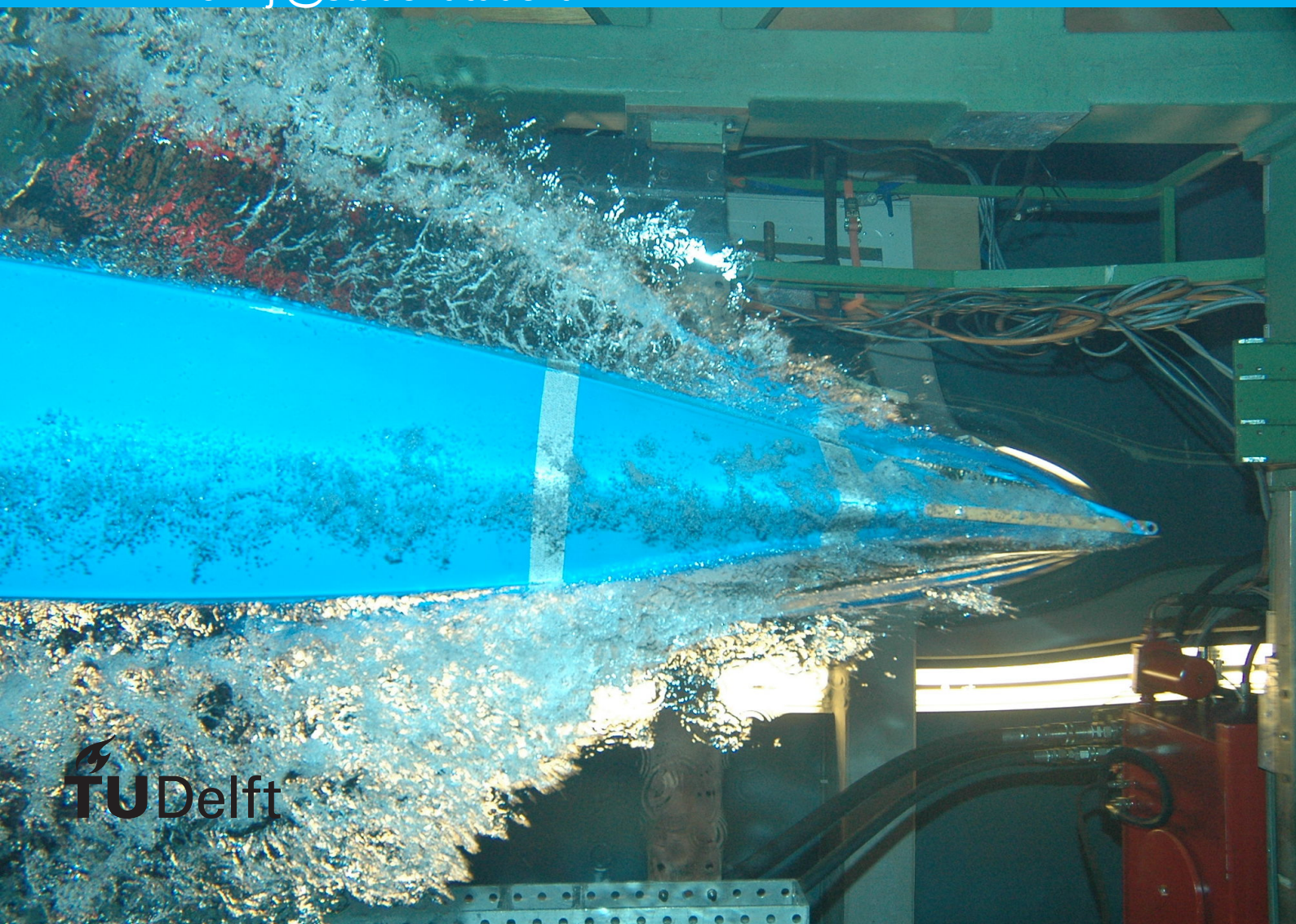
# EuroToken

## A Central Bank Digital Currency (CBDC) with off-line transfer capapabilities


## R. W. Blokzijl


- Stablecoin
- Blockchain
- Cryptocurrencies
- TrustChain
- CBDC


R.W.Blokzijl@student.tudelft.nl

TU Delft

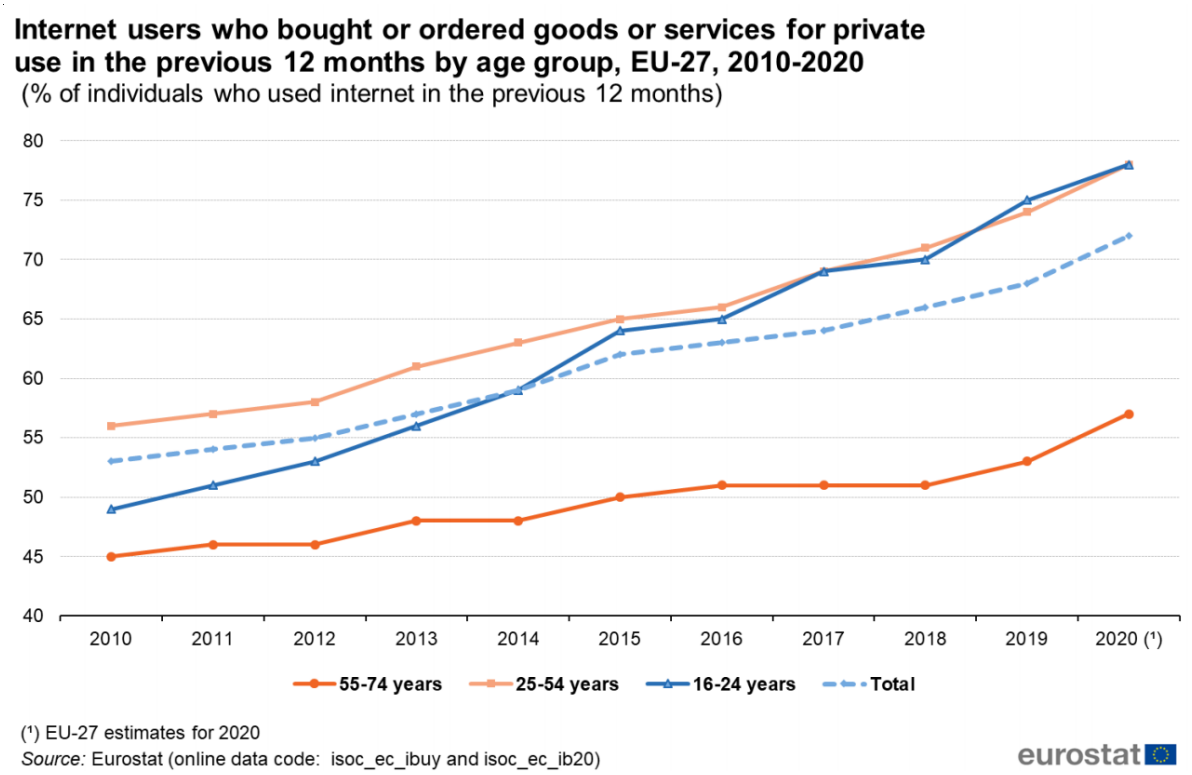



Figure 1.1: E-commerce 2010-2020 [**?** ]

of their coin, Tether Holdings Limited acts as a centralised middle-man exchanging 1 tether for 1 dollar. Centralised stablecoins are often seen as an intermediary solution that provides a wrapper over the old monetary system in order to extend it with the features of digital currencies. These coins are essentially financial derivatives that depend on already existing currencies.

On June 18, 2019, a new currency conceived by a group of Facebook engineers was announced under the brand name “Libra” [**?** ]. Later renamed to Diem, it would be a new free floating currency managed and governed by a consortium of multi-national companies united under the banner of the Diem association. While Diem presents itself as a solution for the worlds 1.7 billion unbanked, it is essentially a private world currency that would be controlled by corporations who will not be accountable to democratic processes.

The battle for the future control of monetary systems is a world wide phenomenon. While distributed open-source communities of engineers are trying to create a system free of corruption and private interest groups are trying to extend their reach, governments around the world are beginning to realise the threat to the established order. In order to not lose their influence of over their respective economies, governments around the world are looking into new digital versions of their currencies. The Federal Reserve has published their “Preconditions for a general-purpose central bank digital currency” [**?** ]. The ECB has published a report specifying a number of Reasons to issue a digital euro, scenarios and implied requirements [**?** ]. Meanwhile, the government that has perhaps progressed the farthest is the Chinese government. With successful public trails [**?** ] they seem to be closest to a working digital currency.

## 1.4. The technical debt of traditional finance

People have been trading various commodities as a store of value nearly since 6000 BC [**?** ]. Since then money has taken various forms, slowly moving up layers of abstraction, but the function of money has always stayed the same: acting as a medium of exchange [**?** ].

