applied system innovation

MDPI

Article

Making Tax Smart: Feasibility of Distributed Ledger Technology for Building Tax Compliance Functionality to Central Bank Digital Currency

Panos Louvieris *, Georgios Ioannou and Gareth White

Department of Computer Science, Brunel University of London, London UB8 3PH, UK; georgios.ioannou@brunel.ac.uk (G.I.); gareth.white@brunel.ac.uk (G.W.)
* Correspondence: panos.louvieris@brunel.ac.uk

Abstract: The latest advancements in distributed ledger technology (DLT) and payment architectures such as the UK's New Payments Architecture present opportunities for leveraging the hidden informational value and intelligence within payments. In this paper, we present Smart Money, an infrastructure capability for a central bank digital currency (CBDC) which enables real-time value-added tax split payments, oversight, controlled access, and smart policy implementation. This capability is implemented as a prototype called Making Tax Smart (MTS), which utilises the open-source R3 Corda DLT framework. The results presented herein confirm that it is feasible to build an MTS capability which is scalable and co-exists with the current payment systems. Smart Money CBDC has the potential to mobilise payments data, transforming the role of money from a blunt instrument to a government policy sensor and actuator without disrupting the existing money system. DLT, smart contracts, and programmable money have a crucial role to play with benefits for government departments, the economy, and society as a whole.

Keywords: CBDC; tax compliance; DLT; programmable money

Citation: Louvieris, P.; Ioannou, G.; White, G. Making Tax Smart: Feasibility of Distributed Ledger Technology for Building Tax Compliance Functionality to Central Bank Digital Currency. *Appl. Syst. Innov.* **2024**, *7*, 106. https://doi.org/10.3390/asi7060106

Academic Editor: Luís Oliveira

Received: 12 September 2024
Revised: 12 October 2024
Accepted: 21 October 2024
Published: 29 October 2024

1. Introduction

Distributed ledger technology is a major catalyst for change in the money system and the administration and control of money. Furthermore, DLT offers promising potential for disrupting the existing information management frameworks of government departments, such as the Bank of England (BoE) and His Majesty's Revenue and Customs (HMRC), in order to overcome the friction of public policy decision making. DLT's are already being employed in diverse business applications and have matured significantly since the inception of the first established network of this kind, Bitcoin [1]. This is the first and most popular digital currency and payment system employed on a DLT platform [2] and was the first of its kind to eliminate the need for a single entity that accounts for and controls a currency [3]. Its success has led to a market capitalisation that peaked USD 1.43 tn in March 2024 and the subsequent emergence of numerous alternative digital currencies (also known as cryptocurrencies or cryptoassets) such as LiteCoin, Ethereum, Ripple, and Bitcoin Cash.

The success of digital currencies is attributed to the perceived failures of governments and central banks during the 2008 financial crisis, including serving as cheaper alternatives to existing debit and credit card systems [4]. Just as with cash, i.e., banknotes, digital currencies are considered a medium of exchange, a store of value, and units of account [5]. However, the adoption of de-regulated cryptocurrencies creates problems associated with the volatility of cryptoassets, which poses a risk for buyers and sellers; the risk of payment systems which operate outside the regulatory framework; and the use of cryptocurrencies for money laundering and financing terrorist groups [6].

Meanwhile, the role of DLT has evolved from a "digital currency ledger" to a medium for realising transactions between unknown participants [7]. Further applications include

disintermediated environments that enable trust-less payments [8] and decentralised and immutable storage [9]. DLT is used in a number of government applications. Examples include notary services for citizens [10], inventory management [11], patent [12] and energy trading [13], securing archives of public documents [14], B2G information sharing [15], and land registry [16].

Central banks across the globe, as part of public sector explorations of DLT, have been considering providing DLT-enabled digital currencies [17–19] directly to the general public through accounts held in their books [20] as a complement to fiat currency, which is currently offered via cash or through commercial banks. Recent research efforts around the globe lead by central banks [21–24] are actively exploring the characteristics, design criteria, implications, and acceptance of central bank digital currencies, also known as CBDCs. Developments in DLT, particularly those related to expanding the functionality of smart contracts, accentuate the concept of 'programmable money' [24], which creates opportunities for fiscal policy implementation. Similarly, controlling spending on social benefits such as Universal Credit [25] can be provided in the form of special-purpose tokens which can be spent exclusively on particular classes of goods and services, e.g., food, rent, and medicine. A CBDC can operate as a public policy actuator with just-in-time targeted interventions in the event of financial or health crises with large economic impact.

The advent of DLT and CBDC comes at a time when payments are undergoing radical changes on a global scale, triggered by events such as the implementation of Payment Services Directive 2 [26], open banking [27], the ISO 20022 electronic data interchange standard [28], the transformation of payment systems such as UK's New Payments Architecture (NPA) [29], and the transformation of cross-border payments via the introduction of SWIFT GPI [30]. Payment messages are now carrying "enhanced data" elements concerned with the contextual information of a transaction, e.g., item description, item quantity, unit cost, VAT invoice number, VAT amount, Legal Entity Identifier, and more. This poses an important question concerning how these data should be exploited for their information value within a proposed Smart Money system for greater societal and economic benefit?

We present Smart Money, a central bank digital currency architecture, as an essential component for an advanced digital economy. This work is motivated by the Bank of England's desire to exploit CBDCs as a potential and feasible supplement for cash, which has implications for the payment system and government authorities [31]. Furthermore, this paper considers that CBDCs have an additional role, namely to assist with other government departments' (OGDs) policies such as automating tax payments at the point-of-sale. For the first time, the design and implementation of a feasible DLT-based approach which integrates smart contracts with the existing and future payment infrastructure and causes minimal intervention and disruption to the financial system is proposed in this paper. The design specification proposed herein facilitates trusted information management and sharing among OGDs in order to mobilise data and introduce smart policy making.

Of particular interest is the issue of VAT split payments, where the VAT is deducted from a payment to be transferred directly to His Majesty's Revenue and Customs (HMRC), and the supplier receives the net amount of the transaction. Within the Making Tax Smart prototype presented herein, split payments are processed via smart contracts based on the information recorded on the invoice. This approach leverages the informational value offered by payments' enhanced data feature, presented in the payment messages of the NPA [29], providing further benefits for OGDs; specifically, CBDC facilitates the gathering of economically relevant data in near real-time to support compliance and oversight of public policy planning and development. Finally, MTS introduces 'smart warrants', a mechanism that enables permissioned access control over the transaction data.

1.1. Aims

Develop a CBDC which provides real-time precision information management capability to transform money by enhancing its informational value for government policy applications in a feasible and scalable manner.

1.2. Objectives

1. Review the current efforts in developing DLT-enabled CBDCs and the opportunities that this technology brings for overcoming the frictions of government policy making.
2. Design a DLT-based CBDC architecture that provides new capabilities for governments which are non-disruptive to existing payment systems. The objectives of these capabilities are (i) to utilise smart contracts for delivering near real-time tax collection via VAT split payments and permissioned data access control for tax compliance via smart warrants and (ii) to demonstrate the programmable money concept for implementing smart policies to ensure control over spending.
3. Demonstrate the feasibility and scalability of the CBDC architecture within the context of a smart contract-enabled VAT split payments use case.

