How the King returns: A digital future for cash

Eduard de Jong *

14th April 2024

(version 2.RC1)
Pre-publication—for review

Abstract

This paper introduces a practical implementation of electronic cash (e-cash) at scale. The key feature presented is a secure value transfer protocol using Aggregating Receipt Tokens (**ART**). E-cash is a form of money with all the features of traditional cash including payer privacy, immediate settlement and no fees. Payments can be made face to face or over a distance using any form of communication, from QR codes and NFC to HTTPS. Received e-cash is stored locally for future spending as digital information in a secure device under the exclusive control of its owner. The devices work together as a large-scale distributed IT system that provides society with a highly scalable, very resilient infrastructure for secure payments between citizens, corporations, banks and government agencies. The implementation of this infrastructure consists of two prime components i) an electronic purse (e-purse) as a *digital bearer payment instrument* to store, pay and receive e-cash and ii) an issuer as the provider of e-cash currency as a type of money integrated in a monetary system and the guarantor of its value and security. A continental-scale e-cash system that realises an offline, cash-like, digital economy, fully integrated in the existing monetary infrastructure, can be realised with the IT architecture presented here.

Keywords: cash, digital money, offline payment, e-cash, transferable e-cash, electronic cash, offline currency.

1 Introduction

In English there is a saying, "*Cash is King*". Cash puts a buyer in the strongest position. Any purchase can be completed immediately and without question. There are no delays, dependencies, or intermediaries. Accepting cash is an easy decision for the seller.

*The author would like to thank Evert Fekkes, Simon Youel, Marc Dencker, Ram Banerjee and Maxim Schulze for their insightful comments on drafts of this document. A special thank you goes to Peter Cattaneo for the many encouraging discussions throughout its long gestation.

Over the past three decades in many parts of the world remote electronically recorded transactions have steadily replaced the use of coins and banknotes as the way to pay.[1] At present in many countries the role of cash in society is in retreat.

In the Arthurian legend, without the king, Arthur, on the throne, dark forces oppressed the people and clouded the future. In analogy, in the digital realm of our present times King cash is found absent too. His absence excludes parts of our society from the larger financial system.

When King Arthur returned to his lands and pulled his sword from the stone, he ushered in a new era of peace and prosperity, much as cash did for commerce scores of centuries ago. While the existing payment products are suitable in some situations, the lack of a digital cash option limits commerce in multiple ways.

Figure 1: King Arthur pulls his sword Excalibur from the stone.
Source: Nilfanion—Wikimedia UK, CC BY-SA 4.0

The return of Cash, the King, in digital form will bridge the digital divide in society and spread prosperity anew.

A digital technology for payments that are offline.
A technology to securely store digital currency detached from a centralised account.
A technology that can be adopted by many people for a range of payments, large and small.
Just like cash.
That's how King Cash can return.

This paper presents a comprehensive system architecture to implement transferable electronic cash (e-cash) with the aggregating receipt token (**ART**) technology introduced in [3]. It is organised as follows: The next section presents the monetary context of a digital form of cash and introduces the key properties of an e-cash payment as a new, digital, type of money. Section 3 presents an overview of the system architecture for implementing an e-cash system at a national scale, highlighting the central roles of the issuer and the e-purse, the secure container of digital currency, as the sole system component that handles money. Section 4 zooms in on the functional architecture of the e-purse. Section 5 describes the IT architecture for e-cash handling operations in financial institutions to support withdrawing and depositing of e-cash. Section 6 presents the functions performed by the issuer's IT system to support the correct and safe operation of the whole e-cash money system. A conclusion and bibliography complete the paper.

The IT system architecture in this paper shows a clear way for King Cash to actually pull his metaphorical sword from the stone where it is presently stuck. With this implementation the King can return to its place on the throne, bringing back money as a widely accessible public good to benefit all, as digital cash, as e-cash.

[1] The extent to which cash is being replaced as a means of payment differs greatly over the world and within countries. Several countries and segments of the population within countries continue to rely on physical cash. E-cash is an instrument to make digital payments accessible to these citizens and small business.

	Cash	Bank Reserves	Commercial Bank Money
Physical	Notes and Coins	Paper Ledgers Legacy/Obsolete	Paper Ledgers Legacy/Obsolete
Digital	?	Digital Ledgers	Digital Ledgers

Figure 2: Hole in a grid of types of money and implementation technology.
Source: This table is taken from [4]

2 Money, cash, e-cash and aggregating receipt tokens

The current monetary system consists of three types of money: i) Cash, consisting of banknotes and coin held by their respective owners; ii) Bank reserves, consisting of accounts recorded and updated in a ledger by the central bank and iii) Commercial bank money, consisting of accounts for individual users recorded and updated by ledgers operated by financial institutions. Over the last seventy odd years the digitisation of the latter two types of money has been realised.

Figure 2 shows that the third type of money, cash, with its foundation in the handling of physical object like coins and banknotes, has not been digitised, except for automating tasks like counting, distributing and bulk handling during physical distribution.

Conventional digital payment methods use the two digitised form of money in fig. 2. These methods are based on gathering and storing identity information for all users and updating digital ledgers maintained by commercial banks and by the central bank to record a payment transaction. They lack essential properties of cash:

- Cash is accessible to all;
- Cash payment is exclusively between two parties;
- Cash payments are anonymous and final there and then;
- Cash does not require fees to finalise a value transfer.

E-cash is money that can be used in any digital environment, both offline and online. A payment in e-cash does not require a digital communication network, it does not need an identity register for authentication and authorisation, and it does not need a transaction database to finalise the value transfer. E-cash complements existing digital payments, adding strong privacy protection and strengthening the resilience of the monetary system.

This paper shows how *electronic cash* (e-cash) can fill the hole in fig. 2.

2.1 E-cash

Like with cash, a payment in e-cash can be from person to person, from consumer to a merchant, from merchant to supplier, form business to business, from citizens and corporations to government and from governments to all. E-cash payments are **A2A**, *anyone-to-anyone*.

E-cash users of can enjoy features enabled by its digital nature. An e-cash payment is i) effortless for any amount by its electronic computations; ii) fast by digital communication and iii) face-to-face or over a distance, e.g. online.

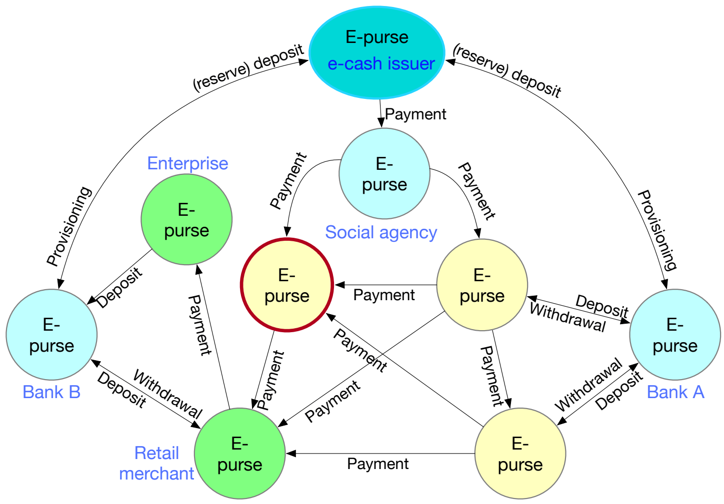

Figure 3: An e-cash system supporting offline payments of any amount for any purpose.
Source: Based on [3, figure 2]

Figure 3 schematically shows a snapshot of a deployment of a comprehensive e-cash system with a selection of its users. It shows different users with their different e-purses used to store e-cash as digital information. An "e-purse," for *electronic purse*, is a personal digital payment instrument; it receives and sends e-cash in payments. The arrows in the figure show e-cash flowing from one e-purse to another.

Private users have e-purses shown in yellow providing a basic level of security. Businesses, like a retail merchant or its suppliers use the e-purses shown in green with a higher security level protecting higher amount. The business e-purses can be used for paying suppliers and service provider from sales receipts before making a deposit in a bank account. A merchant may also withdraw e-cash to pay a supplier of, for instance, as needed to supports its customers with a Plain Old Cash to e-cash conversion service.

A *bank* participates in the e-cash infrastructure as they do for the Plain Old Cash, without the need for big vaults and armoured delivery vans. A bank e-purse, shown in blue, keeps its e-cash stock needed to service all its customers for each day. As digital money it will be much easier for a bank to maintain its liquidity as that can be done with digital payment by the central bank.

No bank account is needed to use e-cash as is shown by the e-purse with the solid (red) outline, where e-cash is received from a social agency and from other users. An unbanked user can fully participate in an e-cash economy with digital payments of any amount to merchants, both local and online.

Social agencies store the e-cash they need to make their support payments in an e-purse with a protection similar to banks. These agencies could have direct access to the issuer's e-purse as they make sure that every citizen has access to e-cash as public money. The e-purse for an unbanked user is just never used to do any deposits or withdrawals with a bank. This the only difference from what is shown for the other yellow e-purses.

The *Issuer* of the e-cash, in many countries the central bank, also operates an e-purse similar in security features as other banks. This e-purse supports one special operation: to create additionL e-cash. Issuing e-cash and distributing it to its users is the task of the *Issuer*, while it requires the same monetary responsibility, the digital nature of the currency makes this task a lot simpler.

The e-cash issuer guarantees that the value received by the payee is genuine and hence can be spent again at a later time by that user. The issuer can provide this guarantee with information technology security tools: a correct cryptographic money-transfer protocol, the managed use of cryptographic keys, cryptographically signed credentials and secure hardware. Through the use of these tools, the issuer plays an essential role in each payment transaction; like the digital information in the transfer token this role of the issuer is largely invisible.

With e-cash stored in a secure device under control by its owner and the secure device available to every citizen, resident and enterprise it is money that can be widely available. E-cash is money that can freely spend and received without interference or constraints.

With no other parties involved in an e-cash payment there are no per-transaction costs, and no fees payable.

E-cash can't be stolen[2], the size to store it does not grow proportional to the amount.[3] An e-cash payment involves data that can act as secure proof of payment, which can be entered directly as binary data in a digital system as reference in a transaction record, both on the payer and payee side.