The first known true standardised gold coins have attributed to Lydian society back in 640 BC. Slowly over time the gold and silver contents of the coins became less important, and currencies as a proxy for trust, became slowly more dependent trust in the system rather than the real value of the

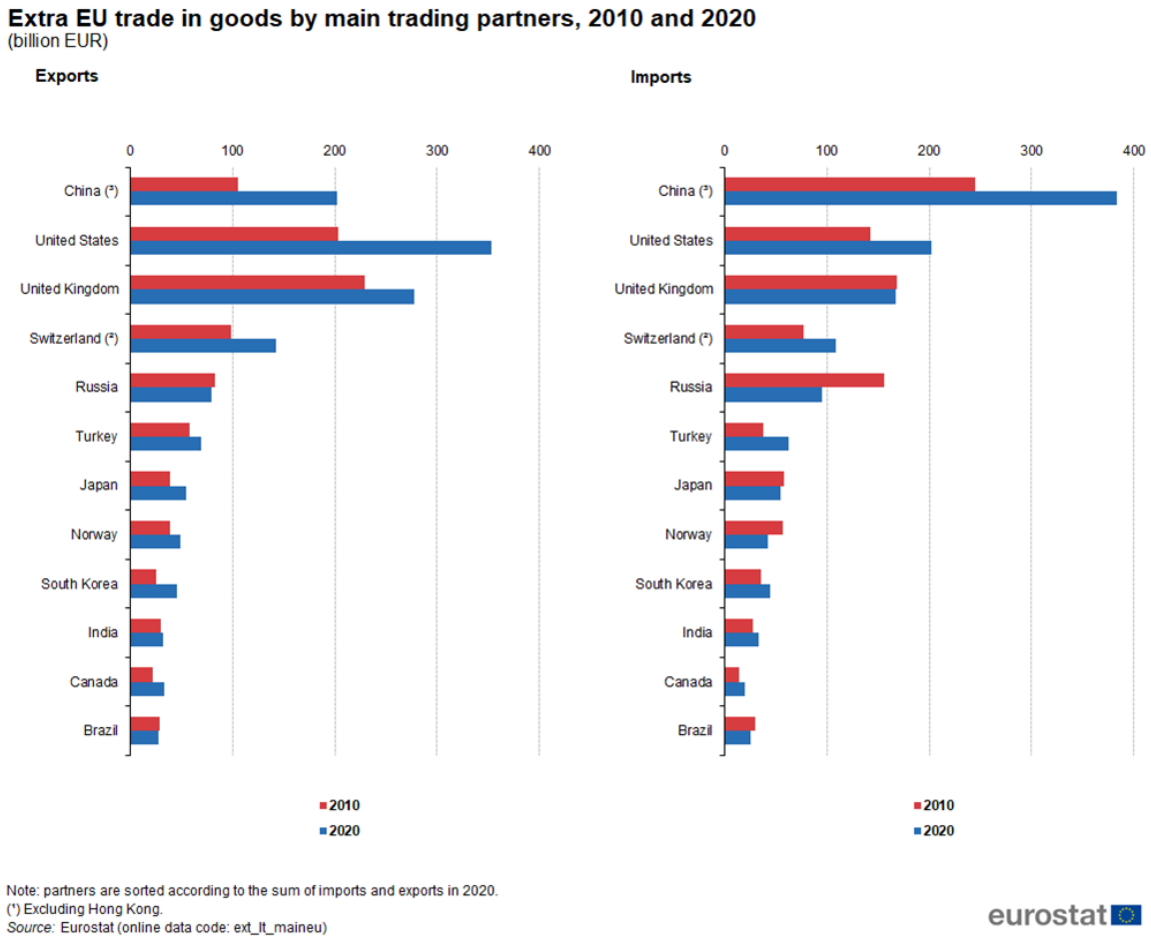



Figure 1.2: E-commerce 2010-2020 [**?** ]

currency. In 806 AD, this culminated in the first use of paper money in China. Since these events, the form of money has varied based on societal conditions. Because of its functional utility, money in the form of bank notes backed by gold has been a popular form of money that has been used in European society since 1440 [**?** ].

At the end of the 1900s Europe had a system of many national currencies, denominated per country. The value of the currency was maintained by different countries, often using the gold standard[**?** ]. Meanwhile private banks would allow people to store their money in a safe institution, while they could lend that money out. For international and cross-currency trade, people would swap different currencies in exchanges when needed, but because of the localised nature of society this was infrequent.

This system worked fine in an era where most exchange was done by cash, and most trade was done within national borders. But as the world became more connected and digital, different solutions had to be built on top of the old system to respond to the changing demands of the population. Private banks, who were once used only for large transactions and money storage, got a more and more central role in day to day transactions. In effect the system as we see it today has accrued a lot of technical debt as the its requirements changed over time. As a result a number of inefficiencies have emerged from this technical debt.

First, international money transfer. Even within the euro-system this can often take up to a week. Second, banks as private institutions are vulnerable to bankruptcy. This makes them a “financial”on their semi-central point of failure, that take a good portion of the financial system with it. Third, transactions are dependent on bank IT systems. This makes them a “technical” semi-central point of failure, potentially leaving people unable to purchase their essentials. Fourth, people are tightly coupled to their banks. While having multiple bank accounts is possible, a lack of standardisation and interoperability makes people easily dependent on their one bank and its features.

# 2

# Problem description

Can we create a digital currency that mimics the properties of Central Bank backed cash? This is one of the questions that keeps the ECB busy.

## 2.1. Double-spending prevention vs Scalability

The search for a safe digital money can be traced back as far as David Chaum in 1983 when he first released his paper on Blind Signatures for Untraceable Payments [**?** ]. In this paper Chaum does not specify a design for a decentralized currency, but a mechanism for preserving user privacy against third parties in digital transactions. Since then many implementations have been attempted, including eCash [**?** ] [**?** ]. However, in its competition with Credit Cards, eCash went bankrupt in 1998 [**?** ].

In 2008, Satoshi Nakamoto publised the design for Bitcoin. Bitcoin was the first digital curren removes third parties out of the equation by maint

Since then, various attempts to create new digital currencies haev

David Chaum since chaum in 1983

[**?** ] pow [**?** ]

The approximated global knowledge of single blockchain networks like Ethereum[**?** ] and Bitcoin[**?** ] lead to limited transactions per second and

image tps ethereum

[**?** ]

[**?** ]

In order to create a fully functional system of accounting thea

How to prevent double spending:

- How banks solved it
  - How its not transparent nor open
- How Bitcoin solved it
  - How blockchain is not scalable
- How Iota solved it
  - How the block-DAG might work well
- How Nano solves it
  - How the block-lattice is great but still requires centralised nodes

## 2.2. The price stability problem

- How stablecoins work
- Why the euro is stable
- Why bitcoin is not

## 2.3. Requirements for a digital euro by the ECB

In October 2020 the European Central Bank published a report detailing a number of scenarios where a new digital euro could provide a benefit [**?** ]. Associated with these a number of requirements are provided.

1. **Enhanced digital efficiency**
2. **cash-like features**
3. **competitive features**
4. **monetary policy option**
5. **back-up system**
6. **international use**
7. Minimise ecological footprint (cost saving and environmentally friendly)
8. **ability to control the amount of digital euro in circulation.**
9. cooperation with market participants
10. compliance with the regulatory framework
11. safety and efficiency in the fulfilment of the Eurosystem's goals
12. easy accessibility throughout the euro area
13. conditional use by non-euro area residents

In this project we aim to conform to these requirements as best we can and we evaluate our solution by these requirements. The technical requirements are emboldened in the list, as they will be guiding in our design. The rest will only be speculated on as they do not pertain to the topic of computer science and fall outside of our area of expertise. Therefore a technical solution to these problems has to conform to the following requirements.

1. Be a fully functional system of accounting
2. Preventing unsanctioned money creation
3. Scale to the size of the European union
4. Disaster resilience through off-line transfer ability

## 2.4. Research Focus and Structure

While a complete redesign of Europe's monetary system is obviously out of scope for this thesis, the previously described problems and requirements lead us to the following question:

---

Can we create a digital currency that mimics the properties of Central Bank backed cash.

---

This document describes the motivation, design, implementation and evaluation of the EuroToken system. The EuroToken system is a conceptual design that aims to fit the requirements stated in the previous section, as well as a limited proof of concept design testing certain aspects of the design. The structure of this work is as follows, in the next chapter we describe the design of the EuroToken system. The design is approached from the fundamental questions of a currency and answers the fundamental questions first. What is a digital currency? What is the double spending problem? And how to design a system that is scalable while not compromising the principle of double-spend prevention? The design aims to provide the following features:

1. Be a fully functional system of accounting
2. Preventing unsanctioned money creation
3. Scale to the size of the European union
4. Be off-line transferable

In order to not be limited in the same way as Bitcoin and similar currencies, we choose to sacrifice the following feature: Decentralization. This give us the required leeway to create a scalable and offline-capable system. However, we do attempt to provide the necessary tools to overcome the downsides of the centralisation.

# 3

# Design

Any payment system that aims to replace public money while being able to operate at the scale of the euro system needs to conform to a number of requirements. Such a system needs to be scalable, privacy aware, allow peer to peer transactions off-line. In needs to be price stable, exchangeable for euros, and most importantly, it needs to be secure and cheating resistant. In this chapter we first describe how a distributed block-lattice provides a good basis for a scalable, private, and off-line friendly transaction system. We then explain how we position the system in relation to the euro, how the price can remain stable, and how a system can mimic the properties of cash. We then go in to the details of how the system is secured, and how we prevent double spending while still remaining scalable and allowing off-line transactions. Finally, we explain how the system could be expanded upon by legal frameworks that can provide varying risk vs privacy trade-offs and how certain guarantees could be enforced in the system.

## 3.1. Distributed accounting and networking

The possibilities and limitations of any virtual currency are dependent on its system of accounting. In order to conform to the off-line, scalability and transparency requirements, a system of distributed accounting is chosen. As the fundamental building block for the EuroToken system we use a Hyper-Sharded block-lattice that keeps track of every users transaction history on their own edge device. By storing all information required for transacting at the physical end points of transactions, we create the possibility of direct off-line transaction between users, without any link to the outside world.

While the EuroToken system design is independent of the underlying communication technology, the off-line requirement leads to there being some limitations on the way users interact. Since off-line users cannot connect to servers we choose to work with a Peer to Peer system that allows users to find each-other based on personal identifiers.

We build on Peer to Peer networking, that provides a mechanism to discover the network location of users based on the same public key that is used to identify their wallet. This allows us to almost completely abstract away from locating users using IP addresses and ports. As a result we only have to worry about maintaining a users public key to identify and communicate with them across time. Our peer to peer network does not only abstract away from IP addresses, bus also from the IP network completely. Namely, it provides communication over Bluetooth without the need for any internet connection. This becomes very useful for demonstrating the off-line capabilities of the EuroToken system.

## 3.2. Block-lattice accounting

As mentioned for our distributed accounting system we choose to build on a block-lattice stucture. As illustrated in figure **??** every user has a personal blockchain structured as a chronological, one-dimensional string of “blocks”. Every block will include a cryptographically secure hash identifying what block preceded it. Because of the trapdoor effect of the hash, any block will uniquely identify all blocks that come before it. This allows anyone to verify the validity of entire history of another user, given the last block in this history. Every block will contain a single transaction that specifies the transfer

of funds from one user to another, as well as a reference to a corresponding block in the chain of the transaction counterparty. This effectively creates a system of double accounting.