1.3. Research Contributions

Through the exploration of a central bank digital currency, this paper will show that the traditional definition of money (medium of exchange, store of value, unit of account) is limited. Due to the latest advancements in distributed ledger technology and the increasing demand for digital government transformation, money can now be a significant catalyst for change by extending the traditional definition of money to include "money as a sensor and a policy actuator". Smart Money is the implementation of this new definition of money, and this paper demonstrates the feasibility and scalability of the Smart Money proof-of-concept within the context of the VAT split payment use case presented here.

The rest of the paper is organized as follows: Section 2 covers the current state-of-the-art in CBDCs and the issues that this tech promises to resolve, which concern state governments. In Section 3, the motivation behind Smart Money and the intersection of DLT-enabled capabilities with smart policies is presented. Section 4 provides the specification of the Smart Money system. The UML diagrams and the methodology for evaluating the feasibility and scalability of the design are presented in Section 5. Sections 6 and 7 present the results of the feasibility and scalability tests. Finally, the conclusions are presented in Section 8.

2. Related Work

2.1. Distributed Ledger Technology Frameworks

A distributed ledger constitutes a medium of transaction storage for participating entities or nodes whereby all participants maintain a synchronised copy of the ledger. The rules for accessing the ledger are determined by the characteristics and requirements of the DLT application. Trust-less ecosystems such as the Bitcoin and Ethereum blockchains utilise a public ledger [32] where all transaction data are publicly available, albeit user identities are encrypted. These ledgers are permissionless [33]; hence, anyone can submit and verify transactions on the network, and all network participants have the same permissions [34]. Blockchains such as the R3 Corda [35] and Hyperledger Fabric [36], which are optimised for fintech applications, employ a permissioned approach. In this paradigm, the ledger is accessible only by authenticated entities, and the level of access for each entity is defined by the business logic [3]. For example, customers are only allowed to submit transactions (i.e., issue payments), whereas payment settlement is processed by entities such as notary services or delegated validating groups of servers [37]. The permissioned approach is commonly employed in the financial sector because transactions involve personal data. Therefore, the ledgers are usually private, and non-authenticated entities are not allocated any access rights. In addition, a ledger may be public and permissioned; hence, its contents may be publicly available, but transaction submission and verification are only permitted by designated entities [9]. A special case of a permissioned access model is the consortium blockchain [38], where a limited number of entities is given certain privileges to validate transactions [39]. These consortiums are formed by multi-lateral agreements between businesses such as financial institutions or mobile telecommunications providers [40].

2.2. Central Bank Digital Currencies

In addition to providing security for financial services, DLT also provides provision for secure central bank digital currencies [41]. CBDCs can be considered simply as fiat currencies issued by a central bank in place of banknotes [42–44] or complementary to cash [45], and are made available in electronic form through direct accounts with the central banks [46]. Similarly, CBDCs can be considered an electronic form of central bank money that allows for peer-to-peer transactions without the need for an intermediary [18]. In addition to serving as a means of payment, CBDCs are considered to operate as a store of value [47]. In [48], the term CBDC refers to "a central bank granting universal, electronic, 24 × 7, national-currency denominated and interest-bearing access to its balance sheet".

So far, only two countries (Bahamas and Jamaica) have CBDCs in operation [49]. Similar efforts are underway, where CBDC is considered as part of a new interbank settlement system or as a replacement for cash [23]. Bank of England's Digital Pound project [50], which is still in the design phase looking at the technology and policy requirements for a digital pound, is designed as a compliment to cash and focuses on the issues of privacy and offline payments. The decision to move to the build phase will be made in 2027, with a possibility of being launched by 2030 [51]. The European Central Bank has also been exploring technical solutions for a CBDC, focusing on the issues of anonymity and privacy [52,53]. Project Rosalind [54], a collaboration between BOE and the Bank for International Settlements (BIS) is investigating how a CBDC wallet and API for a DLT-based CBDC should be designed and implemented. At the same time, central banks are researching CBDCs beyond their jurisdictions to explore their potential for cross-border payments. Notable examples are Project Icebreaker, with the participation of the central banks of Israel, Norway, and Sweden [55] and Project mBridge, which involves central banks of Hong Kong, Thailand, UAE, and China [56].

As common and standardised definitions and frameworks for CBDCs remain to be finalised, current CBDC models are specified according to two key parameters: accessibility (wholesale versus retail) and utility (retail payments versus interbank settlements). Accessibility and utility are tightly coupled because the purpose of a CBDC prescribes who has access to the currency, namely central banks, commercial banks, "narrow banks", non-fintech firms, and households. Wholesale CBDCs are central bank liabilities that are only accessible by banks and non-bank financial institutions. They are concerned with transactions that are processed through the Real-Time Gross Settlement (RTGS) system of a central bank, which involves interbank settlements, liquidity provision, and monetary policy implementation [57]. Retail CBDCs are central bank liabilities which are effectively identical to banknotes, and central bank deposits and can be used as an instrument for retail payments [43]. All transactions between accounts would be validated and processed by the central bank [46] or directly from payer to payee [58]. No third-party intermediaries are involved in payments; therefore, transactions are settled within a settlement platform which is independent from the RTGS system [44]. Hybrid CBDCs aim to bridge the trade-offs between wholesale and retail models by providing intermediary access to CBDCs for households and non-fintech firms. This is achieved by (i) exchange services provided by banks or "narrow banks" for buying and selling CBDCs in exchange for deposits [42,59] or (ii) by fintech firms that offer financial assets which are backed by CBDCs [59]. Cross-border CBDC designs are also being investigated in efforts to overcome the intricacies of costly and slow correspondent banking chains [60]. Domestic CBDC models are illustrated in Figure 1.