With these additional properties e-cash is, in many circumstances, actually better than Plain Old Cash.

2.2 Implementing e-cash

The monetary value in e-cash is represented as digital information, either stored in the memory of a secure computer, ready to be spent, or it is in transit, as a payment, encoded in a specially constructed secure message.

Each of the payments, withdrawals, deposits or e-cash provisioning in fig. 3 involves only two e-purses as shown in fig. 4. The figure shows the two users in an e-cash payment: The payee to provide the information needed to securely receive the payment and the payee to instruct the e-purse to send the payment using the payee information. The e-cash to be received as payment is a single cryptographically secured message computed by the payer e-purse.

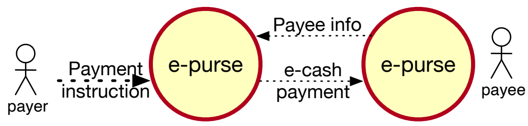

Figure 4: An e-cash payment involves just two parties, each equipped with a dedicated digital device, their e-purses.
Source: [5]

An e-purse is the ensemble of software and computer hardware used to store and process the digital information that is e-cash. The hardware includes a dedicated secure computing device and one or more computers where the user, owner of the money in the e-purse, can generate the payment instruction shown in fig. 4. The secure hardware protects the stored money, the e-cash payment protocol and the cryptographic keys securing each payment.

With offline spendable funds in the e-purse controlled directly by the payer no further data is needed to make a payment. The privacy of the payer is fully protected by the offline nature of the payment.

[2]The e-purse is locked by one- or multifactor authentications, e.g. PIN, fingerprint, for payments over a certain, configurable amounts and stealing an e-purse gives very little gain to the thief, if any.

[3]For example a digital value representation with 64 bits a particular tamper resistant encoding suitable for an e-purse, 10^{12} units of monetary value, e.g. up to 9.999.999.999,99, can be stored.

Diagram 1: E-cash value transfer

$$\left\{\frac{\text{payer user interface}}{\text{amount \& approval \& payee info}}\right\} \xrightarrow[\textit{payee info\&amount}]{\textit{payment instruction}}$$

$$\left\{\frac{\text{payer e-purse}}{\text{decrement balance}}\right\} \xrightarrow{\textit{transfer token}} \left\{\frac{\text{payee e-purse}}{\text{increment balance}}\right\}$$

Diagram 1 is an implementation view of the e-cash payment shown in fig. 4. It shows that a payment in e-cash only involves two parties, a *payer* and a *payee*. The payer uses the user interface to its e-purse to review the amount of the payment as matching the prior agreement reached with the payee. This agreement may be reached some time before continuing with the actual payment. Approving the payment triggers sending the payment instruction the payer e-purse, where its value transfer software computes the *transfer token* and sends it to the payee e-purse.

Each party uses its own e-purse to store their e-cash and to execute programs to send and receive, respectively, the e-cash transfer token.[4]

The functions implemented in the e-cash system can be distinguished as pertaining to the money domain or the security domain.

The money domain lets users experience e-cash payments as very similar to Plain Old Cash: i) it involves only two parties; ii) it protects the privacy of the users; iii) the transfer of the agreed amount of e-cash is finalised there and then; and iv) received money can be stored and spend.

The security domain makes sure that: i) no money gets lost, stolen or created; ii) the system is kept secure with monitoring and the ability to respond to breaches; iii) cryptographic techniques are used to keep the payer privacy protected; and iv) the system cannot be abused for money laundering and tax evasion. In the design and implementation of an e-cash system the functions in these domains are interwoven with the security functions marginally noticeable. Just like they are in Plain Old Cash.

A challenge in implementing e-cash is convincing its users to trust it as money and to maintain that trust. The basis for trust in e-cash is provided by **IT** security techniques like encryption and special hardware that shields secret data and protects against tampering with computations and stored data. Maintaining this trust requires that a large majority of users are convinced by the security properties of the system. Skilled technical professionals in particular will require transparency of many implementation details, which could be provided with open-source software and hardware designs. A formal proof of correctness of the software deployed in e-purses is possible; it will be essential to convince critical IT specialist that e-cash money is safe. Rumours of security issues may arise from time to time, particularly during

[4]The ways a transfer token is constructed, how cryptography is applied to secure its content, the e-cash money, how the changes in the amount of e-cash stored by payer and payee are synchronised and how the system and its security can be managed is specified by a particular e-cash technology suite. Over the last 30 years a number of different e-cash systems have been proposed some of which have been deployed. Unlike the current proposal, none of these previous technologies are suitable for large scale deployment as a form of generally available cash, c.f. [2]

the first few years, which could undermine the public's trust. The ability to detect and respond to security issues is a requirement for the implementation of e-cash.

The **ART** technology meets the implementation challenges in both the money and the security domains.

2.3 A secure, digital, offline currency with ART technology

An aggregating receipt token[5] (**ART**) is a key technical element to implement security and efficient operations in an e-cash system.

In the money domain it fullfils all properties mentioned above for e-cash payments as a privacy-protecting, two-party immediately finalised transaction. Additionally an aggregating receipt token enables i) the offline finalisation of a payment by a computation in the payer e-purse; ii) accepting a payment in software only by the use of public key cryptography; iii) a high transaction capacity for online payments without requiring access to a secure device; iv) giving the payee a secure digital statement of a cash receipt and v) providing the payer with a cryptographic proof of payment.

Complementing the features in the payment domain the security domain of the e-cash system with **ART** technology provides

- a capability-based security mechanism with expiring credentials that prevent unconstrained use without affecting usability;
- the secure transport of e-cash money to a specific payee without any loss of value or double spending;
- peer-to-peer system security confirmation;
- cost-effective risk management through the issuance and re-issuance of configurable credentials, i.e. aggregating receipt tokens and key certificates;
- monitoring the security of all past transactions to detect potential breaches;
- protection of user privacy with payer anonymity and payee pseudonymity allowing full, detailed AML enforcement.

The **ART** technology payment and security features show that the four design trade-off described in the high-level design guide for offline payments in the fourth BIS Polaris report[2, Table 3] are not applicable.

The aggregating receipt token credential is at the center of securing e-cash, it is a personal, cryptographic container that securely transports monetary information from the payer to the payee as the last step in each payment. An **ART** is personal as it contains a unique number that matches a number in the secure hardware; it is cryptographic as its content is secured by a digital signature. The e-cash transported with an **ART** can only be spend with the owner's secure hardware.

Cost-effective payent security with aggregating receipt tokens

The aggregating receipt tokentechnology meets the money and security requirements for to an e-cash implementation with i) dedicated hardware with scalable security to store the e-cash balance and protect secret payment keys;

[5] The **ART** is a security token specific to e-cash as presented here. It should not be confused with tokens discussed in the context of blockchains or distributed ledgers (**DLT**), which are completely differen types of tokens.

ii) an asynchronous, atomic, idempotent, cryptographically-secured digital value transfer protocol; iii) providing the payer with a stream of anonymous payment credentials certifying secret payment keys; iv) providing the payee with a stream of pseudonymous reusable money acceptance credentials, the *aggregating receipt tokens* and v) analysis of acceptance credentials returned after they expire to detect anomalies.

The e-purse hardware that securely stores the digital money implicitly provides the payer secure control of spending the e-cash it contains: Possession of the device is a precondition for spending, just like with Plain Old Cash. The software in an e-purse is run-time configurable by the issuer via the credentials it receives: after expiration of an old one, getting a fresh credential can update the e-purse configuration. The payment credentials provided to a user are configured matching the security properties of its e-purse.

The **ART** credential can be used for multiple payments until it expires. Expiration is configured in the **ART** as a period of use, a number of payments and a maximum aggregated amount.[6]

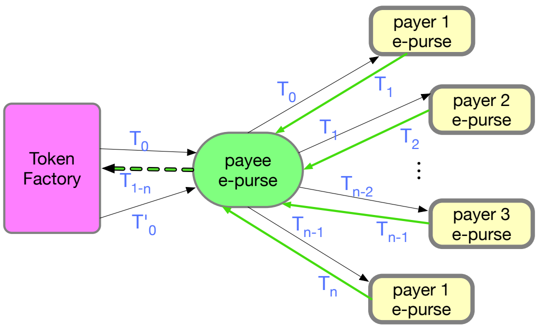

Figure 5: The aggregating receipt token is issued to a specific payee and used to secure multiple payments, aggregating amounts paid.
Source: adapted from [3, figure 1]

An aggregating receipt token is a mechanism to provide security it is not itself money, it has no value at issuance. An **ART** does contain a currency indicator so it can only transport e-cash in one specific currency. It also contains a unique identifier for the owner of the money; any value it transports in a payment is bound to the e-purse operated by the owner.

In fig. 5 the payee obtains a new **ART** T_0 from the issuer's *token factory*. Token T_0 is used in a first payment request to payer 1; as a freshly obtained token its value is 0. The payment with T_0 is returned as T_1 shown as a thick green arrow. The further green arrows in fig. 5 show the transport of paid e-cash with each subsequent payment with the **ART** obtained in the previous one. All tokens used as payee information in a payment contain the configuration from token T_0 which will be honored by the paying e-purse. and the details of the previous anonymous payment.

The payee e-purse adds the received money to its spendable balance. It can do this after each payment or at any later time.

After receiving a final payment T_n, the payee submits the token to the issuer in a request for a fresh one (T'_0). The token, indicated by the dashed line, sent to the issuer contains details of all the anonymous payments received with it. The token sent to the issuer does not contain money.

[6] An **ART** is reusable; every time it is used it becomes a unique new token that contains the most recent payment and a digital signature by the most recent paying e-purse, while retaining the configuration data in the original **ART**. Figure 1 in [3, p. 14] shows how a series of transfer tokens has been created with a single aggregating receipt token tokens.

Performing an **ART** payment with an e-purse is configurable[7] during operations by the issuer through parameters embedded in the credentials used by payer and payee to send and receive money, respectively.

Local payer authorisation validated by its own e-purse results in an identity-free payment, which means that identity based fraud and its associated costs will be largely non-existent.