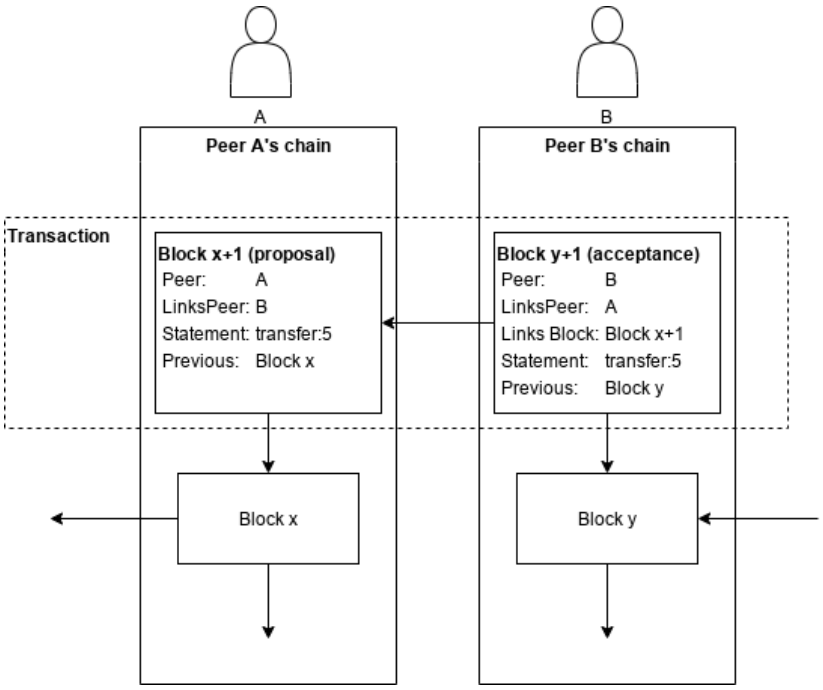



Figure 3.1: Block-lattice structure

Every block can contains a “declaration” by the user, or a reference to the declaration of another party. These declarations are digitally signed by the declaring party and form the base of any transaction. To do a transaction the sending user (Alice) will create a new (half)block with a declaration stating “I transfer 1 EuroToken to Bob”.

When Bob receives this block from Alice, he can accept it by creating a block in his own chain and returning it Alice. Before Bob accepts the block, he first validates the history of Alice by requesting enough of her chain make sure that Alice doesn’t validate any of the network rules that would invalidate Bob’s receiving of the money. Once Bob is satisfied with the correctness of Alice’s transaction history he incorporates a new block declaring the acceptance of Alice’s transaction. This block includes the hash of Alice’s block, thus entangling the chains of Alice and Bob together. Bob now has a signed proof by Alice that the transaction happened. He can use this to prove the transaction happened at any point in the future.

## 3.3. Gateways: Euro to EuroToken exchange

The viability of any currency as a store of value over a given time frame is dependent on the stability of its price over that time frame. This is an issue that has plagued decentralised crypto currencies from the very beginning. The hope is that the currency will stabilise itself when it reaches a critical adoption level. However even currencies like the euro and US dollar don’t remain stable without periodic interventions of their respective central banks.

The euro has long served as the core of the financial infrastructure of the European economy. It has essentially done this using two consumer facing versions of money: the euro as a publicly accepted, physical item of value (the public euro), and the euro as a digital, privately managed, unit of account (the private euro). These public, and private types of money serve citizens in different ways. The public euro is the most stable store of value since its guaranteed by the central bank, it also has the advantaged of requiring no internet connection to use. While the private euro has digital advantages in usability and security, but derive their value from the “reliability” of private banks, and are only insured by governments up to 100.000 euros [CITE]. With the declining usage of public money in favor of digital money, the need for a new type of euro to fill the gap of public money is getting stronger.

For these reasons we present the EuroToken system as a 3rd type of money. Instead of reinventing

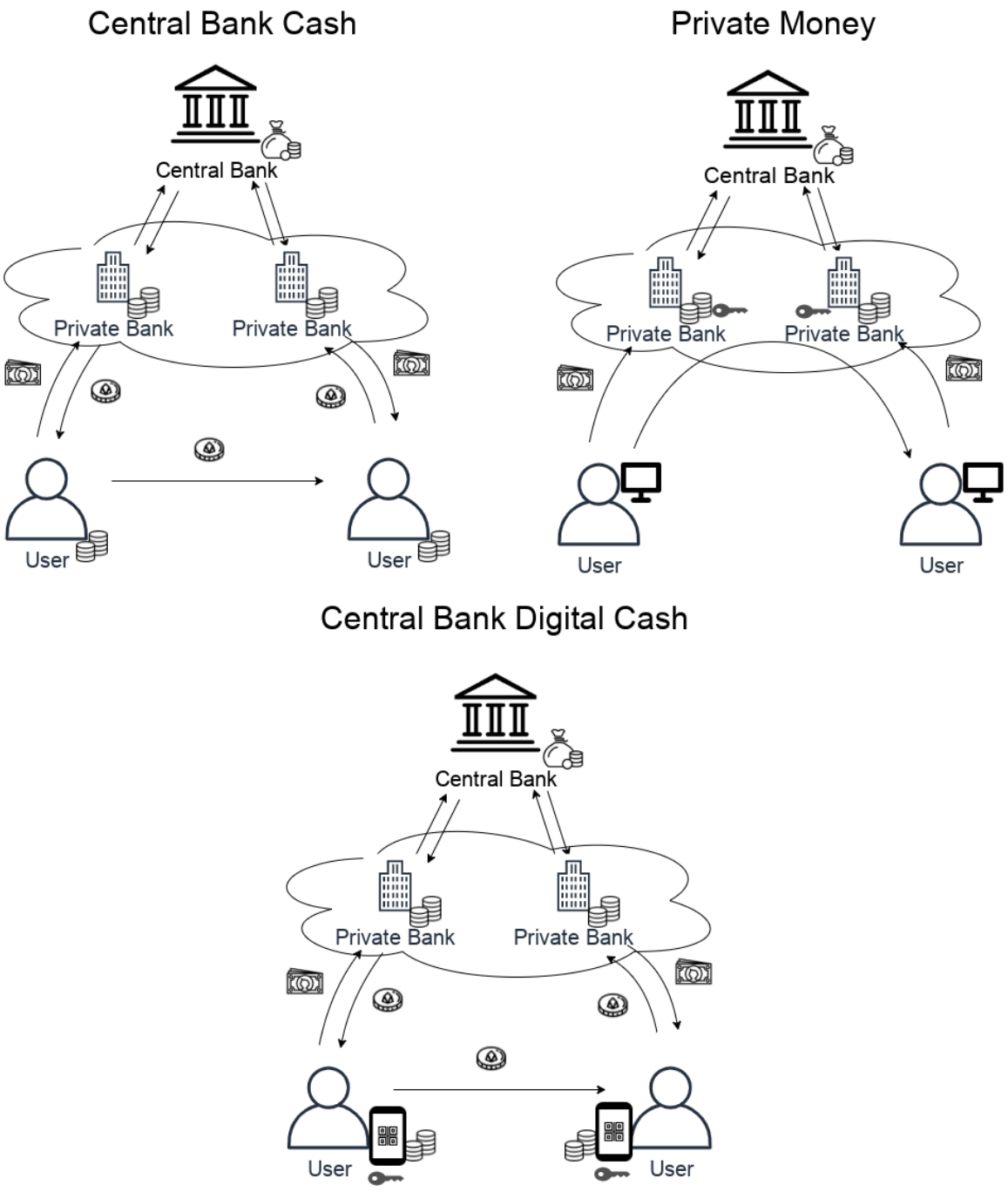



Figure 3.2: Usage flow of cash, private money, and EuroToken.

the wheel of “stability” we connect the EuroToken system directly to the euro system, while providing extra features on top of the current euro system.

In order to properly connect EuroTokens to the euro system, an easy and value-transparent method of exchange is required. Just like private and public euros are exchangeable though local banks, a mechanism is needed to exchange between euros and EuroTokens at a 1:1 ratio. To do this, we implement a “gateway” between the private euro system and the digital euro. This gateway implements the EuroToken protocol on the one hand, and interfaces with banks on the other.

In our current design, the gateways are designed to be run by public parties associated with the central bank. Any detailed speculation on the best way to connect such a system to the established euro is best left to economists. However, we envision a possible future where multiple gateways are run by existing private money institutions who perform the heavy lifting of day to day exchange. In such a system private banks would act as an accounting system for the EuroToken exchange without being allowed to leverage their EuroToken position. The Central bank would allow these private institutions to conform to reserve requirements in the form of EuroToken holdings rather than only cash. By not allowing private banks to mint new EuroTokens, but only exchange them, the central bank can control the amount of EuroToken in circulation in a similar way to current public money. This would insulate the EuroToken from the impact of a failing euro or bank, in the same way as physical public money is currently insulated from such failing.

This way of connecting to the euro could allow for a smooth transition to a digital form of public money, while the established and regulated financial institutions are still positioned properly in a place where financial services can be provided.

## 3.4. Transaction finality and Double-spending

In order to remain a viable store of value, a currency needs to provide protection against any non-sanctioned creation of that currency. If a network allows its users to “create” new money in any significant way, the value of the coin will drop as the supply increases, thus undermining one of the most fundamental function of the currency. The structure of the blockchain provides an immutable and signed history of any transactions, thus enabling users to prove that the funds they are attempting to send actually exist. However the blockchain does not inherently allow users to prove that they have not spent, and will not spend, the same balance again.

In this section we explain how the network prevents unsanctioned creation of currency.

### 3.4.1. The double spending problem

In order to spend their money twice, a user has to create 2 blocks that are positioned in the same place in their blockchain. This is what is called a “double-spend attack”. This attack is only detectable if both of the conflicting blocks are found. Since we have opted for a distributed blockchain this detection becomes a non-trivial problem to solve. The transactions of 2 conflicting blocks might be re-spent many times by the time anyone sees the 2 conflicting blocks and notices that a double spend happened.

Bitcoin and similar currencies solve this problem using a global blockchain that everyone has access to. This allows users to check whether a given balance has already been spent by inspecting the global database of transactions. However, the global knowledge of the Bitcoin chain is inherently un-scalable. Additionally, the details of the Proof of Work method of block generation leaves a certain measure of uncertainty with regards to the “finality” of any transaction in the newest blocks. This often requires users to wait up to an hour to be sufficiently confident their transaction really happened.