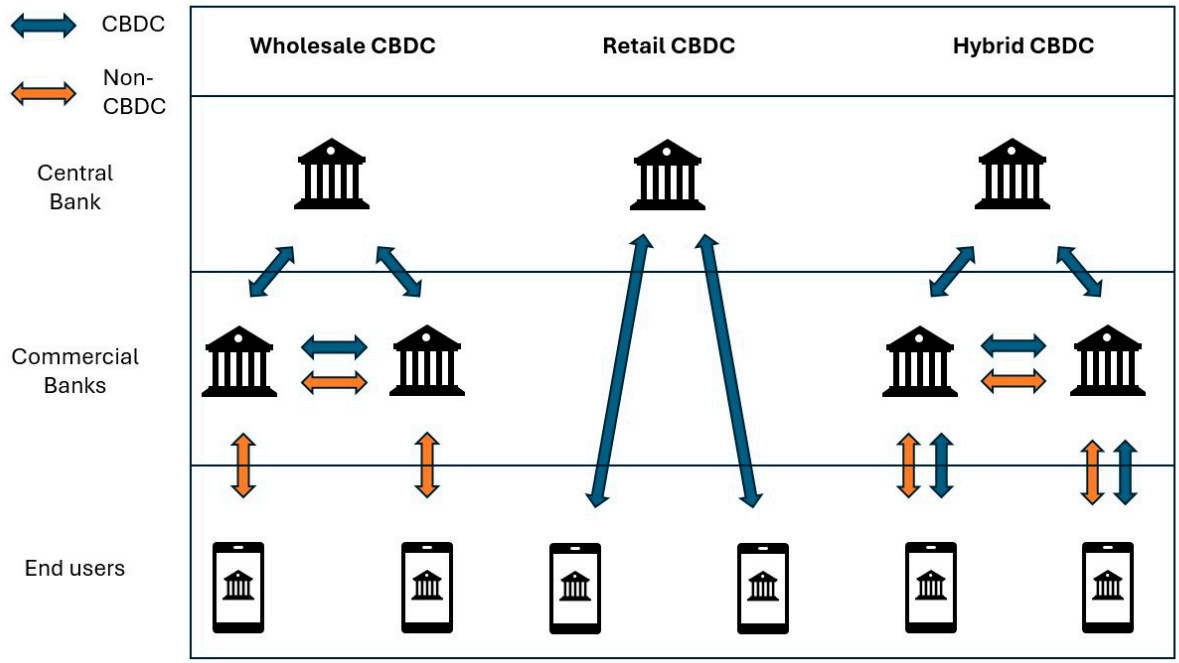

Figure 1. Central bank digital currency models.

Outside central bank initiatives, only a handful of other, yet partial, CBDC prototypes can be found in the literature. A design for CBDCs which is based on a custom blockchain framework is proposed in [61], but an implementation has not been provided. A demo implementation of a CBDC based on the Cosmos blockchain is presented in [62], although it has not been tested for feasibility and scalability. The work of [63] proposes an implementation which is based on a custom blockchain framework using a modified Byzantine fault tolerance consensus scheme [64]; however, no smart contract functionality was adopted. These prototypes have not been developed to their logical conclusion, nor have they been adequately tested.

The work presented herein follows a model for a CBDC which is similar to Riksbank's [23] and the R3 Corda ledger because of its unique permission and access–control management (for more details, see Section 4).

2.3. CBDCs as Programmable Money

Central banks work closely with other government departments. For example, in the UK, the Bank of England works with HM Treasury and HMRC, where CBDCs have a new and major role to play. Namely, CBDCs have heralded the advent of programmable money, enabling controlled policy implementation for these departments and overall governance improvement.

Conventional money has been hitherto issued, stored, and spent in digital form but has only served as a blunt instrument strictly corresponding to value. Developments in distributed ledger technology have the potential to add an important feature to money, thus making it 'programmable' [24]. This property enables a number of new capabilities such as payments conditioned on rules or events [65], delivery versus payment (DvP) settlements [24], or auto-pay bills formalised into multi-party payment contracts in order to permit net instead of gross money flows [66]. Central banks' interest in programmable money is emerging and causes them to rethink the interface to money [67]. Unfortunately. this interest remains mostly constrained to the potential for regulating cryptocurrencies [68] or for micropayments, which are moot points and overlook the potential of programmable money for improving governance and policy implementation.

The functionality of smart contracts, which is to run generalised programs on a shared blockchain [67], creates new capabilities for central banks and governments by utilising

programmable money as a multiplier of public policies, e.g., monetary, financial, and fiscal policies. This brings opportunities to transform the traditional role of money and the way that public policy is informed and applied. With DLT-enabled payments, purchases of goods and services can trigger VAT split payments to the tax authority so that sales tax is paid directly. The concept of DLT-enabled real-time tax payments was put forward by [69] as a means of simplifying accounting and reducing administrative burden for businesses. Neither a practical implementation nor a design were, however, included. Reference [70] has proposed a VAT settlement solution where the entities involved (seller, buyer, banks, tax authority, and auditor) are participating in an Ethereum-based DLT network. In the evaluation, the proposed software artifact does not transfer funds between any accounts, i.e., seller, buyer, or tax authority, nor does it confirm any account balances during the transaction. It serves exclusively as an accounting system but does not initiate or process payments. Transaction information is publicly accessible, which poses confidentiality risks for the counterparties. Furthermore, the software artifact has not been empirically tested for scalability. In terms of enforcing tax compliance, the proposed artifact poses certain risks: VAT settlements are validated by an 'auditor' instead of the tax authority directly, which is not a proper assurance mechanism for tax compliance; also, the VAT payment is made by the seller instead of the buyer. This may cause the payment to fail if the seller is overdrawn and the balance is insufficient for fulfilling the tax payment. Therefore, the works of [69,70] cannot be considered as 'programmable money' applications for VAT split payments.

In addition, the government can issue special-purpose 'money' within the context of social benefits to be used for particular classes of goods and to ensure that public funds are spent for the benefit of the polity. Further, programmable money can be tracked and traced, facilitating 'follow-the-money' investigations on behalf of law enforcement in the pursuit of money laundering and terrorism financing. It is evident that the 'programmability' property assigns new functionality to money, which can now operate as an information sensor and a policy actuator that enables governments to utilise money to develop and enforce policy concerned with tax collection and social care.

2.4. The Opportunity of CBDCs for Governance of Money Flows

A DLT-enabled CBDC can provide real-time access to transactional data [71], which allows for larger sample sizes that support more detailed analysis regarding space and time and is better for measuring certain behaviours because it overcomes biases reported in surveys [72]. Moreover, transactional data that are updated frequently or continuously [73] enable an increase in sample size and granularity and bring considerable benefits beyond existing surveys for creating real-time maps of financial and activity flows across the economy [74]. These maps will enable government oversight, i.e., to observe the effects of policy decisions, observe the flows of funds within the economy, monitor compliance of regulated firms [75], and detect payment fraud. This brings additional implications in terms of overcoming the "data siloing" effect by unlocking the transactional data stored on the ledger for OGDs and providing them with intelligence of wider coverage [72,76,77]. Participants in a distributed ledger network can leverage the shared infrastructure to effectively streamline inter-departmental business processes and gain a consistent view of the data [78]. By having a global view of the data, government departments will now have the ability to access information owned by other agencies more quickly and more transparently, respecting account holders' privacy. This has further benefits for open banking services [27], whereby third-party service providers will have controlled access to account holders' transactional data. The open banking route constitutes an optimal approach to integrating DLT-enabled CBDCs with the existing payment rails in order to avoid disruption to the core banking system.

3. Smart Money

Smart Money is an envisioned national network where UK residents possess central bank digital currencies in addition to, or in lieu of, banknotes and coins. The CBDC is

not kept on a physical device such as a smartphone or a special-purpose device. Instead, the currency is held in special-purpose CBDC accounts which are provided for by private entities such as banks and other payment service providers (PSPs). CBDC accounts can be registered by natural persons and legal entities which are permitted to hold deposits in the UK, and account holders can interface with the provided services by means of electronic devices such as PCs and mobile devices. The CBDC money flows are stored and maintained on the Smart Money distributed ledger, which is governed by a number of departments such as His Majesty's Treasury, the Bank of England, His Majesty's Revenue and Customs, and the Ministry of Justice. These departments have permission to access the transaction data from the DL to fulfil their policy objectives and enforce compliance in accordance with legislation which is codified in smart contracts. The Smart Money CBDC is designed to integrate with the existing and future iterations of the UK's payment system and is therefore not designed as a public cryptocurrency such as Bitcoin or Ethereum.

