`) to allow local RPC connections, define RPC username and password
6. Run bitcoind, encrypt your wallet and wait for successful download and verification of blockchain
7. Checkout MBPS source code from Github and import to Eclipse (make sure Maven is installed)
   - Server: `https://github.com/mbps-project/mbps_server`
   - Shared-Resources: `https://github.com/mbps-project/mbps-shared-ressources`
8. In the server package adapt `Config.java` and `hibernate.cfg.xml` to match your configuration (server ports, usernames, passwords etc.)
9. Export server application as `.war` file (e.g., `Server.war`)
10. Deploy and run `.war` file on Apache Tomcat server

Appendix G

Contents of the CD

The CD-ROM contains the following files:

- **Abstract.txt**
  English version of the abstract
- **Zusfsg.txt**
  German version of the abstract
- **mbps_client.zip**
  Source code of the Android mobile client application
- **mbps_server.zip**
  Source code of the server application
- **mbps-shared-resources.zip**
  Source code of the shared resources. This library contains resources which are used by the client and the server, e.g., model classes.
- **Report.zip**
  Figures and LaTeX source files of this report
- **Report.pdf**
  PDF version of this report
- **Presentation.pptx**
  The slides for the final presentation
- **Presentation.pdf**
  The slides for the final presentation in PDF format