A solution to this problem in a network with distributed blockchains, starts with the realisation that the issue of detecting double-spending can be reduced to the issue of detecting “chain forking” in our network. The usage of the blockchain allows us to make sure that all transactions are ordered and consistent, this means that double-spend needs to be in 2 separate versions of that history. Thus requiring 2 blocks that refer back to the same historic block. This is a fork in the chain. We cannot “prevent” a user from creating 2 conflicting blocks in their chain as their chain is stored on their own device. But we can make sure that the rest of the network only accepts one of the 2 blocks, thus only accepting 1 “spending” of the balance. This choice between 2 conflicting blocks needs to be consistent so anyone in the network is working with the “same history”. Additionally, forks need to be detected and resolved before the balance is spent again by any of the 2 receiving parties. This way a double-spend will not propagate into the network and is limited to the users involved in the 2 transactions. To resolve

the conflict between blocks we define the concept of “transaction finality”. For a transaction to be final, it needs to be “validated” and “stored” in the network, while any conflicting transaction will be rejected by the network. Transaction finality the guarantee that a merchant needs before they can send their goods to a paying customer.

The transaction finality problem in our network has several possible solutions. In [**?** ] Brouwer presents a method of distributing blocks to a randomly and fairly selected list of witnesses that would probabilistically detect any conflicting block before the receiver would accept them. In [**?** ] Guerraoui et. al present a more theoretical method of block broadcast. These might be good candidates for future research. However since these solutions are inherently probabilistic, there is no hard guarantee that any double-spend will be detected in time.

### 3.4.2. Balance vs spendable balance

Currently lacking a good exact and distributed solution, we choose to utilize a decentralized network of trusted validators. These validators maintain the last transaction of users that register with them. Any user who receives money, can verify the non-existence of a conflicting block with the associated validator of the sender.

In the rest of this section, we define the concepts of “spendable balance” and specify the information requirements for marking a transaction as finalised.

In order for Alice verify if Bob is able to send her the money he is sending, she needs to know that Bob has sufficient funds. For this reason a rolling a balance across all transactions could be maintained across all blocks. Where the balance $B$ for a given block with sequence $i$ $(B_i)$ is:

$$B_i = B_{i-1} + C_i$$

.

Where $C_i$ is the change in balance for the block with sequence number $i$. This is negative when sending money. However the balance of a user does not take into account the concept of transaction finality. So instead we maintain the total “spendable balance” instead.

### 3.4.3. Finality statements

Before Alice can add the output of a block she received from Bob to her “spendable balance”, the transaction from Bob first has to be finalised. To achieve this a validation is performed with Bob’s associated validator. This is done by sending the validator a finality proposal.

**??** The finality proposal block includes notes a list of hashes that point to transactions from Bob. Together with this block for the validator to sign, Alice will send all of Bobs blocks from the last transaction to validate to the last block the validator knows about. The way for Alice to determine what information this is, is explained in the section on checkpointing later in this chapter. In addition to Bob’s blocks, she will also send her “accepting blocks” that include the transaction in her chain. This is to make sure she can only claim a transaction from Bob once. Bob’s validator will then verify:

1. That there are no other transactions that conflict with the one to Alice.
2. That there are no other “accpeting blocks” already linked to this transaction.
3. That Bob’s chain is valid up to the last transaction to verify.

If this is the case it will sign the proposal. If a later transaction from Bob is received that marks a fork in his chain, the fork from Alice becomes the only accepted fork, and the other one is rejected. Using this finality statements as proof of this, Alice is now allowed to spend the output of the transaction.

In the case that a different fork from Bob has arrived at the validator first, the fork where Alice receives money is rejected. Since Alice has already accepted the transaction in her chain and may have built other transactions after it (though not spent the output), she could be requested to submit a new finality proposal without this block. Since Alice is not permitted to spend the funds from Bob until it has been finalised this is the point where double spending is handled.

Note that the specific handling of this event might not involve the forfeiture of a transaction. We discuss this further in the section on off-line payments and conflict resolution.

### 3.4.4. Verification

For a block to be considered valid:

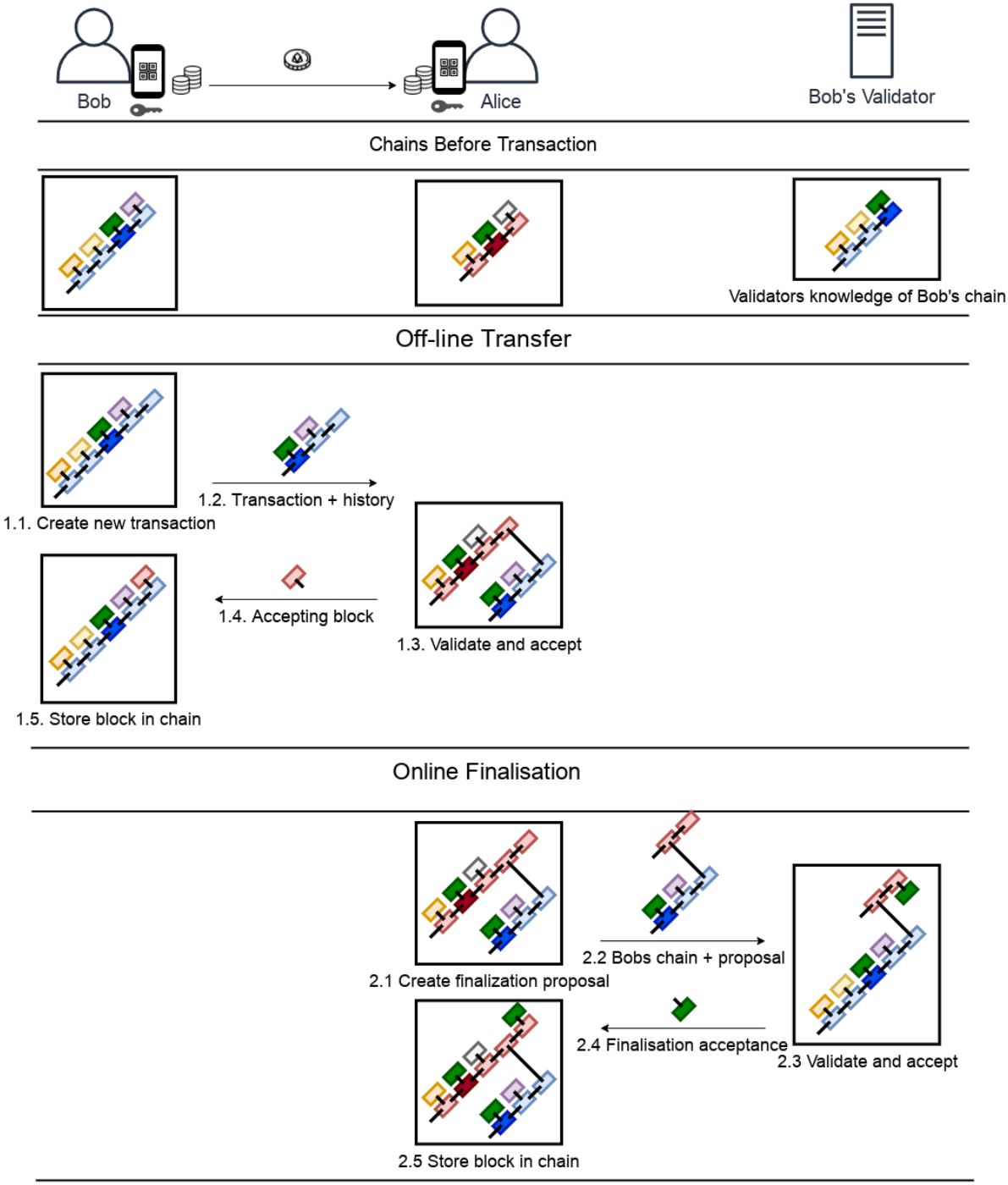



Figure 3.3: Off-line transfer and finalisation

1. All standard block-lattice invariants are maintained.
2. All blocks preceding it are verified to be valid
3. The total spent amount is to be less than the spendable balance.

For a transaction of a receiving block to be considered final:

1. A checkpoint from from the validator of the sender has to be exist in the chain of the user AFTER the transaction.

By introducing checkpoints, the required information at the point of transactions is reduced. When Alice and Bob set transact between them, Alice can determine the validity of Bob's transaction by inspecting only Bob's chain, down do his last checkpoint. However, Alice must also request all Bob's information down to the last Full checkpoint, in order to

### 3.4.5. Spendable balance

Once a transaction if finalised, "spendable balance" of Alice can be calculated. The spendable balance changes at two events, the finalisation of an earlier receiving transaction and when Alice spends her money. As such the spendable balance $SB_i$ for a given block with sequence number $i$ is:

$$SB_i = SB_{i-1} + F_i - S_i$$

Where $S_i$ is the total amount spent in the block with sequence number $i$, $F_i$ is the total amount finalised in the block with sequence number $i$.

### 3.4.6. Conclusion

In the future we envision the system to take one of three routes regarding transaction finality. First, system could be built on a future breakthrough in distributed transaction finality. Second the system could be built on a probabilistic but bounded transaction finality, where the rare double-spend is eventually detected and settled through the legal system. Or third, like in our solution, the system is build on trusted nodes that verify transactions for user. Like the gateways, these validators could be run by regulated financial institutions. Such a system would most resemble the current financial system, with the added benefits of off-line transactions, programmable money, a standardised system of accounting, instantaneous international transactions, etc.

## 3.5. Checkpointing

Because of transaction finality, when Alice receives the transaction from Bob, she can rely on the finality statements, rather than having to validate the chain of everyone he received money from. This reduces the validation load to only Bob's chain. However this still has some issues. First, Bob's chain will grow larger over time, thus slowly increasing the validation load. Second, all this information needs to be stored by Alice until it can be delivered to Bob's validator.

The way this problem has been solved in traditional blockchain systems is through the global blockchain and limited transactions per second. By having only miners or stakers being required to maintain the whole blockchain, only a few machines have to be able to know the entire chain and store all that data. But this is still inherently unscalable.