3.1. Smart Money with CBDC Within the New Payments Architecture

At the core of the Smart Money ecosystem sits a private and permissioned distributed ledger network. Financial institutions (e.g., banks and payment service providers), government departments (e.g., Bank of England, HMRC, and ONS) are direct participants in the Smart Money system by means of dedicated nodes. In addition to payment and banking services through traditional channels, financial institutions provide CBDC wallets to their customers. Within the Smart Money system, these institutions are called CBDC wallet providers (CWPs). In the payments network, the CWPs continue to offer their services as in their traditional role of PSPs. In the Smart Money system, they provide CBDC payment services but also serve as gatekeepers of their customers' payments data. These data then become available to government departments that are pre-authorised, such as HMRC for automating tax collection. Financial institutions that offer payment services can operate under one of the following regimes:

- Operate exclusively on the payments network and interact with the Smart Money system through a correspondent CWP.
- Operate exclusively on the Smart Money system and interact with the payments network through a correspondent PSP.
- Operate on both networks with a dual PSP/CWP role.

The Smart Money system is designed to operate as an overlay to the standard payments network and records all the payments on a distributed ledger which is shared among its participants. The CWPs that participate in both the distributed ledger and payment networks serve as bridges between the two environments. When a CBDC wallet holder (equivalent to an account holder in the traditional banking sense) issues a payment through the payments network, irrespective of the selected money instrument (CBDC or bank money), their corresponding CWP copies the message from the payment network to the Smart Money system, acting as a bridge between the two layers.

The integration of Smart Money with the payments network becomes more relevant in the wake of the forthcoming change to the UK's payment system called New Payments Architecture. The NPA adopts the ISO20022 [28] messaging standard and introduces the following two key features of great importance to governance and oversight:

1. Enhanced data: payment messages will now carry additional information related to utility bills, invoices, salary payments, and financial assets in a standardised, structured format.
2. Request-to-pay: billers will be able to request payments from their customers through the payment system instead of sending an invoice or bill through conventional channels (e.g., e-mail, post).

These novel features will enable suppliers to input granular data associated with the transaction within the payment message and transmit that information directly to their customers, who can respond to that message and fulfil the payment request. This process

creates a money trail of that particular transaction which matches the payment with its associated metadata. This particular money trail is replicated on the Smart Money system and stored on its distributed ledger as part of the payment's lifecycle.

3.2. Smart Money for Enhanced Oversight

Aggregating all the money flows with enhanced data from the payment system in a common medium creates a database of unprecedented intelligence value for government departments. Payments are copied in real-time on the distributed ledger to enhance timeliness in accessing, processing, and analysing payments data. Maintaining a copy of payments from across the economy (retail, wholesale, and cross-border transactions) increases the coverage of intelligence which is recorded. The inclusion of enhanced data in the ledger ensures access to granular information about payments which captures their context and purpose. Moreover, government departments have smart contract-enabled access to an up-to-date database of information where integrity and accuracy are assured, thereby enhancing the availability of information.

The capability to create unparalleled intelligence value from payments transforms oversight and compliance exercised by government departments. Processes which are currently largely based on surveys and forms are now streamlined on the distributed ledger with benefits for all involved parties. Tax authorities can calculate income tax and tackle tax evasion and fraud by reviewing accurate and immutable records of payments. National statistics authorities will be able to collect unbiased and real-time data to produce high-quality statistics and insights to support government decision making such as monetary and fiscal policies.

3.3. Smart Money for Smart Policy

Money flows are involved in many aspects of governance, such as public expenditure, taxation, financial stability, price stability, and social benefits. Unfortunately, money in its current form is a blunt instrument and does not provide the means to gather data in real-time across a broad range of economic activities. Smart Money CBDC transforms public policy by means of exercising more power on the flows of money. This is achieved by realisation of 'programmable money' through the use of distributed ledger technology and smart contracts. Specifically, sales incur value-added tax (VAT), and businesses are liable to pay that tax to the government. In practice, businesses submit quarterly VAT returns where the aggregate VAT for a financial term is declared. This process incurs significant administrative workload both for businesses and government, in addition to foregoing the collection of valuable transaction data. With Smart Money CBDC, VAT can be calculated and deducted at the point of transaction with the use of smart contracts. These contracts are executed by all the interested parties, i.e., the buyer, the seller, and the tax authority, to ensure that only the correct amount of tax can be collected. This VAT split payment feature would be impossible to carry out in a secure, transparent, and trusted manner without a DL-enabled CBDC.

3.4. Controlled Access Through Smart Warrants

Transaction data are stored securely on the ledger, with confidentiality, integrity, and availability being preserved due to the properties of distributed ledger technology. However, law enforcement agencies may wish to obtain access to private data when certain circumstances are present, i.e., there is probable cause that an account holder (the data subject) has committed a criminal offence where their transaction history is highly likely to contain evidence to support a successful prosecution. Smart Money employs smart contracts for implementing functionality equivalent to conventional warrants for searching suspects' premises, albeit on payments data stored on the distributed ledger. This is achieved by incorporating a legal entity (such as a magistrate or public prosecutor) in the Smart Money system by means of a dedicated node. The purpose of that node

is to authorise data access requests by investigatory authorities that are participating on the network.

When a government department, such as HMRC, launches an investigation to seek evidence of illegal behaviour within the payments data of a particular CBDC wallet holder, a smart warrant request is executed as part of a smart contract. The legal entity then authorises that request and grants access to the transaction history of the data subject and stores that authorisation on the ledger. HMRC can then execute the smart warrant to gain one-time access to the data subject's account data by querying the transaction history from the corresponding CWP's transaction ledger. To ensure separation of duties within government departments, the HMRC node hosts two separate accounts for its purposes: one account is used for authorising VAT split payments, and another account is used for investigation purposes; the latter has no visibility of payments data unless a matching smart warrant is issued.

3.5. Extending the Definition of Money with Distributed Ledger Technology

With these new capabilities in place, money exceeds its traditional role as a medium of exchange, unit of account, and store of value [79], insofar as governments are employing money singly as an instrument for enforcing policy decisions within the context of the associated monetary value. To retrieve economic intelligence, governments are hitherto employing conventional, inert tools which cause friction in the way that policy is informed and exercised. Government departments will have real-time access to all economic activity routed through the domestic payments network to obtain valuable intelligence and to control money flows in order to enforce compliance. This new role of money is accomplished by the unique approach embodied within Smart Money, which is to leverage the informational value concealed in money flows by mobilising the enhanced data and intelligence contained within the payment messages. This approach is underpinned by the distinctive features of distributed ledger technology, which ensure integrity, accountability, and transparency of all activity that is concerned with money flows and managing the information associated with payments. Smart Money's contribution is premised on extending the traditional role of money which now also means money can serve as an information sensor for public policy.