The issuing of reusable credentials does not require a high-availability, high-capacity IT infrastructure with low-latency database operations. Maintaining the credential-issuing infrastructure will be considerably less costly than a online-payment infrastructure with the same capacity.

An offline ART payment

Diagram 2 is the aggregating receipt token version of the generic implementation of an offline payment in diagram 1. The payment operation consists of an asynchronous exchange of just two digital messages between the two e-purses involved:

1. A request message from the payee to provide an **ART**, which is one of the credentials provided earlier by the issuer to a particular payee, also specifying an amount to pay and;
2. A response message with a transfer token based on the **ART** received in the request containing the amount of payment and secured by a fresh digital signature computed with a secret cryptographic payment key issued to the payer and certified by the issuer.

Diagram 2: E-cash payment with aggregating receipt token

$$\left\{\frac{\text{Payee e-purse}}{\text{selects } \mathbf{ART}}\right\} \xrightarrow[(a,)\ \mathbf{ART}_{\text{Payee e-purse}}]{\textit{request}}$$

$$\left\{\frac{\text{Payer UI}}{\text{approves amount } a}\right\} \xrightarrow[\text{approval},\ a,\ \mathbf{ART}_{\text{Payee e-purse}}]{\textit{payment instruction}}$$

$$\left\{\frac{\text{Payer e-purse}}{\texttt{atxr}(\text{approval},\ a,\ \mathbf{ART}_{\text{Payee e-purse}})}\right\} \xrightarrow[\mathbf{ART}^{+a}_{\text{Payee e-purse}}]{\textit{transfer token}}$$

$$\left\{\frac{\text{Payee e-purse}}{\texttt{accept}(\mathbf{ART})}\right\} \xrightarrow[\mathbf{ART}]{\textit{at redemption time}}$$

$$\left\{\frac{\text{Payee e-purse}}{\texttt{redeem}(\mathbf{ART})}\right\}$$

In the request message in diagram 2 the payee sends one of the stored **ART**s that is configured for the expected payment amount. The payee marks the stored **ART** as '*in use*' so it can't be used in another payment request until it has been returned by the payer as the transfer token.

The amount to pay in the request can be provided by the payee or by the payer depending on the reason of the payment. In either case the payer

[7] Configuring offline digital payments is described in more detail in [4]. It presents the outline of fine grained run-time adjustable configurations to differentiate between various payment use cases based on their different risk profiles.

approves the payment and a payment instruction is sent to the payer e-purse with an approval token, the amount and the **ART** received from the payee resulting in increased value for the **ART** to be sent to the payee.

In this diagram $\texttt{atxr}_{Payer}(a,\ \mathbf{ART})$ is the programmed computation executed inside the payer's secure device in its e-purse. This computation *completely* implement the payment operation: It subtracts the payment amount a from the payer balance and adds that amount to the value of the aggregating receipt token received from the payee. As the payee's personal digital e-cash-transport container incrementing its value and then cryptographically sealing it immediately transfers ownership of the payment amount to the payee. This computation, fully done in the payer device, makes an **ART** payment atomic.

The cryptographic key used in the `atxr` computation to seal the new token value is certified by the issuer with an anonymous certificate. The anonymous key certificate is one of the payment credentials stored in the e-purse; it guarantees payer privacy.

The payer e-purse stores the computed transfer token in its *persistent memory*[8] before sending it to the payee, this makes an **ART** payment idempotent, which means that in the exceptional case that a payment result has been lost in communication, repeating the payment request will result in exactly the same result: the receipt token that did not get received. When receiving a payment request the payer e-purse first checks its persistent memory for an earlier payment request with that same token. If an earlier request is found, the **ART** computed and stored as transfer token for that request is sent as the result of the request, which, after receipt by the payee, completes the interrupted payment. Otherwise, the computation `atxr` is done and its freshly computed and stored result is sent to the payee. The idempotency of an **ART** payment enables local recovery from a communication failure, only involving the payer and the payee.

The payee accepts a payment by validating the transfer token received from the payer's e-purse. The payment is valid if the transfer token is an **ART** computed from the one sent to the payer in the request, has the expected new value and has a correctly computed digital signature. The received token can be stored in the e-purse memory replacing the reusable part of the **ART** that had been used in the request. The updated **ART** is ready for reuse in a further payment request.

At any time after accepting payment, the amount of e-cash received with the **ART** can be made available for immediate further spending. The computation in diagram 2 referred to as $\texttt{redeem}(\mathbf{ART})$ is an e-purse operation[9] to transfer the money aggregated in the token to its balance. This operation only accepts tokens that are owned by the owner of the e-purse and is idempotent so received money can only be added once.

Public key cryptography is used to compute the digital payment signature on the token. Validating a digital signature requires a public key and therefor acceptance of a payment does not require the use of the hardware security

[8] Persistent memory that is also protected against tampering can be realised with secured computing hardware such as a smartcard.

[9] Like the payment computation $\texttt{atxr}_{Payer}(a,\ \mathbf{ART})$, the $\texttt{redeem}(\mathbf{ART})$ computation is done inside the secure hardware device. A redeemed **ART** is recorded in the secure persistent memory to make the computation idempotent.

component of the e-purse. Section 4.1 describes the payment, acceptance and redemption operations of the e-purse in more detail.

A payment with **ART** technology is asynchronous as the **ART** needed in the payment can be send at any time before the payment. The **ART** that results from a payment can be retained by the payer and used for a repeat payment to the same payee with the same or another amount. An example of synchronous **ART** payments is presented in section 5, where it is used for regular deposits of possibly different amounts of e-cash in a bank account.

In the remainder of this paper a reference to e-cash or an e-purse is to one implemented with **ART** technology.

3 E-cash system architecture

Figure 6 shows the top-level architecture[10] of the **IT** system implementing e-cash based on the **ART** technology. There are just two principal components:

- The e-purse, the hardware and associated software provided to each users to store monetary value as e-cash, to implement the payment operation as a secure message protocol, to provide a user interface and to interact with the user's IT infrastructure;
- The issuer, an IT system with a specially configured e-purse to implement issuance that is further dedicated to maintaining the security of system operations.

The issuer issues e-cash, assures that e-purses are available to users and manages the operations and security of the system. While there can be hundreds of millions of e-purses, operated by individual residents, government agencies and enterprises small and large, only a single instance of the issuer component is required.

The e-purse is the only component in the e-cash system that is involved in handling money, it is the functional part specifically equipped with hardware security to securely make and receive payments, to store money and to control its spending. The hardware security in an e-purse can be designed in distinct classes of strength to reflect the different amounts of money its owner needs to store and to pay. Complementing the classification of hardware security payment keys and algorithms can be configured for exclusive use to secure large payment amounts only for keys stored in devices with stronger security.

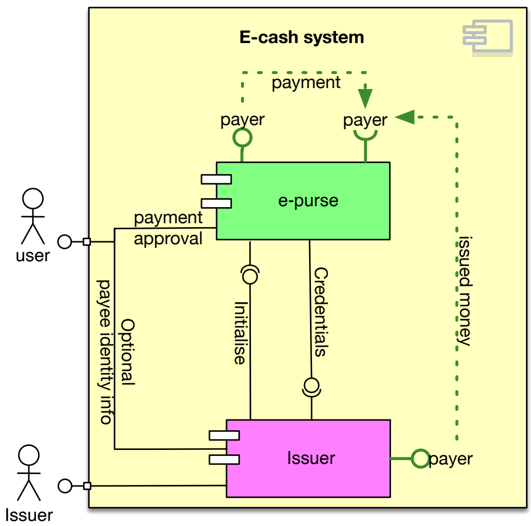

Figure 6: The two functional components in e-cash money.

[10]The IT architecture figures in this paper use the UML notation for functional components in an IT system. The diagrams show a component composed of sub components that provide services to each other and other, outside components.

Interacting between themselves in payments, the e-purses form a network of peer-to-peer computers with digital information being stored and processed locally within each computing node. In each payment the two computing nodes involved, payer and payee e-purses secure the money as it flows from user to user.

E-cash is a true *distributed-finance* system providing the resilience, the high transaction capacity and the security by peer validation that are inherent to distributed peer-to-peer computing. At the same time, the use of issued payment credentials in each transaction provides the issuer with complete visibility and control of the money flow between all these *technical* peers.

Figure 6 shows the payment operation between two e-purses as a thick, dashed, green line between the two *payer* functional interfaces of the e-purse component. In a payment two instance of the e-cash components are involved, one as payer and the other as payee. The user, owner of the e-cash in an e-purse, controls the spending of money via user interface functions provided by the e-purse.

Another dashed, green line in fig. 6 shows how the issuer supplies liquidity to the system as an e-cash payment to an e-purse. E-purses used for liquidity are implemented with hardware protection fitting the larger amounts stored and paid, which are typically operated by a bank or other issuer approved institutions like a post office. Details of the IT architecture for financial institutions to handle e-cash in support of liquidity provisioning and interface to bank accounts are found in section 5.

Initial liquidity is provided when a user obtains an e-purse that has been preloaded with a set amount and pays that amount to obtain its new e-purse, either in cash, bank payment or as special social security support payment. The liquidity in this case is provided by the issuer via a payment to an e-purse incorporated in an '*e-purse provisioning station*' as presented in section 3.2. The e-purse provisioning station is also involved in the optional registration of user data.

Details of the IT architecture for the e-purse and its secure hardware component are found in section 4.

3.1 Issuer operations

The tasks of the issuer in the e-cash system is to

- keep it operating smoothly, with sufficient liquidity, and ensuring monetary stability;
- make sure the money and payments are secure;
- fulfil its social objectives of accessibility, availability and usability, ensuring e-cash is well integrated in the monetary system next to cash and commercial bank money; and
- prevent abuse of the invisible nature of digital information.

E-cash security is based on i) strong cryptography, with secret keys to digitally sign data in each payment; ii) compact, portable, secure computing hardware, in the form of an e-purse, operated and physically controlled by the user who is the owner of the digital money stored within; iii) a value transfer protocol secured against tampering, loss and double spending; iv) regularly providing users with fresh credentials enabling them to make and receive payments; and v) the observation of system operations, including the velocity of money, in order to adjust system operational parameters when needed in response.