A second issue is one of privacy, when Bob has to send Alice all of his chain for verification, Alice can derive much from this information. Though we would like to see methods of privatization added to perhaps conceal transfered amounts, we still need a way to minimize the information leakage to 3rd parties.

To solve this issue of validation scalability, we define a form of checkpointing. We periodically create a checkpoint block in a users chain that , that includes a summary of the entire chain before it. This information is:

1. The total "spendable balance" at that point in the chain
2. The public key of the validator who is responsible for this wallet.
3. A statement that the validator has received all blocks before this point

Alice now knows the blocks that are already stored by the validator. When Alice is receiving money from Bob, she only requires Bob's blocks down to the his last checkpoint.

## 3.6. Off-line transactions and online validation

The EuroToken system has the intentional distinction between transactions and their finalisation. Because of this, the first step of transactions only require a direct connection between users. In theory, this allows to transact off-line, if they're willing to risk that a conflicting block already exists in the validator. Of course, in this case, the transfer of funds depends on the trustworthiness of the sending party.

In this section we discuss a few ways of interacting with the system that allows for different risk exposure to the parties.

### 3.6.1. Online transactions

When users are connected to the internet, a real life interaction can easily combine the finalisation step with the transaction, only transferring goods or services once the transaction is finalised. We envision this as the default way for users to interact, especially for large transactions, and transactions with strangers, since this reduces the risk to either party to zero.

### 3.6.2. Off-line transactions

Since money only becomes spendable after finalisation, the receiving user is the one that will lose funds when a double spend happens. To lower the risk and damage of this, certain systems might be put in place. For this, we build on the fact that transactions are always signed by both parties. This makes sure that a proof of double-spending always exists, and is obtained no later than the finalisation attempt.

A way to ensure a user that they will receive the funds is by allowing senders to register their identity with their validator. The validator would sign a statement that the identity of the sender is known and that they will take legal action in the event of a double spend. This then optionally allows the validator to accept the risk of double spending. In the case of a double spend the validator would sign a special statement with the receiver, that invalidates the double-spent transaction, but transfers and finalises the funds from the validator instead. The validator will then pursue legal action against the sender for fraud.

In the meantime the validator could block the sender to perform online transactions and checkpoints until they first settle the double spent funds. The details of what is both technically and legally possible here is a good subject for future research.

## 3.7. Regulation of validators

One could argue that system hasn't solved the issues of transaction finality and double-spending, and that we only defer the problem to a different point. It is entirely conceivable that trusted validators could cheat by allowing certain wallets to double-spend. To add to this, using the checkpoint functionality a validator can specify a higher spendable balance than is actually logged in the chain.

However, when comparing our system to the current way private institutions are regulated, the blockchain structure of transaction can provide a powerful method of maintaining the integrity of the institutions. While we cannot prevent fraud at the institutional level, we do provide an option for detection to allow for regulation.

In order for a regulator to check that a validator has done their job with integrity, they need to be sure of 2 things:

1. That all "statements" have been made consistently with the rules of the network.
2. That no other "statements" have been hidden from the regulator.

"Statements" in this context, describe anything that the rest of the network puts their trust in. These are:

1. Finality statements
2. Checkpoints

Both of these statements are created in the form of "accepting blocks" and are stored by users and their validators with an associated hash. We now propose a two round system for validating all transactions within a given time period. In round one, we validate that all statements have been made

correctly and publicly store the hashes. In the second round, once we have the hashes of all statements available, we validate that all statements from other validators exist.

In the first round all information in the database of the validator is processed for consistency. Since all statements by the validator are made in the form of blocks in their personal blockchain, they have an explicit order. The blocks of the validator, together with the blocks of all the users the validator is responsible for, are processed in the same way as the validator was responsible for processing them. This step in the process ensures that all statements are made correctly.

In the second round, we ensure that there is no statement withheld by the validator. This is done by publicly publishing a signed list of the hashes of all statements made by the validator. This allows regulators to cross-check that all inter-validator statements have been reviewed by a validator. To make this step more efficient, we propose that when checking a validators consistency, regulators generate a list of statements for each distinct validator to increase the efficiency of distributing these hashes to relevant parties.

A possibility also exists to allow the public access to these records to ensure the integrity of their institutions.

# 4

# Implementation

In this section we describe the implementation of the EuroToken protocol, as well as the prototype we built to test and showcase the capabilities of the EuroToken system. The protocol is implanted on top of IPv8. In includes an android/kotlin implementation as well as a python implementation. We then built a Euro to EuroToken exchange and transaction validator on top of the python implementation. On top of the kotlin implementation we built a wallet app that is fully capable of securely transferring eurotokens between wallets, as well as exchange them with the EutoToken exchange.

## 4.1. Architecture

The architecture of the EuroToken system has two main components. The gateway and the wallet. The gateway is managed by a central trusted party and fulfills two main functions from the design. These are to asynchronously validate transactions made by users, as well as handling the exchange between EuroToken and Euros. As such it maintains a bank account as well as its own wallet. The wallets are operated by each user, and they are fully capable of transfering funds between each-other without having to interact with anyone in the euro system.

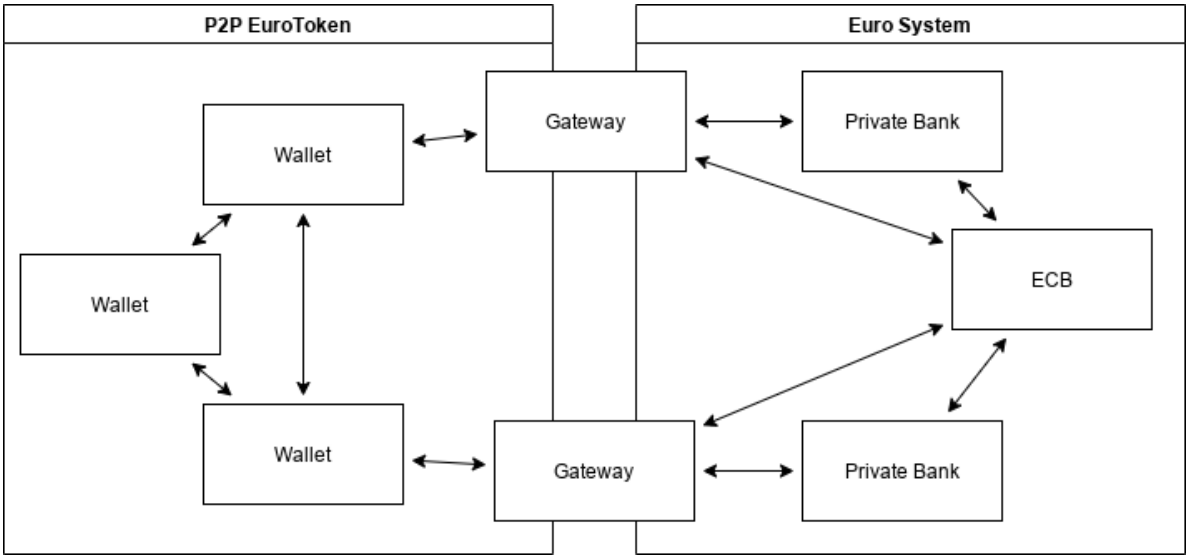



Figure 4.1: EuroToken architecture

In figure **??** we model the main communication channels. Within the P2P EuroToken system we have all wallet to wallet, and wallet to gateway communication. This communication happens directly between the communicating nodes using Peer to Peer technology.

In the Euro System, we make use of bank API's for gateway to bank communication. The communitcation with the ECB symbolises monatary policy enacted by the ECB on bank EuroToken reserve

requirements or possible direct exchange of euro for EuroToken.

Rather than implementing both an exchange as well as a validator we chose to implement and test these as a single entity. However, since the gateways roles might be split in the future the technical implementation of the gateway keeps the validator roles separate from the exchange roles. This results in a single EuroToken exchange software product, that is able to perform either or both of the functions.

## 4.2. EuroToken transfer protocol

The method for accounting and transferring of EuroTokens lies at the heart of this project. Because of this the choices regarding the implementation of the networking stack and blockchain technology will have a direct effect on the feature set and scalability of the whole EuroToken network. We need a network stack that allows communication both off-line directly between devices, as well as online across the world. Finding and connecting to any wallet without relying on central servers is a main requirement. In addition, the off-line transfer ability of the system is best demonstrated by creating an android client. Another requirement is therefore that an implementation is available for android as well.

One option is to implement a full blockchain protocol and associated network stack from the ground up to adhere to our exact requirements. This would give us a lot of say in the exact feature set of the network. However, since the science of distributed networking algorithms has mostly settled, most peer to peer communication technologies have already been implemented somewhere.

The second option is then to build upon some existing peer to peer networking library, while implementing the blockchain protocol ourselves. This option has some benefits as the usage of a block-lattice is not yet very common, and thus is not implemented as a stand alone package anywhere. For the P2P library we have several options. We considered libtorrent[**?** ], libp2p[**?** ] and IPv8[PyIPv8:online]. Libtorrent has a number of interesting peer to peer features like peer discovery and data transfer but sadly fell short when it comes discovery of peers based on public keys. It can be classified more as a file location protocol than a peer location protocol. This would mean we would have to implement a peer location system ourselves. Libp2p is a modular peer to peer networking stack that provides a large suite of p2p tools. Libp2p uses a Distributed Hash Table (DHT) to allow peer discovery based on a peer-id[**?** ]. There is an jvm/android implementation available, which also makes it possible to create an android client. Finally we looked at IPv8. IPv8 offers direct peer discovery based on public key and provides a framework for interaction called Overlay networks. Overlays provide a context for peers to interact within with particular message types. Crucially, IPv8 has an implementation in kotlin[**?** ].

Rather than implementing the blockchain mechanism ourselves, there is a third option. IPv8 includes a module called TrustChain. TrustChain is in essence a block-lattice type distrusted ledger technology. The technology does not fully solve double spending they way we originally designed it, so some work is required to adapt TrustChain to the EuroToken system, but it would provide a good basis for our implementation.

We choose to build on IPv8/TrustChain for this project as it allows us to build on their kotlin implementation for the wallet as well as the python implementation for the gateway.