3.6. Use Case: Making Tax Smart

Making Tax smart (MTS) is the core use case for Smart Money, which investigates the feasibility of issuing VAT split payments for supplies of goods and services within the UK. With MTS, every payment which incurs VAT, such as purchasing a computer, paying for rent, or legal services, triggers an additional payment that secures the matching value-added tax for the tax authority, His Majesty's Revenue and Customs. The split payments are issued on the Smart Money DLT by means of smart contracts and are validated by the associated parties (i.e., buyer, seller, and HMRC) in a VAT-able payment. The exact amount of VAT is derived at the point of transaction by inspecting the enhanced data elements contained on the payment message, i.e., the invoice issued by the supplier codified in a structured format. Split payments means that the VAT from each transaction is collected by HMRC in real time, which minimises the opportunities for fraudsters to perform VAT fraud or evade paying tax. This is of great importance to the UK tax authorities, as the estimated current VAT gap (i.e., total theoretical VAT liability minus the actual VAT collected by the tax authorities) is measured at £9.8bn, which amounts to 7.5% of the theoretical VAT liability [80].

4. Smart Money Corda Implementation

To assess the scalability and feasibility of the proposed CBDC, an experimental environment for the Smart Money system has been implemented in the R3 Corda open-source blockchain platform. Corda, as a permissioned blockchain framework which has been designed for financial services, incorporates features such as accounts, tokens, notarisation

services, and permissioned, smart-contract-enabled, on-ledger access, all of which are suitable for the implementation of the Smart Money building blocks.

4.1. Smart Money DLT Architecture

The Smart Money DLT is designed to be interoperable with existing and future payment services in the UK and to not interfere with the clearing and settlement systems. The Smart Money layer (which hosts the distributed ledger) co-exists with the payment systems layer (the existing payment systems infrastructure) and stores the exchanged messages in a transaction ledger, which facilitates the secure and accountable exchange of information. Effectively, the DL is not used for any type of clearing/settlement processes. Its purposes are (i) to maintain an immutable registry of money transfers for rapid retrieval and analysis, (ii) to facilitate citizens in exercising control and leveraging the value of their personal data, and (iii) to provide an audit trail for accountable data processing that preserves the privacy rights of data subjects.

In the Smart Money DLT architecture, payments are initialised by the seller by means of a "Request-to-Pay" message, or R2P for short. The recipient of the payment is responsible for registering the enhanced data information, such as value, VAT, buyer reference, seller reference, and the invoice or receipt for the transaction. In contrast to the current paper-based scheme where contextual data concerned with the transaction are not digitally recorded, the R2P message records all the relevant information as it is routed through the payments network. The buyer's PSP receives the message from the payments network but does not take any further action until such instruction is provided by the Smart Money layer. The payment service provider of the seller is responsible for dispatching the contents of the payment message to the Smart Money layer by registering the transaction on the distributed ledger. The buyer's CWP is therefore informed of the pending transaction from both sources: the payments network and the DL. The payment, however, can only be authorised from the Smart Money DLT by means of executing a smart contract. There are three stakeholders in this transaction: the buyer's CWP, the seller's CWP, and the DL node of the tax agency, HRMC. HMRC's role in the payment is to enforce tax compliance by validating that the buyer allocates the correct amounts to the seller and to paying VAT. As soon as the payment has been fulfilled in the Smart Money DLT, the buyer's PSP can release the funds through the payment network and finalise the payment.

4.2. Implementing Smart Money with R3 Corda

R3 Corda is a permissioned and private distributed ledger technology platform specially designed for financial industry applications and with smart contract support [81]. In contrast to other DLT frameworks, transactions in Corda are only announced to the involved parties on a need-to-know basis (see Figure 2). Each node (labelled CWPx) records only those transactions that it is involved with (i.e., via the accounts registered in the node CWPx) with the exception of the HMRC node, which records all transactions. All communications in a Corda network take the form of small multiparty protocols, called flows [35]. The unique permission and access–control management adopted in Corda is aligned with the Smart Money approach in terms of data sharing, particularly with the MTS scheme, where payments become multilateral instead of bilateral transactions. One major difference between Corda and other permissioned DLT platforms is the adoption of notary nodes. These are special-purpose dedicated hosts which participate in the network and serve the purpose of time-stamping and preventing double-spends of funds.

Corda supports the use of accounts within a node, which act as logical partitions of the state objects which belong to that node. This functionality, which is implemented via the *accounts* library [82] enables the hosting of multiple entities within a single node and eliminates the need for each entity to host its own proprietary node. Accounts are employed for two purposes: representing CBDC wallets and implementing permissioned data access control. Wallets act as bank accounts and are assigned to Corda accounts. Thus, each wallet in a node is isolated from the other wallets in order to protect funds

and personal data from eavesdropping or stealing. In the experiment, the *SellerCWP* hosts multiple accounts for retail businesses and the *BuyerCWP* hosts accounts for consumers. In the experiment, the *SellerCWP* hosts multiple accounts for retail businesses, and the *BuyerCWP* hosts accounts for consumers. For government departments, Corda accounts represent different administrative entities within an organisation, with a clear separation of duties. For HMRC in particular, there are two different accounts: *VATpayments* and *VATInvestigator*. The former is involved exclusively with validating payments with respect to the tax compliance element, and the latter is used for requesting access to an account's data through a smart warrant.

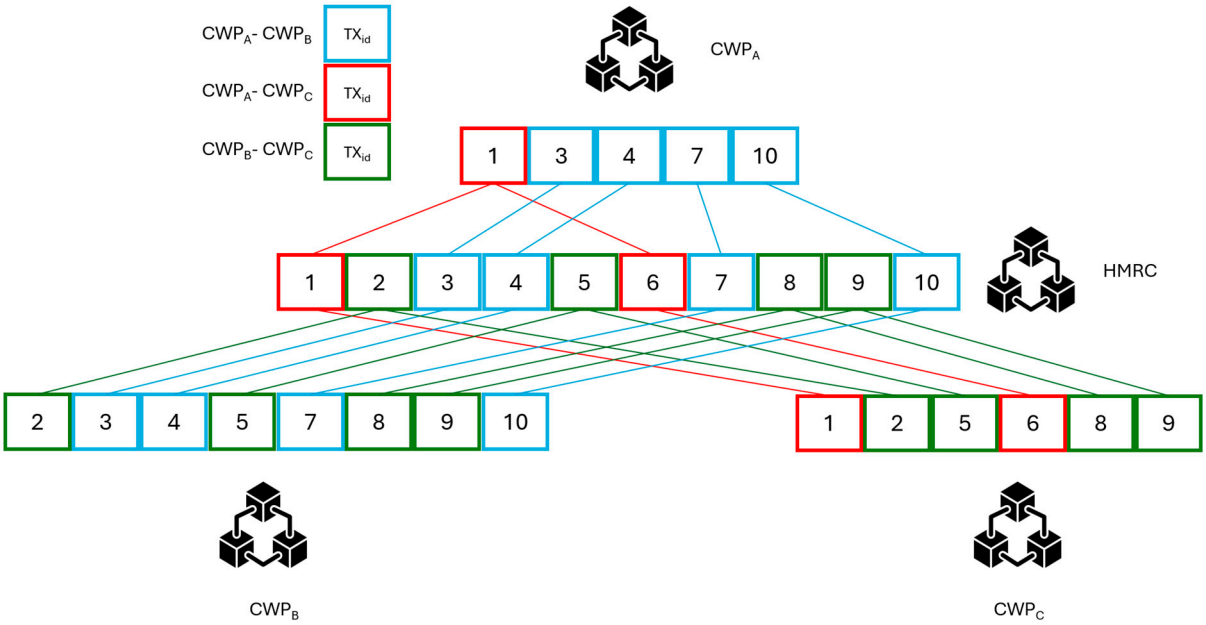

Figure 2. Corda "need-to-know" approach.