Figure 7 shows the high-level IT architecture with three functional components to realise the issuer tasks.

- System agent components to provide users with the digital credentials needed to make and receive payments, the latter of which are collected to obtain a data stream detailing the many distributed operations in the system;
- System agent components that integrate an e-purse as a sub component for paying or receiving e-cash to or from users, e.g. as withdrawals or deposits;
- the *issuer core* to control and observe system security and operations through operational configuration of the system agent components.

The two functional components, indicated with a reference to *"system agent"* in fig. 7, support e-cash users in their regular use and implement all interactions between the issuer and the e-purse shown in fig. 6. These user-focused issuer tasks can be delegated to other parties, some of which may already be involved in the financial system; independent entities like banks, social services or a postal service could operate these functional components under supervision of the issuer. Financial institutions can provide system agent component services as enhancements to other online services offered to their—e-cash-using—customers. A system agent could implement both these operational components.

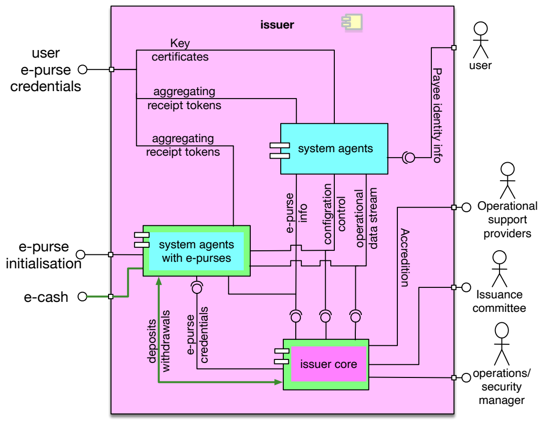

Figure 7: Functional components of the issuer

System agent components with an e-purse provide liquidity to the e-cash users. In particular they can provide users with a securely initialised e-purse preloaded with e-cash for the amount at which it is 'sold' to the user as a bearer payment instrument. Section 3.2 presents details for the operations of the two types of system agent components involved with e-cash distribution and e-purse provisioning.

The system agent components, in processing requests from a user e-purse to refresh the credentials that are needed to make and receive payments, obtain credential configuration data from the *issuer core* to incorporate in the credential provided to the user in response. Extracted from aggregating receipt tokens returned in a request for fresh ones, the agents also provide the issuer with a stream of operational data for system operations and security analysis.

A user's e-purse will when the need arises to refresh payment credential call online services provided by System agents. These intermittent credential updates can be scheduled at regular intervals or automatically initiated where and when internet connectivity is available.

The functional operations implemented in each of the two system agent components can be implemented with a growing number of separate identical servers matching an increasing size of e-purse deployment. Section 3.3 describes how the payment supporting system agent components can be implemented as loosely coupled distinct functional sub components.

The *core issuer* functions are to:

- Provide the root of trust and the public key infrastructure used by the agents;
- Manage the amount of e-cash in circulation and create additional liquidity when needed;
- Analyse the operational data stream extracted from returned **ART**s;
- Set the configuration for **ART** credentials to be applied by agents;
- Provide payment credentials for use in the e-purses operated by agents;

Operations by the issuer are controlled by two kinds of human operators either to make decisions on the volume of e-cash in circulation or to interpret operational statistics and adjust policies that will be affected through parameters in **ART** credentials.

The issuer also maintains a register of accredited providers of agent services that operate an e-purse to support provisioning **ART** credentials to those system agents. Section 6 presents the IT architecture for the issuer's core operations.

3.2 Agent components with an e-purse

Two types of agent components use an e-purse in the services they provide e-cash users, each implementing a different functional component in the overall e-cash system.

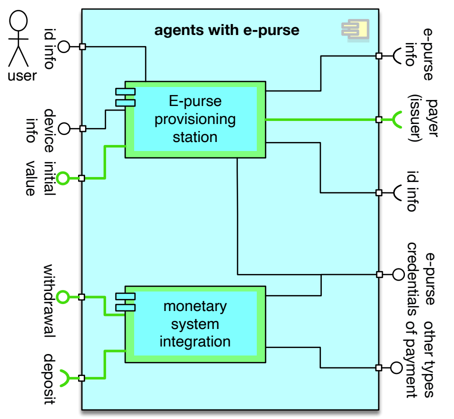

Figure 8: Issuer agents with an e-purse.

One type of system agent component, the *e-purse provisioning station*, is dedicated to the final interaction with an e-purse before it is handed over to a user in order to initialise it for use by this specific user as owner of the money it contains at that time and in the future. During initialisation the e-purse is loaded with its initial value in e-cash from a dedicated e-purse operated by the system agent.

Provisioning an e-purse to its user can be done[11] by banks or social services for their clients or by post offices for the general public, in particular for users without a bank account.

In the process of provisioning a user with an e-purse its persistent unique identifier is obtained from the system agent component that manages *user pseudonyms* (c.f. fig. 9). For a payer, who as

[11]The provisioning scenario described above is a face-to-face process for issuing the secure hardware component of an e-purse (the MonEbox, c.f. section 4.2) to a user and in the process also performing the secure binding to e-purse software installed in a first user device, e.g. a mobile phone. The provisioning of an e-purse by a bank to its customers could also be done online in stages initiated by the customer. A partial initialised device can then be sent to the user who completes the initialisation online secured by the device. Partial initialisation instals user related data including withdrawal and deposit tokens as described in section 4.2. for their e-cash using customers.

payee wants to be anonymous, the system agent component managing identity information will generate a random e-purse identifier. Alternatively, a payer can opt to provide an identity information system agent component with the personal details legally required for AML enforcement, in which case this information is stored and the unique e-purse identifier is generated as reference to this information, e.g. as a cryptographic hash. In both cases the e-purse identifier completely shields the user identity from any other user, the system agents and the issuer.[12]

The *monetary system integration* component is operated by i) banks, providing deposit and withdrawal services for their account holding customers; ii) social services, post offices, local government agencies etc. aimed to support users without a bank account; iii) operators of **ATMs** that accept an e-cash payment to provide bank notes and iv) merchants that operate automated cash handling devices to receive Plain Old Cash for their products and services configured to provide banknotes for e-cash. Together these system agents fully integrate e-cash into the existing monetary infrastructure.

Section 5 gives details of the IT architecture for the e-cash-specific functional components to extend the account-managing IT system for financial institutions. It shows a seamless integration of deposits and withdrawals enabled by **ART** technology by including account info and, for a withdrawal, user authentication as payment details in the aggregating receipt tokens used to implement these operations. The account referenced in a deposit or withdrawal **ART** can be one holding **CBDC**.

3.3 Non-financial agent components

To make and receive payments an e-cash user requires regular updates of credentials, it needs both certificates for cryptographic payment keys that have expired and aggregating receipt tokens, respectively. Figure 9 shows the two system agent components to support the e-cash payment infrastructure with providing these credentials. Two other system agent components in the figure, the *identity register* and *user pseudonym*, securely store and process the optionally provided user information. Figure 9 clearly shows that an **ART** e-cash system only uses identity information in its support of an e-cash payee, while doing so indirectly as a pseudonym.

The *user pseudonym* component provides a pseudonymous user identifier to be installed in an e-purse when it is issued to a user. An

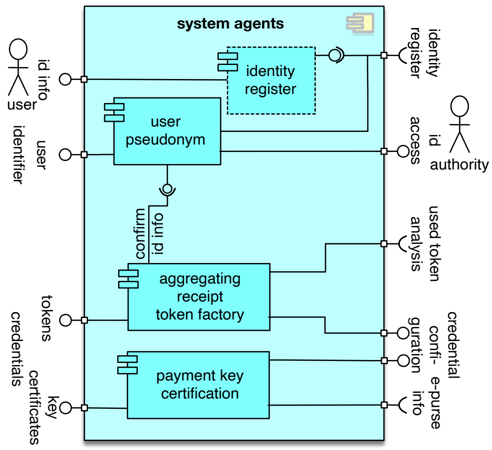

Figure 9: system agents supporting user credential provisioning.

[12]In **ART** technology the identity of a user as a payer is unconditionally protected. The identity of an anonymous payee is also unconditionally protected. For an anonymous payee generic constraints on payments will be needed to meet AML requirements, implemented in configuration of **ART**s issued to the user that is recognised as anonymous by properties of the anonymous purse identifier. An anonymous payee could request special tokens configured to receive a higher amount from specific class of payers, e.g. an employer paying wages.

e-purse provisioning station (cf. fig. 8) requests this identifier with user information obtained from the user. User information provided in this request will be stored in the *identity register* component. This component may not need to actually store all user information, instead it could reference data already reliably stored elsewhere, e.g. by a bank for its customers, in which case this identity information reference is also provided to this component.[13]

Storing full user information, or a reference to it, supports the eventual identifying of suspected violators of AML regulations. The *user pseudonym* component strongly protects the payee identity. Any user data stored can only be accessed by an authority with a specific legal reason to unseal a specific identifier.

The unique e-purse user identifier will be embedded in all the aggregating receipt tokens for this e-purse, it will bind all monies received in e-cash payments to its unique owner. As a service to the payer some not-unique payee information like a first name, the kind of merchant and neighbourhood will additionally be included in the payee's **ART**s.[14]

The *aggregating receipt token factory* component creates new **ART**s needed by an e-purse. It creates these tokens upon request. An **ART** request contains the pseudonymous owner identifier, an expiration date and any applicable user selectable payment configuration, like the maximum amount of each payment that can be received with the token and the total number of payments.[15]

Typically, the e-purse generates requests for new **ART**s as existing ones near expiration. An **ART** request contains the pseudonymous identifier and the *aggregating receipt token factory* uses the *user pseudonym* component to determine that it refers to a user that owns and operates the requesting e-purse. Configuration data to include in a new token is regularly updated by the issuer. An **ART** request usually contains expired tokens e.g. to be exchanged for fresh ones; these tokens will be forwarded to the issuer for analysis of its usage.

A freshly created **ART** has a value of zero; it is signed by a dedicated cryptographic key and the signature is added to the token. The digital signature cryptographically binds the token value to the e-purse of its owner and to its operational parameters. The **ART** factory key is certified by the issuer's root of trust.