### 4.2.1. TrustChain structure

Every user runs a *Peer* which consists of a public/private key pair as well as a collection of their *transaction* history in the form of their *blockchain*. The Peer can be uniquely identified by their public key. Every statement made by the peer is signed using their private key, and the validity of any signature can be verified using the public key of the Peer.

Every peer has a list of their own history of transactions in the form of a collection of *blocks*. Every block is created and signed by a Peer, and includes the details of the transaction as well as a cryptographically secure hash of the previous block signed by the user. Importantly, the hash of a block uniquely identifies the block, as the trapdoor effect of cryptographically secure hashes ensures the infeasibility of finding another block with a given hash. The block thus uniquely references the previous transaction of the Peer. Since every transaction uniquely references the block before itself, the hash of any one block, recursively identifies *every* transaction made before by the Peer. This is as long as the Peer honestly references to their previous block. This referencing mechanism effectively links all blocks together in a gradually growing chain, thus making is a *blockchain*.

Every Peer in within TrustChain has their own chain, yet most transactions are *between* users. For this reason all transactions are made to happen in the chains of both users involved. In TrustChain,

this is achieved by having one of the two parties create a proposal block. In addition to the public key of the Peer, their previous hash, and the contents of the statement, the proposal also includes the public key of the counterparty. When the counterparty receives the proposal and agrees to the terms in the statement, they create an acceptance block. This acceptance block functions includes the public key of the counterparty, as well as the hash of their previous block, thus placing it in their blockchain. In addition the acceptance includes a reference to the proposal, thus linking them together. Both the proposal and acceptance blocks are then stored by both users, so they can both prove the transaction fully happened.

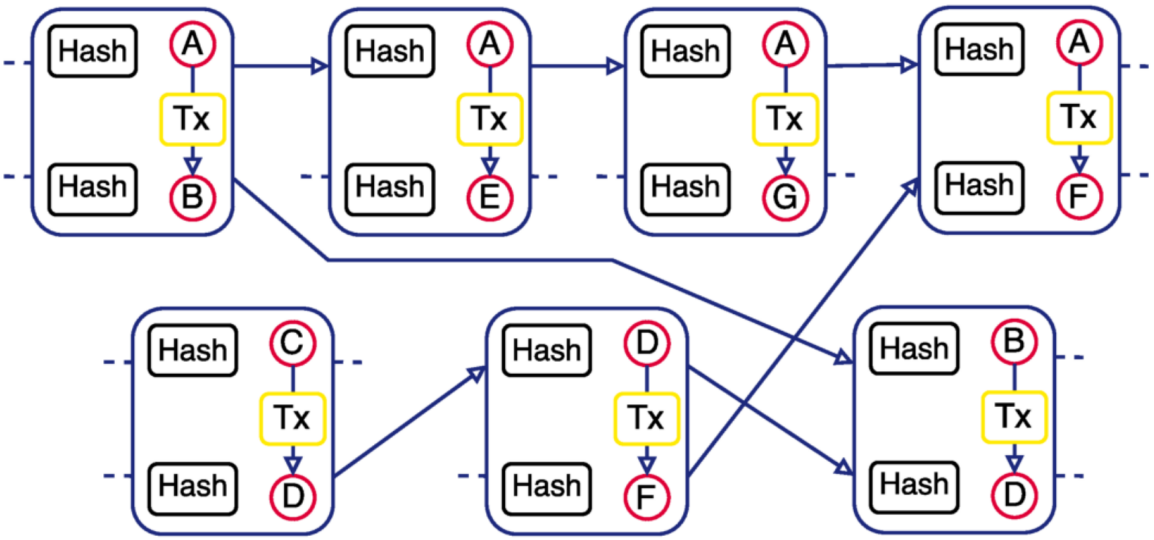



Figure 4.2: TrustChain block-lattice, interconnected personal blockchains[**?** ] .

### 4.2.2. EuroToken extension

The structure inherited from TrustChain serves us quite well as it comforms quite well to the block-lattice design we require. However, neither the python, nor the kotlin implementation includes any logic for running a currency. Before TrustChain can be used for EuroToken, it needs to be expanded to allow for value tracking and transfer.

TrustChain is quite open for expansion. It allows users to define their own block-types as well as validation logic for these blocks. TrustChain will make sure blocks are valid as a chain, by enforcing typical block invariants like hash correctness and signature validity. TrustChain makes use of IPv8 for its communication and exposes an API to create and sign blocks to other peers. TrustChain will then handle the process of sending the blocks over the IPv8 network.

In order to create the EuroToken logic we defined a number of TrustChain block types to achieve our goals. In order to conform to the scalability requirements all EuroToken proposal blocks by a user will include the balance of that user. This is part of the rolling-checkpoint mechanic that allows us to scale each users personal blockchain indefinitely without sacrificing scalability. The EuroToken block subtypes are as follows:

#### Transfer block

The transfer block is the core of how users interact. The proposal is created by the sender of a transaction and the acceptance by the receiving party. The block includes the amount to be sent as well as the balance of the sender at that point. The receiver will verify that the balances of the sender are valid before creating the acceptance. The receiver will then calculate the spendable balance all the way back to the last “full” checkpoint block in order to validate whether the balance of the sender is even spendable.

#### Checkpoint block

In order to be able to spend the balance a user has received they need to proof that a validator has taken notice of the blocks of the senders. The checkpoint block serves as this proof of validator. The proposal is created by the user and the acceptance is created by the validator. A checkpoint block is only considered “full” if the both the proposal and acceptance exist. If the acceptance does not exist, the block is meaningless and any validation will keep recursing the chain until a full checkpoint is found.

#### Creation block

The creation is a special type of transfer that is done by a trusted exchange. This block is the only way in which new EuroToken are allowed to enter the system and will only be considered valid if it is made by a trusted party. The proposal is made by the exchange and the acceptance is made by a user.

#### Destruction block

The destruction is the opposite of a creation. The proposal is made by a user and acts as a transfer to an exchange. The exchange then also creates an acceptance block. The creation and destruction blocks are used to convert between Euro and EuroTokens.

## 4.3. Exchange

For the EuroToken to be part of the Euro system a mechanism of exchange is required. The exchange forms the bridge between the digital EuroToken and the rest of the Euro systems. The exchange consists of

The gateway is implemented in python, and provides the user with 2 flows first is the I on the IPv8[**?** ] software stack.