An event, such as issuing new money, transferring money, or submitting a data access request, represents an on-ledger event, or what is called a Corda state. This is the fundamental data block stored on the Corda ledger and constitutes an immutable record of a shared fact. When that fact has to be "updated", a new state is recorded, rendering the existing one as historic. The lifecycle of that fact over time is called a state sequence. The invoice state constructor in MTS is illustrated in Figure 3**.** In Smart Money, states represent paid and unpaid invoices, requested and accepted smart warrants, and money issued to end users. Corda contracts are rulesets that define how a Corda state can be updated to a new one. A contract is the "control" which ensures that a transaction has been fulfilled according to the pre-defined conditions. If the conditions in the contract are fulfilled, then the state update can take place and record itself on the ledger.

When a supplier issues an invoice under MTS, their CWP initiates an invoice issue flow which executes all the necessary steps for completing the transaction in Corda, such as creating a new state, verifying signatures from interested parties, validating the transaction against matching contracts, and allocating funds to accounts. The Corda DL platform provides a toolset used to create a native token for representing assets that can be exchanged among the network participants. Smart Money employs the Corda Token SDK for representing financial support that is provided by government departments as part of benefit schemes, e.g., the Universal Credit scheme offered in the UK [83]. These tokens are issued by government-owned Corda nodes to Corda special-purpose accounts which are hosted within CWP's (see Section 4.6). The nodes are owned by government departments for which social benefits are within their remit, e.g., the UK's Department of Work and Pensions. A beneficiary may hold both types of accounts: one for paying from a conventional wallet and one for paying from the token wallet. As this type of money is

provided as a benefit scheme, it can be programmed to be used only for essential purchases, such as utilities or food, ensuring responsible spending of funds and avoiding spending mistakes. An allowed goods list is prescribed for that purpose. A check against the allowed goods list is facilitated by the *InvoiceContractTk* contract; whenever a wallet holder pays an invoice with tokens, the contract checks if the items in the invoice are part of the allowed goods list. If the check fails, the payment fails as well. This functionality is a demonstration of how Smart Money employs programmable money for supporting governance.

```
constructor(val amount: Amount<Currency>,
    val issuedDate : LocalDateTime,
    val amountPaid: Amount<IssuedTokenType>,
    val paidFrom : Party?,
    val seller: AnonymousParty,
    val sellerPostCode :String,
    val buyer: AnonymousParty,
    val buyerPostCode : String,
    val hmrc: AnonymousParty,
    val typeOfGoods : String,
    val quantity : Int,
    val vatRate: Double,
    val vatPaid: Amount<Currency>,
    val paidDate : LocalDateTime?,
    val nettAmountReceived: Amount<Currency>,
    override val linearId: UniqueIdentifier)
```

Figure 3. Invoice state constructor.

4.3. MTS Experiment Participating Entities

A Smart Money experimental environment has been implemented for testing the feasibility of MTS. There are six participating nodes in the experimental network, each with a distinct role:

1. A Corda node which holds CBDC wallets for businesses that make sales, labelled SellerCWP.
2. A Corda node which holds CBDC wallets for citizens that make purchases, labelled BuyerCWP.
3. A Corda node which is owned by HMRC, labelled HMRCCWP.
4. A Corda node which is owned by a legal authority that authorises smart warrants, labelled LegalCWP.
5. A Corda notary node for validating transactions, labelled Notary.

For the purpose of this experiment, *SellerCWP* contains only wallets for entities that conduct sales, whilst *BuyerCWP* contains wallets for entities that conduct purchases. In reality, any CWP can host both types of wallets and facilitate any type of transaction for their customers. Moreover, each entity may participate in the network with more than one node in order to achieve high availability or boost performance under heavy workloads. The experimental scenario does not consider the communication exchange between the Smart Money system and the payments network; however, it employs a native digital currency (the CBDC) within Corda that is checked and updated for each transaction. When an invoice is issued or paid on the Corda ledger, the CWP (seller's or buyer's) handles the task of passing on messages between the two layers.

4.4. MTS Issuing and Paying an Invoice

In MTS, payments are initiated by the seller's CWP, who transmits a request-to-pay (R2P) message to the buyer's CWP when the corresponding invoice is generated. That message includes the enhanced data element, such as an invoice, as structured data. Enhanced data is a capability which is already being provided by payment processors [84] and is due to become standard in the UK in the years to come alongside R2P messaging [29],

using ISO20022 [28]. As soon as the seller's CWP issues the R2P message, it also initiates a transaction in the Smart Money DLT.

That transaction involves the preparation steps on behalf of the seller node such as generating one-time keys, creating the *SMInvoiceState*, signing the transaction, and opening a session with each interested party. The interested parties, i.e., *BuyerCWP* and HMRC, then run the transaction against the *InvoiceContract*, and if the conditions in the contract are fulfilled, they proceed with signing the transaction and notify *SellerCWP*. Then, the *SellerCWP* shares the finalised *SMInvoiceState* with HMRC and *BuyerCWP* and submits the fully signed transaction to the *Notary* node. The *Notary* validates the timestamp of the transaction, notifies the interested parties to update their ledgers with the new DL state, and returns a unique transaction ID for the completed payment. When the transaction is completed, the ledger contains a new state which specifies an unpaid invoice that is only visible to the interested parties. The functionality of the system with reference to issuing an invoice is presented using UML behavioural diagrams, i.e., activity, sequence, and state diagrams, in Figure 4.