The *payment key certification* component creates certificates for the secret cryptographic keys used in an e-purse to secure a payment. Configuration data in a payment key certificate specifies a maximum payment value, expiration, type of payment, frequency of use and other constraints.[16] The e-purse

[13] In this case user authorisation will be needed to access the remote identity register to obtain data needed to generate the identifier, e.g. as used for logging in to an online bank service.

[14] The short payee information in an **ART** is a service for the payer, during payment it gives assurance that the intended party will receive the money and when reviewing payments made it acts as a reminder. While the short payee information has been verified to match the full identity information, actual payment security does not depend on this user service: An attacker might succeed in inserting its own **ART**in a payent request but his identity can be revealed from the identifier in the token.

[15] A user typically has multiple aggregating receipt tokens configured for different types of payment and different amounts.

[16] Constraints on the use of a particular key protects the key; they do not in anyway constrain the use of the e-cash in an e-purse.

sends a request to this agent for freshly generated keys to replace keys that are about the expire.

The key certification component queries the *issuer core* component with the unique identifier for the secure e-purse to determine it is genuine. As result of that query the agent learns about the e-purse hardware security (cf. section 4.2) properties to determine appropriate payment parameters, e.g. the maximum amount of a payment to be signed by the key. The digital signature that certifies a new key is blinded in order to make the payment anonymous. The system agent's certification key is certified by the issuer's root of trust.

In e-cash based on aggregating receipt token technology there is a clear functional separation between the payment system and managing identity information. A payer credential, a key certificate, does not need any user information. A payee credential, an **ART**, contains either a pseudonymous identifier or a fully anonymous one. Both credentials provide strong protection of user privacy.

The services provided by these system agents are provided on request without a strict requirement for a timely response. As a consequence the supporting infrastructure for e-cash provided by these agents is highly scalable. Agent functions can be implemented by existing stakeholders in the payment system e.g. banks and payment service providers.

4 The e-purse

The e-purse is the one functional component in an e-cash system that actually handles money, storing it, receiving it and spending it. The e-purse implements the **ART**-based secure, atomic, anonymous, idempotent payment protocol. In addition to interacting with other e-purses to make and receive payments, the e-purse interacts with the system components presented in section 3.3 to keep the system operating securely.

Together with these supporting system components e-purses build a large network of computers intermittently communicating with each other to realise digital money with strong privacy protection, that can be used anytime and anywhere. With the e-purse digital money can be experienced as Plain Old Cash.

A large majority of e-purses would be used by citizens, as individuals or as families; many e-purses would be owned by merchants and other small and medium sized businesses. Corporations of any size that sell products online operate e-purses to directly receive payments in e-cash over the internet. Banks, post offices, and social services use their e-purse to play a role

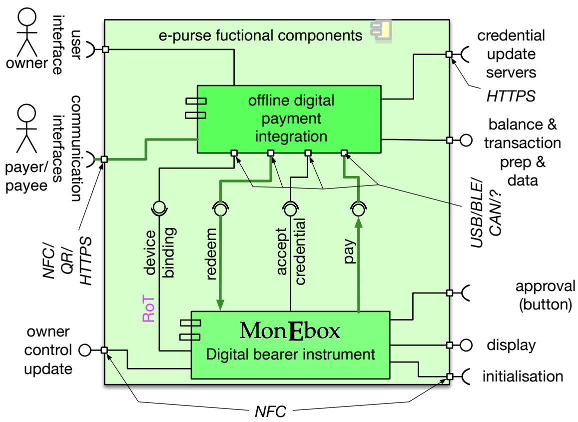

Figure 10: Functional components of the e-purse

in distributing e-cash (cf. section 3.2). The e-purse enables these different users to securely store, and pay and receive e-cash.

The e-purse functions are to:

1. initiate a future payment by sending an **ART** in a request to a payer;
2. validate the **ART** received in a payment request;
3. approve payment of a specific amount to a specific payee;
4. update the request **ART** with the amount of a payment and communicate this result to the payee;
5. validate the **ART** received from a payer as the expected payment;
6. make the amount received in payment available for spending;
7. present the user the balance of funds available and an overview of past transactions and
8. interact with system agent components to obtain payer and payee credentials.

Figure 10 shows the two functional components of an e-purse: i) a software module, the '*offline digital payment integration*' component, installed on a host computer operated by the e-cash owning user; ii) a dedicated tamper--detecting, secure device as the foundation for security of the money. The Mon**E**box is the owner's personal digital bearer payment instrument.

The '*offline digital payment integration*' component acts as the communication bridge to e-purses of other e-cash users to send or received digital money. It provides the user interface for the digital bearer payment instrument.

The two components of the e-purse communicate with each other over a local connection like Bluetooth Low Energy (BLE), USB, CAN or any other internal communication method available in the IT system in which the e-purse is integrated. The local connection implements a security *air gap* separating the digital bearer payment instrument from the internet.

The *offline digital payment integration* software has four main functions:

- as a payee, initiate and receive payments and to make the funds received in payments available for further spending with the Mon**E**box's *redemption* function;
- as a payer, approve a request for payment, obtain the payment result from the Mon**E**box and make it available to the payee;
- manage the cryptographic certificates used by the system to secure the credentials issued to the user, validating new credentials with a root of trust (**RoT**) obtained from the Mon**E**box;
- to establish and maintain the secure binding with the Mon**E**box and to funnel the communication with its owner, other, outside, e-cash users and the credential provoding system agents to the three functions above.

The *e-purse integration* component software can be installed on a mobile phone, tablet, desktop computer, cloud application container or local administrative IT system, e.g. a 'backoffice computer'. The host for this software gives the e-purse access to its user interface functions and to its communication links to to send messages to other devices in the system. Operating details for this module are presented below in section 4.1.

The Mon**E**box holds a balance of spendable e-cash, the secret cryptographic keys to make a valid payment, the executable code that implements the **ART** payment protocol and the executable code to refresh credentials. It implements the following security providing functions of the e-purse: i) validate the approval for a payment conforming to user specified limits; ii) compute, if the balance is sufficient, a transfer token as response to an approved payment request; iii) in case of multiple users with spending approval rights, securely manage these rights; iv) get initialised with a unique device identifier and associated cryptographic keys; v) get initialised with an amount of e-cash; vi) get initialised with an identifier for the user; vii) prepare the secure requests to system agent components to refresh credentials; viii) to protect, before being handed to its user, its provisioning supply chain with secure control of the lifecycle of its existence as a tested, secure, operational IT component; ix) detect physical abuse in order to respond to attacks by disabling its operations while saving the amount stored for later recovery; and x) enable recovery of funds once disabled.

The two e-purse components provide security to payments in a layered fashion, the MonEbox to securely store the money and compute the cryptographic security that protects an e-cash payment and the offline digital payment integration module to locally establish the user as the controlling owner of the money. A third layer of security is provided by the issuer in the form of the two types of payment credentials required to make and receive payments, respectively.

Section 4.2 presents details of the IT architecture for the MonEbox and its security features. Based on the generic e-purse architecture in fig. 10 section 5 presents the integration of the e-purse in a banking IT system to support e-cash deposits and withdrawals.

4.1 E-purse offline digital payment integration

The *offline digital payment integration* module acts as the switch board between a local user, which is the owner of the money, remote e-cash users to make or receive payments, the owner's IT infrastructure with accounting services and web access, and the MonEbox as the secure digital bearer payment instrument that stores spendable money. For an individual user this IT infrastructure can consist of a mobile phone, tablet or desktop computer. In a corporate use case the IT infrastructure includes a financial administration program to digitally record payments and receipts in e-cash.

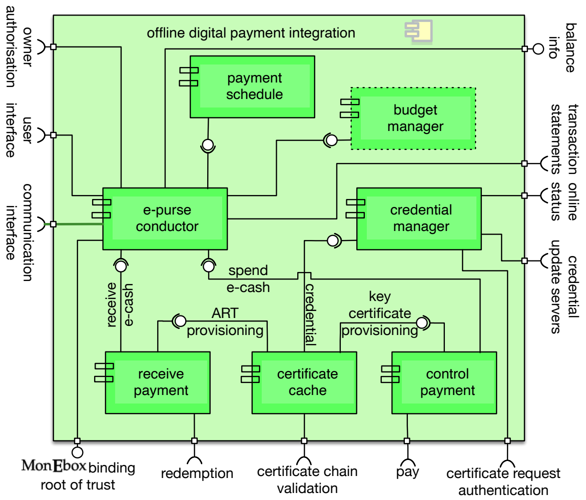

Figure 11: Functional components for e-cash IT integration

Figure 11 shows the functional components implementing the digital payment integration module. The components shown implement the functions to i) establish a unique secure connection with a MonEbox; ii) prepare and send a payment request; iii) receive and accept a requested payment; iv) prepare making a payment in response to a payment request and v) proactively manage the required payment credentials.

After installing the software module in a host computer it is configured to use a specific MonEbox. The software initialises itself by establishing a long term communication channel with the configured MonEbox. In that initialisation a cryptographic key exchange protocol between the MonEbox and the software module is done to establish shared cryptographic keys to protect future data exchanges over the communication channel.

Initialisation of the e-purse software component installs the cryptographic root of trust for validating both payments and fresh credentials, (**RoT** in fig. 10). The root of trust represents the issuer of the monetary value stored as e-cash in the MonEbox.

The *conductor* component implements the configuration with a specific MonEbox, the binding with it and installing the payment root of trust in the host system. It connects with the user interface functions available in the host to support the user in paying and getting paid, delegating the actual sending and receiving money to the other dedicated functional components. It also connects with the communication interfaces available to communicate with payers and payees.

To adapt to the options available in the host to interact with its user the *conductor* is typically realised with host system specific program code. Where available in the host, the conductor supports configuring biometric authorisation for some of the higher value payments.

The **ART** protocol uses two messages to realise a payment, which can be send to payer or payee over any channel available in the host: as **QR** codes, using a camara and screen; or over **NFC**; or as internet communication via the web, e.g. as part of a **URL**; or as a dedicated service message, send to a WEB API; or as data embedded in a PDF of an invoice. It also supports combining different communication modes for the incoming and outgoing message.