The role of the exchange is to allow users to excha

### 4.3.1. Exchange flow

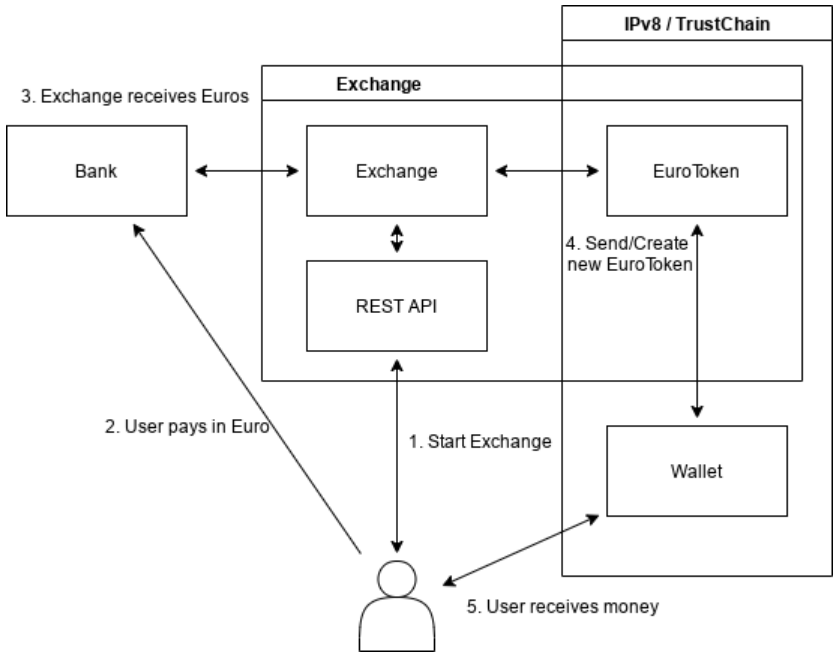



Figure 4.3: EuroToken Gateway Architecture

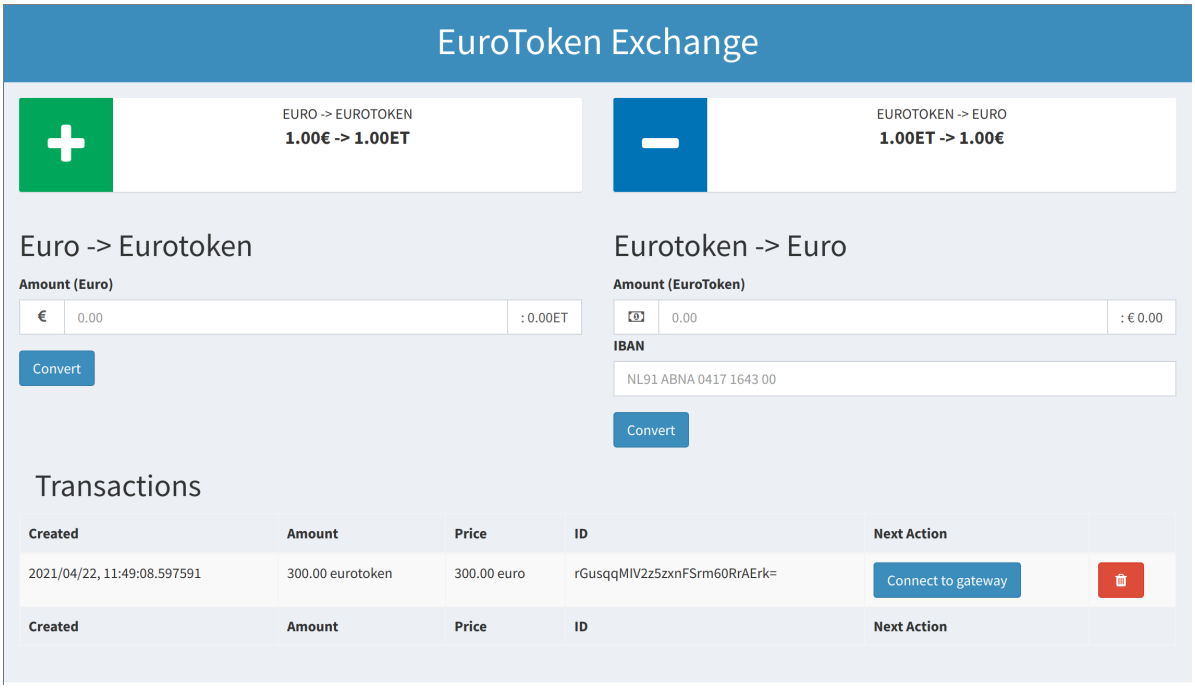



Figure 4.4: EuroToken Gateway Frontend

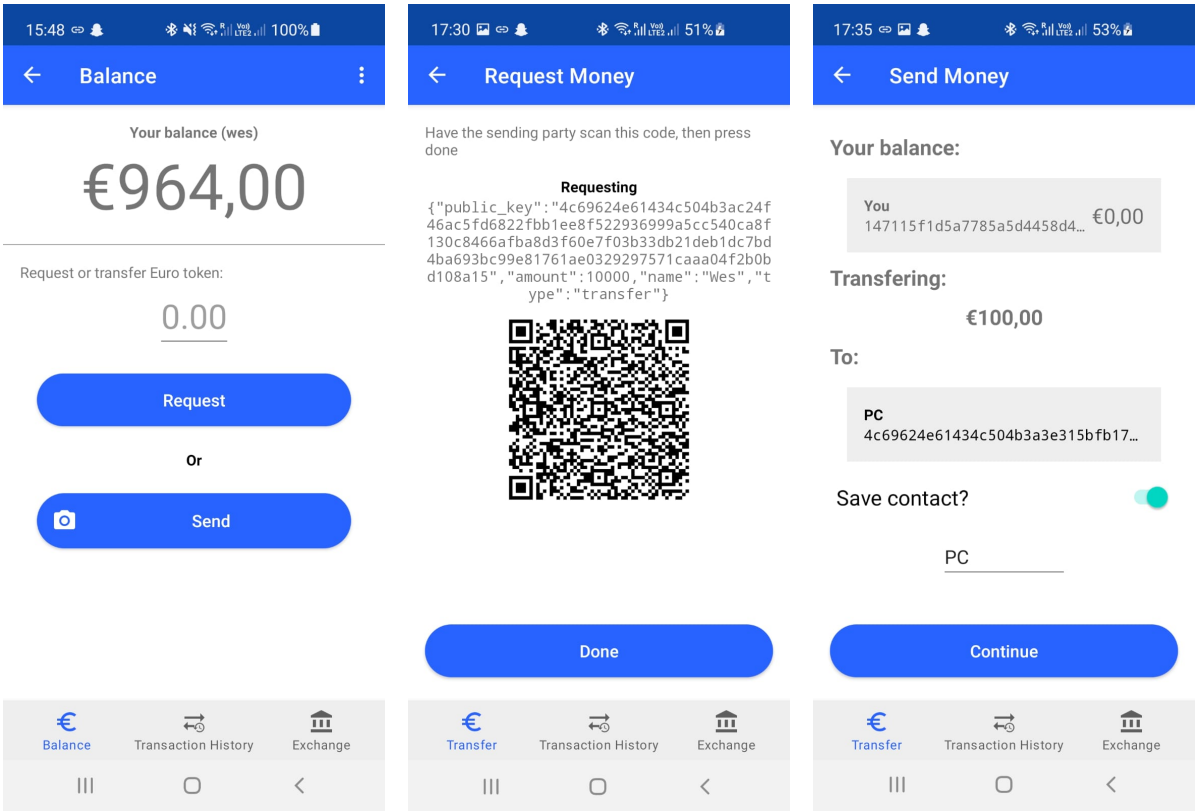



Figure 4.5: Wallet transfer

### 4.3.2. Frontend
### 4.3.3. Validation
## 4.4. Wallet
### 4.4.1. Peer to Peer transfer
### 4.4.2. Interaction with the exchange

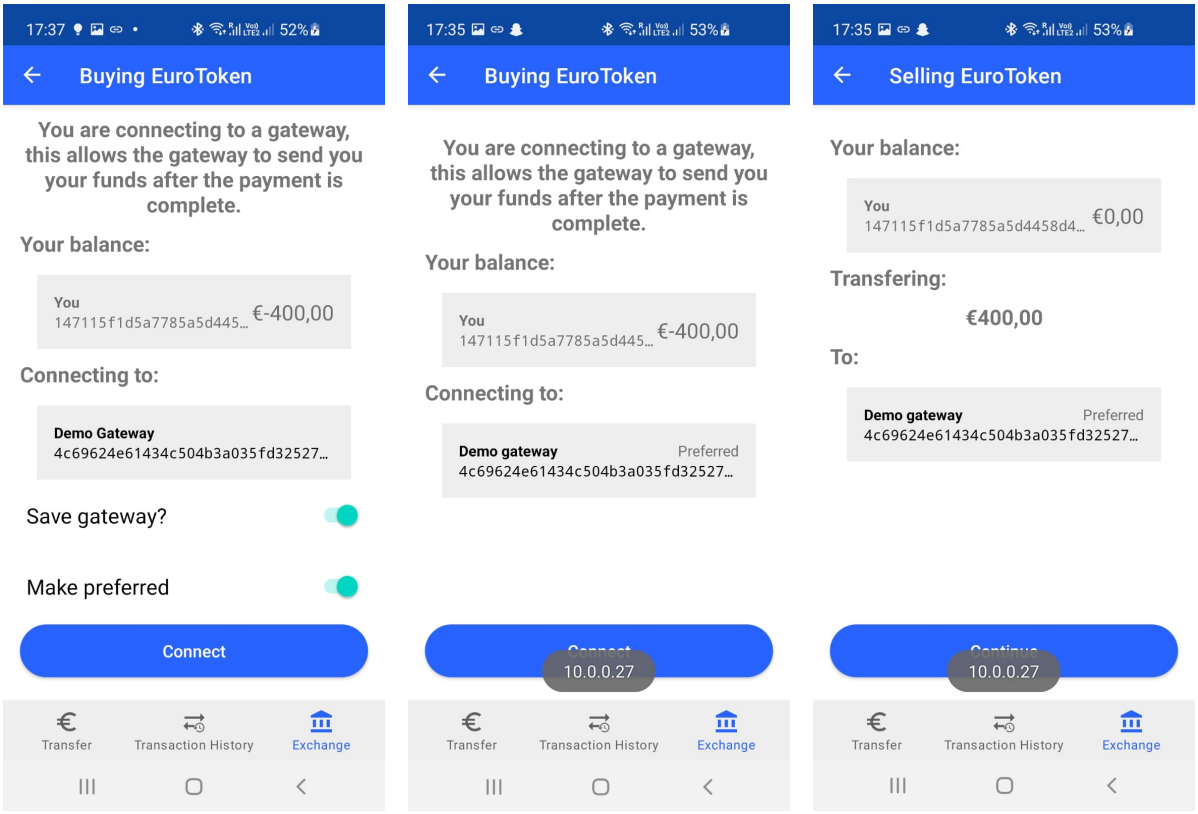



Figure 4.6: Wallet exchange

### 4.4.3. Programmable money benefits

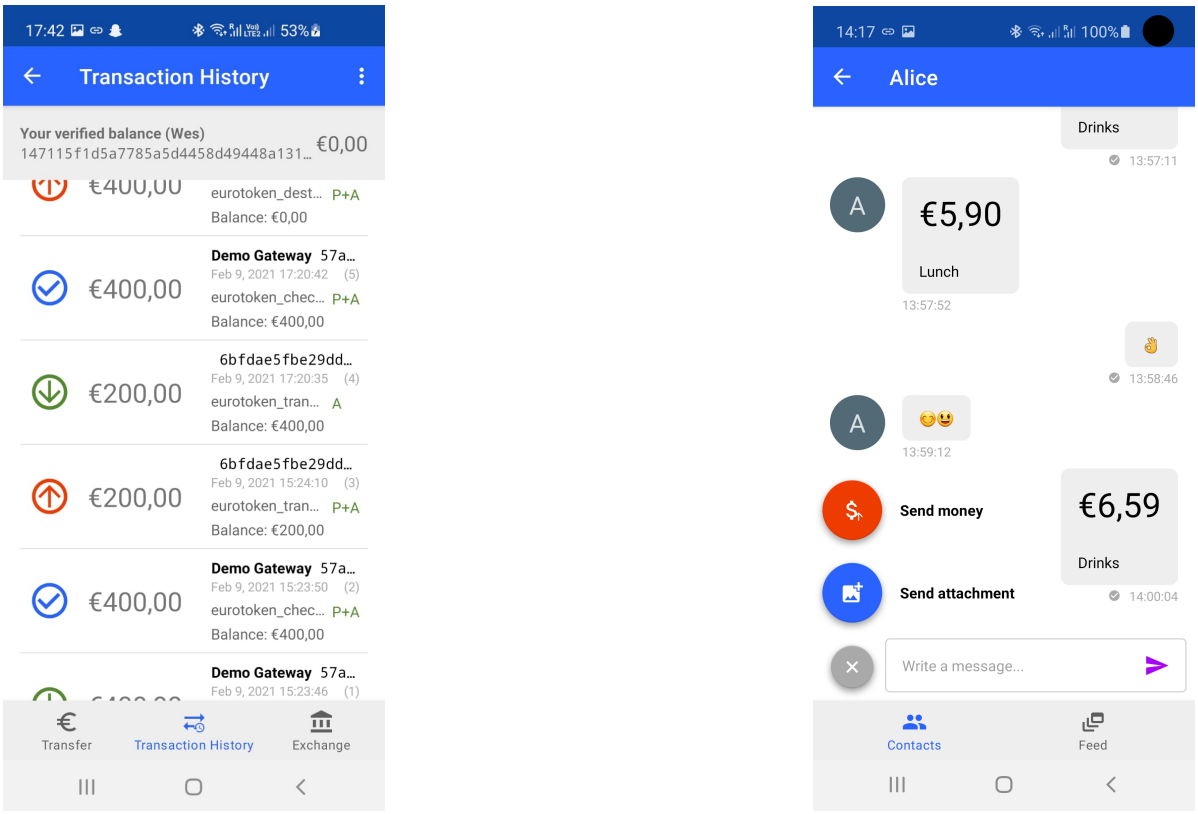



Figure 4.7: Wallet transactions

# 5

# Evaluation

In the problem description we specified the following requirements as derived from the ECBs report on a digital euro [**?** ].