To pay the issued invoice, the buyer responds to the R2P message through the Smart Money layer. The *BuyerCWP* informs their customer of the pending payment and invoice through their CBDC wallet user interface in the form of a smartphone or web banking app. The buyer has the chance to review the details of the payment and agree to the terms with the push of a button, which grants permission to the *BuyerCWP* to respond to the payment request. The *BuyerCWP* executes the necessary preparation steps such as generating keys, creating the new *SMInvoiceState*, and signing the transaction. This is followed by calculating the amounts to be paid, i.e., the net amount to the seller's account in the *SellerCWP* and the VAT amount to HMRC's account in the *HMRCCWP*. The flow then checks if the buyer's account has sufficient funds for fulfilling both parts of the payment and starts building the transaction for completing payment. At this stage, *BuyerCWP* creates the new state, signs the transaction, and initiates sessions with the counterparties, all of which check and verify the transaction against the invoice contract. If the conditions in the contract are fulfilled, they proceed with signing the transaction and notify *BuyerCWP*. Then, the *BuyerCWP* shares the finalised *SMInvoiceState* with *HMRCCWP* and *SellerCWP* and submits the fully signed transaction to the *Notary* node. The *Notary* finalises the transaction by running the same steps as for issuing an invoice. The functionality of the system with reference to paying an invoice is presented using UML behavioural diagrams, i.e., activity, sequence, and state diagrams, in Figure 5.

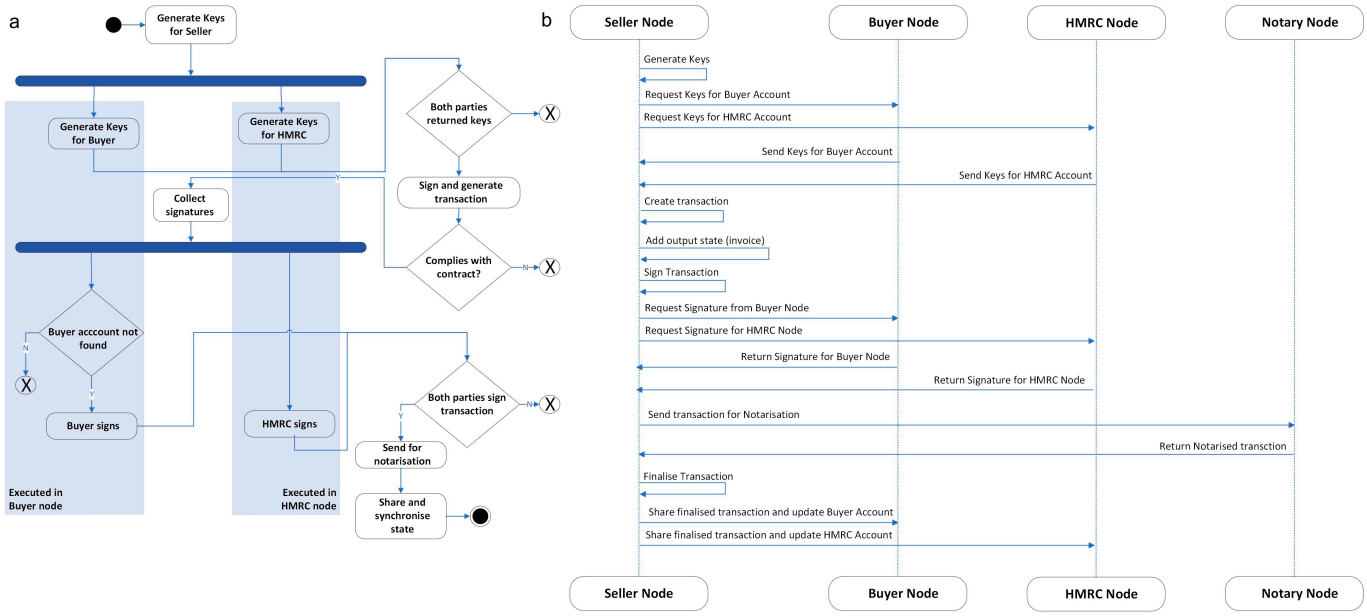

Figure 4. *Cont.*

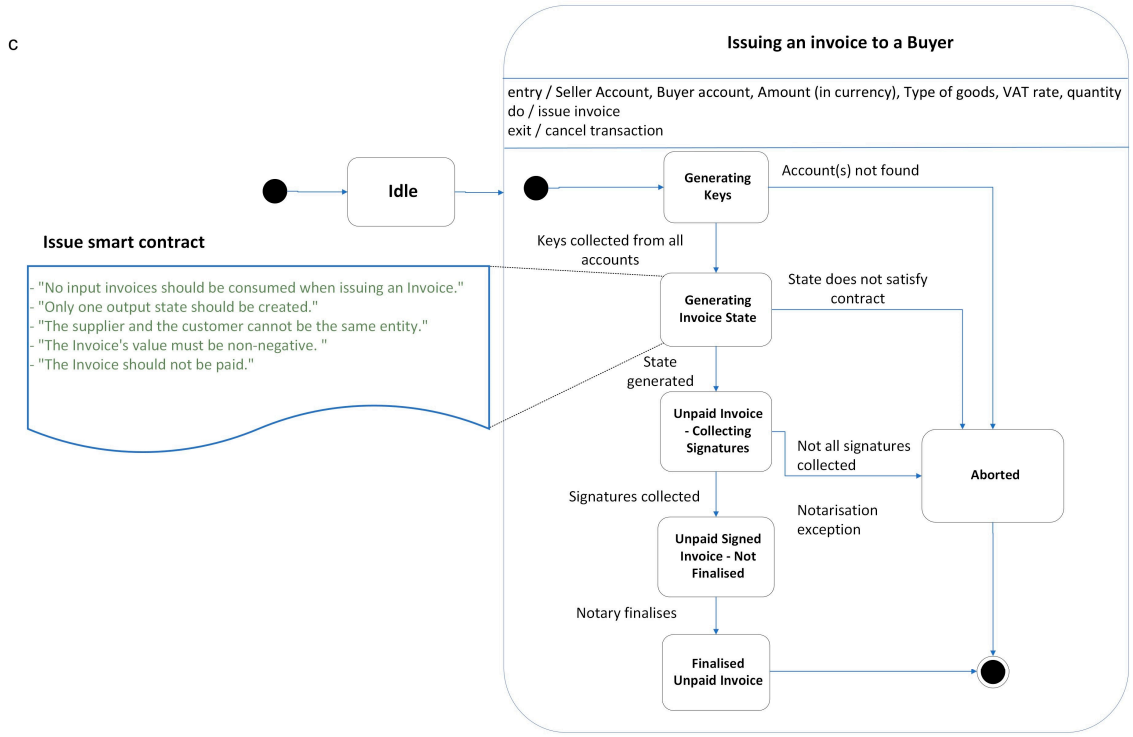

Figure 4. (**a**) Activity, (**b**) sequence, and (**c**) state diagrams for issuing an invoice.