The **ART** payment is asynchronous and a payment request can be sent at any time prior to the moment of payment. The *conductor* component forwards such early requests to the *payment schedule* component. With an early payment request only containing an **ART** the payer can make a series of payment of different amounts as agreed with the payee before receiving the **ART**.

The *conductor* component also provides two functional interfaces to the user's IT infrastructure: One interface supports the presentation of the current balance of spendable funds and the reviewing of payments made and received by the e-purse. The second interface supports the entering of e-cash payments in a financial administration package.

An **ART** can be directly used as the digital justification in the transaction record by either party. The aggregating receipt token recorded with an e-cash transaction can be validated at any later time by an auditor. The auditor sees the **ART** as an immutable digital statement of the transfer of ownership of an amount of monetary value in e-cash at a particular date and time. The digital signature in the **ART** with the payer's cryptographic payment key, endorsed by the issuer through a certificate chain, is a proof for the auditor that it represents a genuine e-cash transaction.

By verifying the digital signature in the **ART** with the installed issuer root-of-trust the *receive payments* component can fully accept a payment. Accepting a payment with a software module means it does not require interaction with the MonEbox. The money received with an **ART** will be transfered to the MonEbox in a separate *redemption* operation, which can be done immediately or at a specific time, e.g. every hour, or before a payment is to be made, or when the owner explicitly instructs it.

With the software only acceptance of a payment the *receive payment* component can be deployed on another device operated by the payee. A payee device other than the host where e-cash payments can be accepted must be able to communicate with the *conductor* component in the host to obtain **ART**s to be used in payments.

In fact, multiple e-cash accepting devices can be connected to the same e-purse host. An example of such a replicated deployment of payment acceptance software is a shop with different points of sale (**POS**) each accepting e-cash. The **POS** devices in this example intermittently communicate with their e-purse *conductor*, which could be hosted on a back-office computer, to refresh exhausted or expired **ART**s and redeem the moneys received.

The main tasks of the *control payment* component is to

1. determine that the same **ART** has not already been processed by the MonEbox in order to implement payment idempotency
2. verify the authorisation of the owner for a payment;
3. a check is made of the availability of sufficient e-cash funds, and, if needed, to redeem any unredeemed moneys to increase the spendable amount in the MonEbox;
4. do a pre-flight check of the payment operation by validating the data to be sent to the MonEbox.

Validity of the token data is determined by using the installed root of trust to validate both the digital signature on the token configuration parameters and the signature on the transaction most recently received with it.

Determining the validity of the token received from a payee in addition to checking digital signatures also checks that the last transaction in the token conforms to the token configuration. This check replicates the acceptance computation done by the payee after it received the token in that payment. An invalid token will be rejected by the MonEbox, as it also performs this check as a pre-condition to computing the transfer token. This peer validation of the token is one of the mechanisms that maintains system integrity.

The owner's authorisation of a payment can be established by a PIN or password or with a biometric check or a combination of these. If available on the host the reference value for PIN or password can be stored in secure memory, which is typically provided by personal devices like smartphones. In all cases payment authorisation is established locally by the software in the payer's device.

The payment authorisation is represented by a unique cryptogram generated by the *control payment* component with the protection keys exchanged at MonEbox-binding time. This unique authorisation cryptogram is verified by the MonEbox as the first step in processing the payment. In an IT context with multiple configured payment-approving users, each user's authorisation cryptogram is distinct, in this case the MonEbox authorisation verification includes a check of user specific spending limits.

After the pre-flight check the payment request is sent to the MonEbox to obtain a transfer token as the payment result. To implement payment idempotency, the token obtained from the MonEbox is first stored before it is forwarded to the payee via the *conductor*. Idempotency is also implemented in the MonEbox; storing the payment result by the *control payment* component in preparation for its future pre-flight computations.

The *credential manager* is tasked with making sure that credentials are replaced when they expire. Credentials are provisioned with overlapping expiry periods so they are always available. This task does not require action from the user and can be performed whenever the host is connected to the internet where it connects to a system agent component. Each credential is a message digitally signed with a cryptographic public key.

A certificate chain, with the root-of-trust as an endpoint, is used to certify the public key used by the system agent in generating the credential. To speed up the processing of incoming and outgoing payments the validity of certificates in a chain is cached; certificate chain data may be proactively included in responses from system agent components for recently issued certificates. The certificate cache is shared with the MonEbox to reduce data communication and computing demands in the actual payment operations.

The *credential manager* component prepares a request message to refresh a credential; the request is cryptographically sealed by the MonEbox in order for the system agent to determine that the request comes from a genuine e-purse. A credential request does not include personal information other than the previously established payee pseudonym in a request for an **ART**.

The *offline digital payment integration* module can include further functional components. Figure 11 shows two of them:

- a *payment schedule* to specify and pre-authorise e-cash payments to be made at specific date and time, and
- a *budget manager* to give the user the discretion to conceptually partition the available e-cash balance to be spent on specific purposes.

The *payment schedule* component supports recurring payments, like for rent and energy, or for bills for which payment is due by a specific date. A scheduled payment can be pre-authorised, in which case the e-purse can be configured to warn the user when an intervening payment that lead to an insufficient balance. When not authorised yet, the schedule component will prompt the user for it close to the scheduled time.

The *budget manager* aims to support individual users with a low income to plan for essential expenses in a fashion that is similar to budgeting Plain Old Cash with jars or cans[17] marked for a future spending purpose, which could be set up in the *payment schedule*. When a budget has been set, this component affects the payment approval user interface by adding an option to select one of the budgeting 'jars' to take the money from in making the payment. The importance of budgeting with digital money for social justice and financial inclusion is discussed by Ignacio Mas[6].

The *budget manager* functional component is shown with a dashed outline as an indication that it may be selectively enabled. Other optional functional components can also be selectively enabled in the e-purse software, e.g. to access a bank account for deposit or withdrawal (c.f. section 5).

For the same e-purse, i.e. connected to the same MonEbox, *offline digital payment integration* software can be installed on different trusted computers,

[17] The user interface can be designed to make the experience of allocation digital cash for budgetting much like dropping coins in jars.

which gives the user multiple options to use e-cash to partake in digital payments. Each installed instance of the e-cash software component is securely bound to the same associated MonEbox, which is authorised by the owner at the time of binding. Each binding between a MonEbox and an instance of the *offline digital payment integration* software is unique. The MonEbox typically limits the number of software modules it can be bound to.

The offline digital payment integration module is software. It can be installed in multiple instances on the same host, where supported, with each instance bound to a different MonEbox.

4.2 The MonEbox

The MonEbox is a secure device built to protect itself against attempts to exfiltrate secret keys and tampering with stored data, i.e. its balance, or the results of computations, i.e. an **ART**, when making a payment. Upon detecting tampering the secret payment keys are disabled and the balance is securely saved.[18]

Figure 12 shows the functional structure of a MonEbox: i) a *security core* that implements the cryptographic security operations, storing secret keys, the balance of e-cash, and the program code that implements the payment functions and ii) a *communication and detection* processor that implements the communication with one or more of the offline digital payment integration modules bound to it, handles the display and button, monitors tamper detection sensors, responds to tampering and manages the battery.

The *security core* can be one of the widely available, runtime-configurable, high security smartcard chips. A runtime configurable security core e.g. based on Java Card™ technology, supports implementing the different security functions as one or more applets, which can be updated separately with the latest software when the MonEbox is issued to its user. For security reasons updating configurable data in the security core can only be done through its NFC interface. This could also be done, for instance, by the user with a dedicated one-time use mobile app. The NFC interface can also be used for low value payments, which can then be made irrespective of the battery state. The NFC interface also loads the battery that powers the MonEbox components.

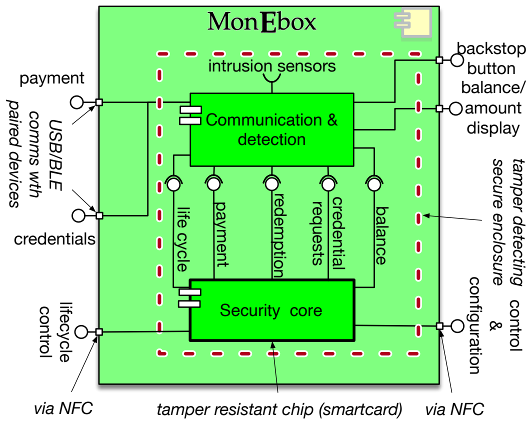

Figure 12: The MonEbox: the implementation of a digital bearer payment instrument.

To its user a MonEbox is a digital bearer payment instrument, an instrument that supports making cash-like payments digitally. To guarantee security of the digital payment it has been manufactured in a secure fashion

[18] A recovery system agent component could be implemented to provide a recovery service for these saved e-cash funds.

and provisioned to the user under control of the issuer, very similar to how banknotes are currently produced and distributed.

As a bearer payment instrument, albeit a digital one the MonEbox has a value (its balance) and it can support a display to show this value. It supports a button to push as a physical means of approving a payment. As the *offline digital payment integration* module normally handles payment approval, the MonEbox button acts as configured backstop approval for extraordinary payments.

At issuance a MonEbox is preconfigured for the minimum amount that needs approval with a press of its button. This limit constrains the damage that might be done by rogue software that could corrupt the host of the offline digital payment integration module. The user can set a lower approval limit for this button. The presence of this button is a deterrence against attacking a host in order to steal e-cash.

The MonEbox can be built in different forms and with scalable physical strength against tampering. A basic appearance of a MonEbox for use by individuals can be as a key fob, or another small portable device, with an exterior design that makes it clearly recognisable as the bearer payment instrument for a particular currency, e.g. with a currency symbol, text and a banknote-like texture. For commercial users a version of the MonEbox can be implemented with sufficient strength to protect a higher value stored as e-cash.

The MonEbox is a keystone in implementing security for the e-cash system. It is complemented by the two types of cryptographic credentials and the secure implementation of credential provisioning system agents. The foundation for e-cash security is provided by the issuer in observing system operations and adjusting the operational configuration for credential provisioning in response to observed system behaviour.