1. **Enhanced digital efficiency**
2. **cash-like features**
3. **competitive features**
4. **monetary policy option**
5. **back-up system**
6. **international use**
7. Minimise ecological footprint (cost saving and environmentally friendly)
8. **ability to control the amount of digital euro in circulation.**
9. cooperation with market participants
10. compliance with the regulatory framework
11. safety and efficiency in the fulfilment of the Eurosystem's goals
12. easy accessibility throughout the euro area
13. conditional use by non-euro area residents

In this chapter we will evaluate our solution by these requirements. We emboldened the technical requirements as they will are guiding in our design, and we will go into more detail on how we met these requirements. The rest of the requirements will only be touched on lightly as they do not pertain to the topic of computer science and fall outside the area of expertise of the author.

We show how the EuroToken can be used to create: - a scalable CBDC - and provide all the benefits of programmable money - with the price stability of the euro.

## 5.1. Field trial

A field trail was conducted

Off-line transfer

Central to this project is the ability to transfer funds without an internet connection.

2 **cash-like features** 3 **competitive features** 5 **back-up system**

- Offline payments
- Peer to Peer
- Instant transfer
- No intermedeary to the initial transaction
- In the future the validator might be replaced by a more decentralised system
- Disaster mode
- "Once over" spending
- Could be expanded to include "emergency mode" where trust is increased and reprocessing is performed later to find instances of double-spending

Figure 5.1: Field trial

Figure 5.2: EuroToken off-line trial

## 5.2. Controlled experiments

2. Scalability Experiments In this chapter we evaluate to what degree the system conforms to the r 1 and 6 are met by the system.

1 **Enhanced digital efficiency** 6 **international use**
In order to evaluate whether the EuroToken system can be deployed at a global scale while also

- Scalability is very imporant
- TPS of the gateway
- graphs and tables
- Scaling limits and how to potentially mitigate
- Description of the benefit of edge computing
- Instant international transfer
- Increased efficiency in regulation due to full standardization
- The innovation boost
- programmable money
- smart contracts
- new forms of money streaming

## 5.3. Potential feature set

## 5.4. Real world viability

## 5.5. ECB requirements

4 **monetary policy option** 8 **ability to control the amount of digital euro in circulation.**

- Central bank controlled supply
- Option for “global inflation rate”
- More granular and “smart contract based” policy enactmet

## 5.6. Deployment consideration

ye

# 6

# Conclusion and future work

n

# Bibliography


[] Delft café premieres with eemcs blockchain euro. https://www.delta.tudelft.nl/article/delft-cafe-premieres-eemcs-blockchain-euro. (Accessed on 04/19/2021).

[] E-commerce statistics for individuals. https://ec.europa.eu/eurostat/statistics-explained/pdfscache/46776.pdf. (Accessed on 04/12/2021).

[] Statistics | eurostat. https://ec.europa.eu/eurostat/databrowser/view/ext_lt_maineu/default/table?lang=en, . (Accessed on 04/12/2021).

[] International trade in goods. https://ec.europa.eu/eurostat/statistics-explained/index.php/International_trade_in_goods#Strong_increase_in_trade_in_goods_with_China_in_2010-2020, . (Accessed on 04/12/2021).

[] Tangle_white_paper_v1.4.2.pdf. https://assets.ctfassets.net/r1dr6vzfxhev/4i3OM9JTleiE8M6Y04Ii28/d58bc5bb71cebe4adc18fadea1a79037/Tangle_White_Paper_v1.4.2.pdf. (Accessed on 04/12/2021).

[] Nano_whitepaper_en.pdf. https://content.nano.org/whitepaper/Nano_Whitepaper_en.pdf. (Accessed on 04/12/2021).

[] Tribler/py-ipv8: Python implementation of the ipv8 layer. https://github.com/Tribler/py-ipv8/. (Accessed on 03/23/2021).

[] Tribler/trustchain-superapp: Kotlin implementation of trustchain and ipv8 with rich networking: multihoming of local bluetooth+4g, decentral social networking, udp hole punching, etc. https://github.com/Tribler/trustchain-superapp. (Accessed on 03/23/2021).

[] Requiem for a bright idea. https://www.forbes.com/forbes/1999/1101/6411390a.html?sh=4e0608c6715f. (Accessed on 04/12/2021).

[] libp2p. https://libp2p.io/, . (Accessed on 04/26/2021).

[] libtorrent. https://www.libtorrent.org/, . (Accessed on 04/26/2021).

[] R.W. Blokzijl. rwblokzijl/stablecoin-exchange. https://github.com/rwblokzijl/stablecoin-exchange. (Accessed on 03/23/2021).

[] Stefan A Brands. An efficient off-line electronic cash system based on the representation problem, 1993.

[] Jetse Brouwer. Consensus-less security, 2020. URL http://resolver.tudelft.nl/uuid:d3d56dd8-60ee-47f7-b23a-cdc6c2650e14.

[] Vitalik Buterin. Ethereum: A next-generation smart contract and decentralized application platform. https://github.com/ethereum/wiki/wiki/White-Paper, 2013.

[] D. Chaum. David chaum on electronic commerce how much do you trust big brother? *IEEE Internet Computing*, 1(6):8–16, 1997. doi: 10.1109/MIC.1997.643931.

[] David Chaum. Blind signatures for untraceable payments. In *Advances in cryptology*, pages 199–203. Springer, 1983.

[] David Chaum, Christian Grothoff, and Thomas Moser. How to issue a central bank digital currency, 2021.

[] CoinDesk. Tether price | usdt price index and chart – coindesk 20. https://www.coindesk.com/price/tether, 03 2021. (Accessed on 03/19/2021).

[] Richard N. Cooper, Rudiger Dornbusch, and Robert E. Hall. The gold standard: Historical facts and future prospects. *Brookings Papers on Economic Activity*, 1982(1):1–56, 1982. ISSN 00072303, 15334465. URL http://www.jstor.org/stable/2534316.

[] Glyn Davies. *A history of money: from ancient times to the present day*. Cardiff: University of Wales Press, London, 2002.

[] Martijn de Vos and Johan Pouwelse. Real-time money routing by trusting strangers with your funds. https://repository.tudelft.nl/islandora/object/uuid:c51ac99d-3013-44b3-8ddd-fbd951a2454a, 2018.

[] Martijn de Vos, Can Umut Ileri, and Johan Pouwelse. Xchange: A blockchain-based mechanism for generic asset trading in resource-constrained environments, 2020.

[] TU Delft. Tribler/kotlin-ipv8: P2p communication library for android. https://github.com/Tribler/kotlin-ipv8. (Accessed on 04/26/2021).

[] Diem. White paper | diem association. https://www.diem.com/en-us/white-paper/, 04 2020. (Accessed on 03/19/2021).

[] ECB European Central Bank. Report on a digital euro. https://www.ecb.europa.eu/pub/pdf/other/Report_on_a_digital_euro~4d7268b458.en.pdf, 10 2020. (Accessed on 03/19/2021).

[] Forbes. Alibaba, tencent, five others to receive first chinese government cryptocurrency. https://www.forbes.com/sites/michaeldelcastillo/2019/08/27/alibaba-tencent-five-others-to-recieve-first-chinese-government-cryptocurrency/?sh=33d423fb1a51, 08 2019. (Accessed on 03/22/2021).

[] Rachid Guerraoui, Petr Kuznetsov, Matteo Monti, Matej Pavlovic, Dragos-Adrian Seredinschi, and Yann Vonlanthen. Scalable byzantine reliable broadcast (extended version). 2019. doi: 10.4230/LIPIcs.DISC.2019.22.

[] LibP2P. Peer identity :: libp2p documentation. https://docs.libp2p.io/concepts/peer-id/. (Accessed on 04/26/2021).

[] Tether International Limited. Tether whitepaper. https://tether.to/wp-content/uploads/2016/06/TetherWhitePaper.pdf, 06 2016. (Accessed on 03/19/2021).

[] Karl Menger. On the Origin of Money. *The Economic Journal*, 2(6):239–255, 06 1892. ISSN 0013-0133. doi: 10.2307/2956146. URL https://doi.org/10.2307/2956146.

[] Michael Ehrmann Miguel Ampudia. Financial inclusion: what's it worth? https://www.ecb.europa.eu/pub/pdf/scpwps/ecbwp1990.en.pdf, 01 2017. (Accessed on 03/23/2021).

[] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2009. URL http://www.bitcoin.org/bitcoin.pdf.

[] Reuters. China's $1.5 million digital currency giveaway impressed analysts. shoppers, not so much | reuters. https://www.reuters.com/article/china-currency-digital/chinas-1-5-mln-digital-currency-giveaway-impressed-analysts-shoppers-not-so-much-idUSL4N2H71NR?rpc=401&, 10 2020. (Accessed on 03/19/2021).

[] Matouš Skála. Technology stack for decentralized mobile services | tu delft repositories. http://resolver.tudelft.nl/uuid:bd3a5fbd-430b-4af6-bc33-eab436f4f7db, 08 2020. (Accessed on 03/23/2021).

[] The Maker Team. The maker protocol white paper | feb 2020. https://makerdao.com/en/whitepaper/, 02 2020. (Accessed on 03/19/2021).

[] Fed The Federal Reserve. Preconditions for a general-purpose central bank digital currency. https://www.federalreserve.gov/econres/notes/feds-notes/preconditions-for-a-general-purpose-central-bank-digital-currency-20210224.htm, 02 2021. (Accessed on 03/19/2021).

[] ECB Statistical Data Warehouse. Share of card payments in number of total payment transactions. https://sdw.ecb.europa.eu/quickview.do?SERIES_KEY=169.PSS.A.U2.F000.I1A.Z00Z.NP.X0.20.Z0Z.Z. (Accessed on 03/23/2021).