5 E-cash and financial institutions

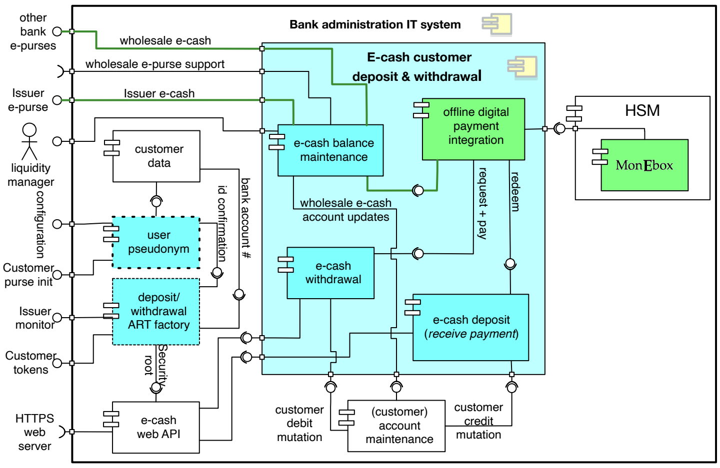

Figure 13: IT architecture for financial institution to support customers using e-cash.

Banks play the same role in e-cash as in Plain Old Cash, except that all financial operations and most customer interactions are digital. Deposits and withdrawals are payments to and by the bank, respectively, using aggregating receipt tokens that are additionally configured with a bank account number.

Figure 13 shows the functional component integrated in a bank's IT infrastructure to implement the withdrawal and deposit of e-cash. It shows how the bank's e-purse interfaces with the bank's existing account maintenance function to credit and debit user accounts with amounts corresponding with amounts received or payed out in e-cash. The MonEbox in the bank's e-purse is implemented in a commonly used banking Hardware Security Module (**HSM**) communicating with the *offline digital payment integration* component that is at the heart of the *e-cash customer deposit & withdrawal* component.

Figure 13 also shows the three main existing functional components in the bank's IT system with which the *e-cash deposit & withdrawal component* interacts. The *account maintenance* component is needed to record the withdrawal and deposit operations as debit and credit mutations, respectively. This component also records payments in e-cash received from, or made to, the issuer or other banks that could be made to manage e-cash liquidity. The bank's web API server has to implement the API calls to handle the online customer requests for forwarding to the *e-cash withdrawal*, the *e-cash deposit* and the *withdrawal/deposit **ART** factory* components. The bank's customer database provides the personal data required for provisioning identifiers for customer **ART**s.

The *e-cash withdrawal component* handles requests for withdrawals.

The *e-cash deposit component* handles the deposits. This component is realised as an externalised and specialised version of the *receive payment component* in the *offline digital payment integration* component, leveraging the software only e-cash acceptance inherent to aggregating receipt token technology.

The *withdrawal/deposit **ART** factory* component is an instance of the aggregating receipt token factory component presented in section 3.3. In addition providing the specially constructed deposit and withdrawal tokens, this component can service the bank's customers in providing any other **ART**s they need.

A deposit token is an **ART** that contains the identifier for the bank's e-purse[19] and is additionally configured with the account number of a particular customer; it is used to make a series of deposits into this account. This token is stored in the customer e-purse as mentioned in section 4.1 where it is updated after each deposit.

The *withdrawal/deposit **ART** factory* provides the first deposit token to a customer when they sign up for e-cash deposits and withdrawals. It is then automatically replaced by the bank with a new one shortly before its expiration as part of the online message conforming the deposit.

The user initiates an e-cash deposit in the e-purse user interface by making an e-cash payment to the bank with the provided deposit token specifying the amount of the deposit. The bank's *e-cash web API*[20] receives this token, which includes the deposit amount, and feeds it to the *e-cash deposit* component, where its acceptance as a valid e-cash payment leads to recording a credit mutation for the benefit of the customer in the bank's *account maintenance* component. The bank maintains an account for the assets it holds as e-cash balance in its e-purse that follows all e-cash movements; this account is debited in the deposit database mutation.

[19] Technically the token is one that has been issued to the bank.

[20] A web interface is likely to be the most prevalent way of facilitating communication between a bank and its customers for e-cash withdrawal and deposits. Other forms of communication, e.g. emails with QR codes could also be used.

A withdrawal token is an **ART** that contains the identifier for the customer's e-purse. Like a deposit token, it is configured with the customer account number and provisioned to, and refreshed in, the customer e-purse. Additionally, the withdrawal token is also configured with a public cryptographic withdrawal authorisation key[21] for the user. This cryptographic key was created by its e-purse at customer sign up. The **ART** is digitally signed by the *withdrawal/deposit **ART** factory* component; with this signature the token acts as the certificate for the withdrawal authorisation public key. Refreshing the token also refreshes the certificate.

The user initiates an e-cash withdrawal in the e-purse user interface - this can either be done explicitly or it can be triggered by a low balance. The withdrawal is done as an e-cash payment request containing the withdrawal token sent to the bank's *e-cash web API*. The payment request is extended with a digital signature created by the e-purse with the customer's withdrawal authorisation key. After sending it to the bank's *e-cash web API* it is processed by the *e-cash withdrawal* component.

The *e-cash withdrawal* component accepts the request by validating the digital signature and then initiates the e-cash payment to the customer's e-purse. The first step in this payment is a debit mutation of the customers account for the requested amount into a pending e-cash withdrawal account. In this database mutation the withdrawal request is recorded as justification with the **ART** in the request as reference.

The *payment schedule* component in the bank's *offline digital payment integration* component monitors the pending e-cash withdrawal account and performs any uncompleted e-cash payment to the customer by passing the request with the recorded **ART** to the bank's Mon**E**box. After the Mon**E**box has computed the **ART**, the withdrawn amount is transferred from the pending account to the e-purse shadow account with the updated token as reference. The token with the withdrawal amount is sent to the user via the *e-cash withdrawal* component and the *e-cash web API*.[22]

As both deposit and withdrawal are a payment to the user itself AML checks by the bank are not needed.[23]

The e-purse *offline digital payment integration component* (c.f. fig. 10) for the user is extended: An additional *deposit&withdrawal* functional component stores the two special tokens provisioned by the bank. The customer's *payment schedule* component can be used to configure regular deposits, for instance, a merchant depositing daily receipts remaining after paying its suppliers in e-cash.

[21] The withdrawal key can securely be stored in the e-purse, it's use is illustrated by Auer en Böhme[1, p. 94] as token access to a (**CBDC**) bank account.

[22] Recording both the incoming and outgoing **ART** as references in the pending account mutations guarantees atomicity of the full withdrawal transaction. It makes the withdrawal API call asynchronous and also idempotent. This means that the user can retrieve the withdrawn funds at any time after the request was sent.

[23] In theory a user could pass on a deposit token to a third party in order to receive an e-cash payment directly in its bank account. Doing this will exhaust the deposit token early, which could be noticed by the bank. The user will not know that a payment has been received until a check of the bank account, whereas with a direct e-cash payment the user will know this immediately, so there is little benefit in doing so. Also, like all used **ART**s, deposit tokens are reviewed by the issuer for suspicious transactions, and a pattern of many deposits can be recognised as different from other user deposits in which case the account owner, indicated by the number in the token can be asked to explain this.

In addition to operating an **ART** factory, a bank also operates the *user pseudonym* component needed by the token factory. The *identity register* component for customers enrolling as e-purse users can be created as an extension of the bank's existing customer database.[24] These e-purse support components are essentially generic system agent components, indicated in fig. 13 by the blue colour also used in fig. 9.

The *e-cash balance maintenance* component interacts with e-purses owned by other banks, including the issuer, to manage the bank's liquidity in e-cash.[25] This component executes the liquidity management decisions by making and receiving wholesale e-cash payments as reciprocal deposit and withdrawal operations to simultaneously update both e-purses the corresponding shadow account and the counterparty account. To this end this component effectively replicates the functions of both the *e-cash deposit* and *e-cash withdrawal* components modified to use standard inter-bank messaging formats for the value transfer. For this purpose the interbank message format can be extended to include the aggregating receipt token that corresponds with the requested or completed wholesale movement of e-cash. The **ART**s and key certificates used in wholesale e-cash payments could be provided by the issuer (c.f. fig. 14).

Figure 13 shows the MonEbox implemented in just a single banking-standard Hardware Security Module (HSM). To scale withdrawal capacity, with each withdrawal requiring a payment computation by the bank's MonEbox, the e-purse *offline digital payment integration component* can be extended to communicate with multiple MonEboxes sending them payment requests in round-robin fashion. With multiple MonEboxes operating in parallel latency for withdrawals, which is an asynchronous operation anyway, can be kept low. The deposit function, on the other hand, does not need more than a single MonEbox as its capacity to receive deposits can be scaled up by adding instances of the e-cash deposit software component to the bank's server infrastructure. Redemption operations by this single MonEbox of received e-cash can be queued to even out bursts in deposits. The MonEbox that redeems the deposit payments stores the identifier for the bank embedded in the deposit **ART**s. If at some point in time redemption capacity for deposits actually needs to be increased another MonEbox can be deployed which will have a different identifier, and which could be used to provide load balancing for queued redemption operations.

The use of **ART** technology for deposits, withdrawals and wholesale tranfers shows how e-cash can seamlessly interface with online banking by adding extra, account holder specific data to aggregating receipt tokens dedicated to deposits and withdrawals.

As both the deposit and withdrawal of e-cash are operations that have associated costs, a bank could ask a fee for these operations. For users where this is suitable an itemised fee could be made payable at the time of provisioning the withdrawal and deposit tokens as these tokens are configured for both volume as well as individual and aggregate amounts.

[24] In practical implementations most of the personal data stored in the user identity component can be represented as a reference to the customer database.

[25] The business relationship between banks and the issuer for managing e-cash will typically be specified in a *'rule book'* established by the issuer.

6 The issuer's core operations

Figure 14 shows the IT components implementing the *issuer core* component to support system agent components (c.f. Section 3.1) and to process system operational data to analysis performance, security and monetary aspects. It also shows how the core components interact with existing functional components in the issuer's IT system and its human operators.

System security for e-cash is realised with specifically provisioned secure hardware in the form of the MonEboxes with different levels of strength, applying strong cryptography to protect communications including the transfer of monetary value, the systematic management of the cryptographic keys used and the observation of the flow of money to detect anomalies. The issuer manages the system through payment credentials that are periodically provisioned to each user by system agents (c.f. section 3.3). Payment credentials communicate operational parameters to the user devices and are digitally signed on behalf of the issuer by the system agent.[26]

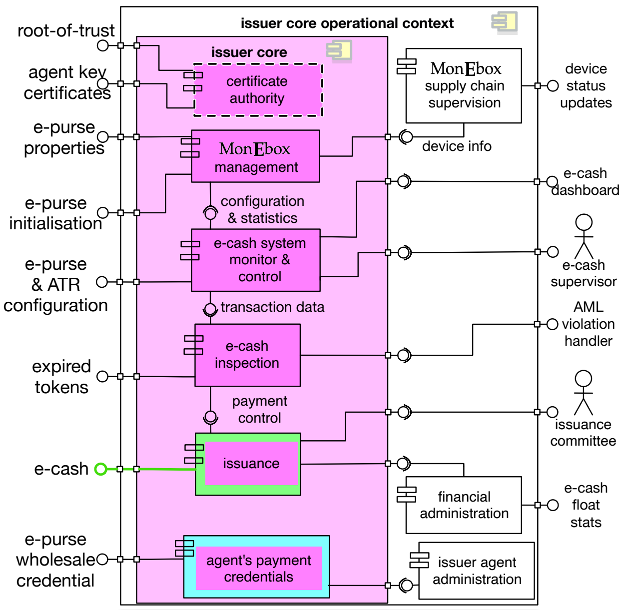

Figure 14: The core functional components of the e-cash issuer.

The *certificate authority* component validates the cryptographic keys used by an agent with cryptographic certificates. The agent keys are used to certify the keys that sign the payment credentials; an issuer certificate expresses that these user credentials are endorsed by the issuer. These top-level key certificates periodically expire and the system agent component creates a new key and obtains its certificate from the *certificate authority*. This functional component is the foundation of all digital security certificates used in the e-cash system; the public key used in this component to digitally sign the key certificates is self-certified, specifying it as the e-cash system's root of trust. As mentioned in section 4, the root of trust is installed in all e-purse related devices.

The issuer is responsible for the provisioning of MonEboxes to e-cash users, manufactured to an approved design and supplied in accordance with a specified secure supply chain. The *MonEbox supply chain supervision* component shown in fig. 14 implements this. The functions of this component are very similar to the operational involvement of an issuer with provisioning banknotes; showing this component outside of the *issuer core* indicates this similarity.

The *MonEbox management component* in the *issuer core* keeps a database of

[26]System agents store and use their secret keys in a Hardware Security Module (HSM). Such modules typically support key generation and certification functions.

all supplied devices with physical security characteristics and unique manufacturing identifier. It records their operational status and the deployment of an Mon**E**box is marked by a unique anonymous user identifier and associated secure communication keys to be used in the credential requests sent during its operational live.

The Mon**E**box supply chain includes the secure delivery of fully tested and security initialised 'blank' Mon**E**boxes to system agents operating an e-purse provisioning stations (c.f. fig. 8). The provisioning station interacts with the *Mon**E**box management component* to authenticate the device as genuine to then complete its initialisation by installing user specific details like the e-purse configuration and its user identifier and communication protection keys.

The *e-cash inspection* component receives expired aggregating receipt tokens from ***ART** factories* operated by system agents and reviews the payment log contained in each of them. The analysis addresses:

- system security by revalidating each recorded payment to determine i) the validity of the computed payment signature, ii) the validity of the certificate chain for the payment key used, iii) the correctness of the computed token value, iv) the use of each payment key as conforming to its configuration and v) the integrity and correct use of the credential-certification keys by system agents;
- monetary integrity by comparing the differential velocity of money for randomly partitioned users to detect counterfeit payment keys;
- the aggregate velocity of money to inform monetary policy decisions;
- potential violations of regulation to prevent money laundering and tax evasion (**AML**).

Except for the results of the AML analysis, the system-operational data obtained from analysing the used **ART**s provides input for the *e-cash system monitor and control* component. This component provides the system operators with a dashboard of digital money flows. The velocity of money can be tuned to address monetary concerns by adjusting credential configuration and then pushing these changes to the system agents.

A potential security violation can be responded to by disabling further use of involved keys and by initiating a process to isolate a suspected e-purse. This component also plays the operational part in supervising the system agents that are providing the e-cash payment credentials. The operational information can be presented in real time but reflect activities that happened over a period of time in the past.

The detection of AML violations is based on a pseudonymous register of payments received by users to detect suspect patterns. A regular random selection of users for more detailed monitoring over different periods to both gather average spending patterns and detect anomalies are complemented by a rules-based analyses of one-off anomalous payments. When an anomalous transaction or sequence of suspect transactions is detected, the transaction data is passed on the *AML violation handler*, an (external) entity that has been empowered to investigate such matters. The investigation may result in personal information being demanded from the *user pseudonym component* that supplied the user pseudonym in the **ART** used by the payee (c.f. fig. 9). For one-off exceptional payments such an investigation could entail a request to a user to provide a supporting document, which could actually be done without full knowledge of personal payee information with intermediation by the system agent that operates the user pseudonym component, e.g. by forwarding the request to a registered (e-)mail address without revealing that address.

The issuer provides liquidity for e-cash users with its *issuance* component that incorporates the issuer's e-purse, which has a Mon**E**box extended with a function to handle issuing events where its e-cash balance in increased without receiving a payment. An issuing event can be implemented to requires an instruction from a qualified number of specially designated users for which identifying information, e.g. a public key, is stored in the issuance Mon**E**box. The figure shows these operators as an *issuance committee*.

Like any other e-purse the issuer's e-purse can receive or send e-cash, in this case for *wholesale* transfers, to and from financial institutions and to fund new user e-purses while they are being issued. In the latter case the e-cash flows via the e-purse in a *provisioning station* (c.f. fig. 8). The issuer's e-purse further operates like any other bank's e-cash component in interaction with its financial administrative system and it support withdrawals and deposits by the issuer's e-cash-using "customers", the e-cash handling banks and issuing station operators.

The IT architecture for banks in supporting their e-cash using customers shown in fig. 13 is the template for implementing the issuer operations for distributing issued e-cash. The payment credentials needed in the interbank transfer of e-cash can be provided by the issuer with its *agent's payment credentials* component. Provisioning credentials uses a register of approved agents to create the needed identifiers and other information like account numbers to be incorporated in them.

The implementation of the core issuer operations does not need high-capacity, high-availability transaction processing. Many issuer functions can be implemented as separate functional units that can be easily replicated for resilience and to increase capacity when growth of the e-cash system user base shows this could be beneficial.

7 Conclusion

E-cash is a new type of money distinct both from physical cash and from digital money in bank accounts. Payments in e-cash are transferable and it can circulate securely between its users.

With aggregating receipt token technology e-cash can be realised at the scale required for a large economy with hundreds of millions of users where it can provide offline, and online, payments in any amount with the appropriate security to any user over decades of operation. In his technology e-cash seamlessly complements existing types of money.

The *money domain* e-cash combines high system security with a consistent, true cash-like user experience with payer anonymity and without per-payment transaction costs. The e-purse software provides a rich user interface and seamless integration with other IT systems, in particular those used by banks. An e-cash system provides a very high transaction capacity with operational costs lower than both cash payments and bank intermediated ones. E-cash provides *income transparency*: a pseudonymous, undeniable, secure digital record of any digital income. This makes it unattractive for money laundering and tax evasion.

The *security domain* in operating an e-cash system includes a secure device, the e-purse, operated by its users. It also includes the secure devices (HSMs) operated by system agents to intermittently provision specially constructed, issuer-endorsed digital credentials to users that are required to make and receive each payment. Inspecting expired payment credentials provides the issuer with knowledge of system performance and actual security; in setting the configuration parameters of these credentials the issuer can impose monetary and security constraints on every transaction without impacting ordinary use to tune operations and respond to security incidents.

An e-purse operates a tamper-detecting secure hardware component operating as a digital bearer payment instrument in combination with matching software running on a trusted user computer. The e-purse security can be constructed with scalable physical security to match the amounts of money stored. The e-purse software provides a rich user interface and seamless integration with other IT systems, in particular those in banks.

Aggregating receipt token technology e-cash is an innovation in money that would offline digital payments practical and ubiquitous.

8 Bibliography

[1] Raphael Auer and Rainer Böhme. 'The technology of retail central bank digital currency'. In: Quarterly Review. Bank for International Settlements, Mar. 2020.

[2] Innovation hub experts. 'A high-level design guide for offline payments with CBDC'. In: Project Polaris part 4. Bank International Settlements, Oct. 2023. ISBN: ISBN 978-92-9259-701-6.

[3] Eduard de Jong. *Cash: The once and future king*. Jan. 2023. URL: https://eduard.dejongfrz.nl/papers/latest-kingreturns.pdf.

[4] Eduard de Jong and Peter Cattaneo. *Ledgers and tokens for Central Bank Digital Currency*. online. June 2023. URL: https://eduard.dejongfrz.nl/papers/latest-ecashorledgers.pdf.

[5] Eduard de Jong and Peter Cattaneo. *Offline payments and CBDC: A tale of three currencies*. online. Mar. 2024. URL: https://eduard.dejongfrz.nl/papers/latest-3currencies.pdf.

[6] Ignacio Mas. 'Savings as forward payments: innovations on mobile money platforms'. In: ed. by Essam Yassin Mohammed and Zenebe Bashaw Uraguchi. (September 2, 2011). Chapter 10 in ".". London: Routledge. , Available at SSRN: https://ssrn.com/abstract=1825122 or http://dx.doi.org/. Routledge, London, 2011. Chap. chapter 10, Financial inclusion for poverty alleviation: Banking on the unbanked. DOI: 10.2139/ssrn.1825122.