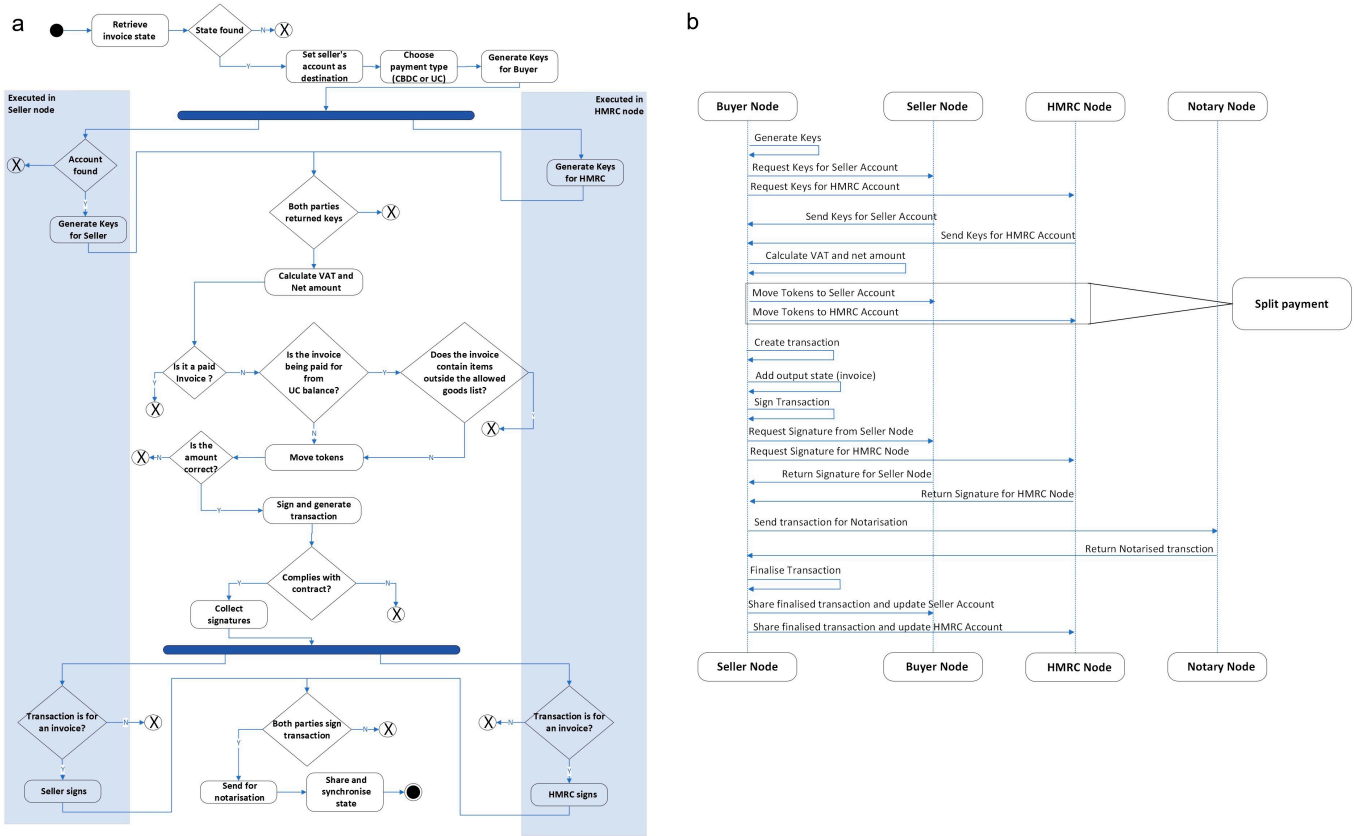

Figure 5. *Cont.*

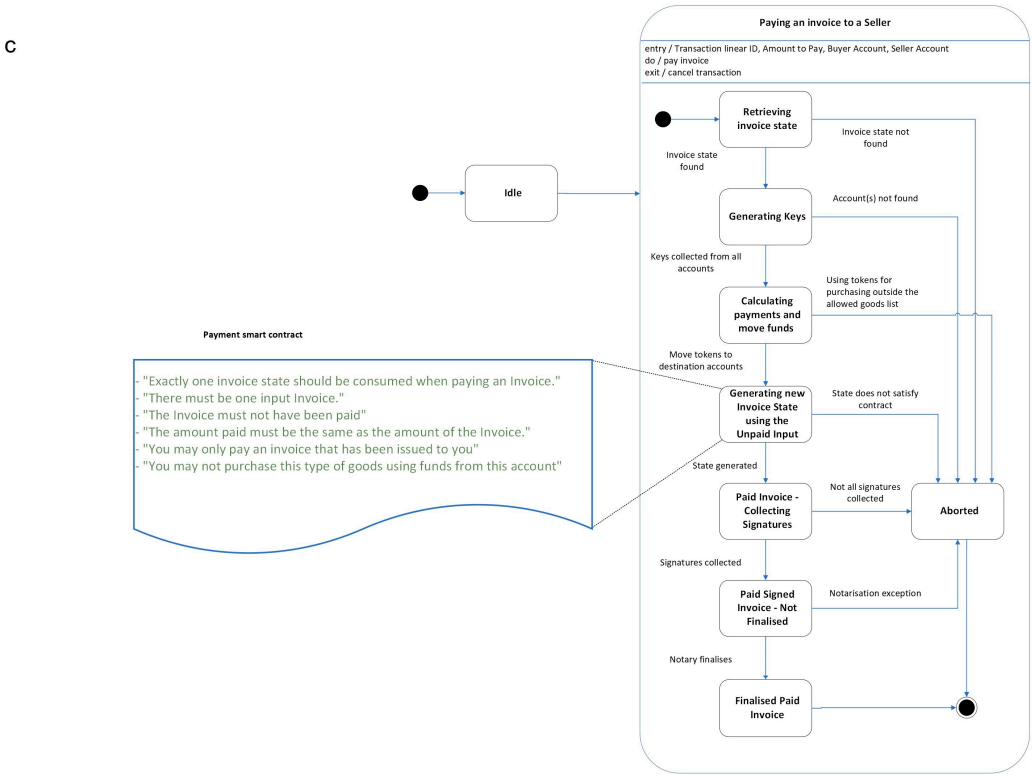

Figure 5. (**a**) Activity, sequence (**b**), and (**c**) state diagrams for paying an invoice.

4.5. Requesting and Executing Smart Warrants

The purpose of smart warrants is to serve government departments for conducting fraud investigations and analytics upon a CBDC wallet holder's transaction history when reasonable grounds for criminal activity on behalf of the wallet holder exist. Conventionally, when similar access requests are conducted, investigation is authorised by a legal authority, such as a magistrate. Smart Money incorporates a *LegalAuthority* node for that particular purpose: to review, authorise, and audit data access requests to payments data. In the experiment, smart warrants are employed by a special-purpose HMRC account named *VATInvestigator* for conducting investigations on the transactions of a CBDC wallet. The retrieved transactions are returned to the warrant requester account, but they are not stored on the ledger in a new state.

The smart warrant request transaction involves preparation steps on behalf of the HMRC node, such as generating one-time keys, creating the *DataAccessRequest* state, signing the transaction, and opening a session with the *LegalAuthority* node. *LegalAuthority* runs the transaction against the *DataAccessContract*. If the conditions in the contract are fulfilled, it signs the transaction and notifies *HMRCCWP*. The *VATInvestigator* account also signs the transaction and submits it to the *Notary* node for validation. At the last stage, the *Notary* notifies the parties involved to update their ledger copies and returns a transaction ID for the warrant request. The functionality of the system with reference to requesting data access is presented using UML behavioural diagrams, i.e., activity, sequence, and state diagrams, in Figure 6.

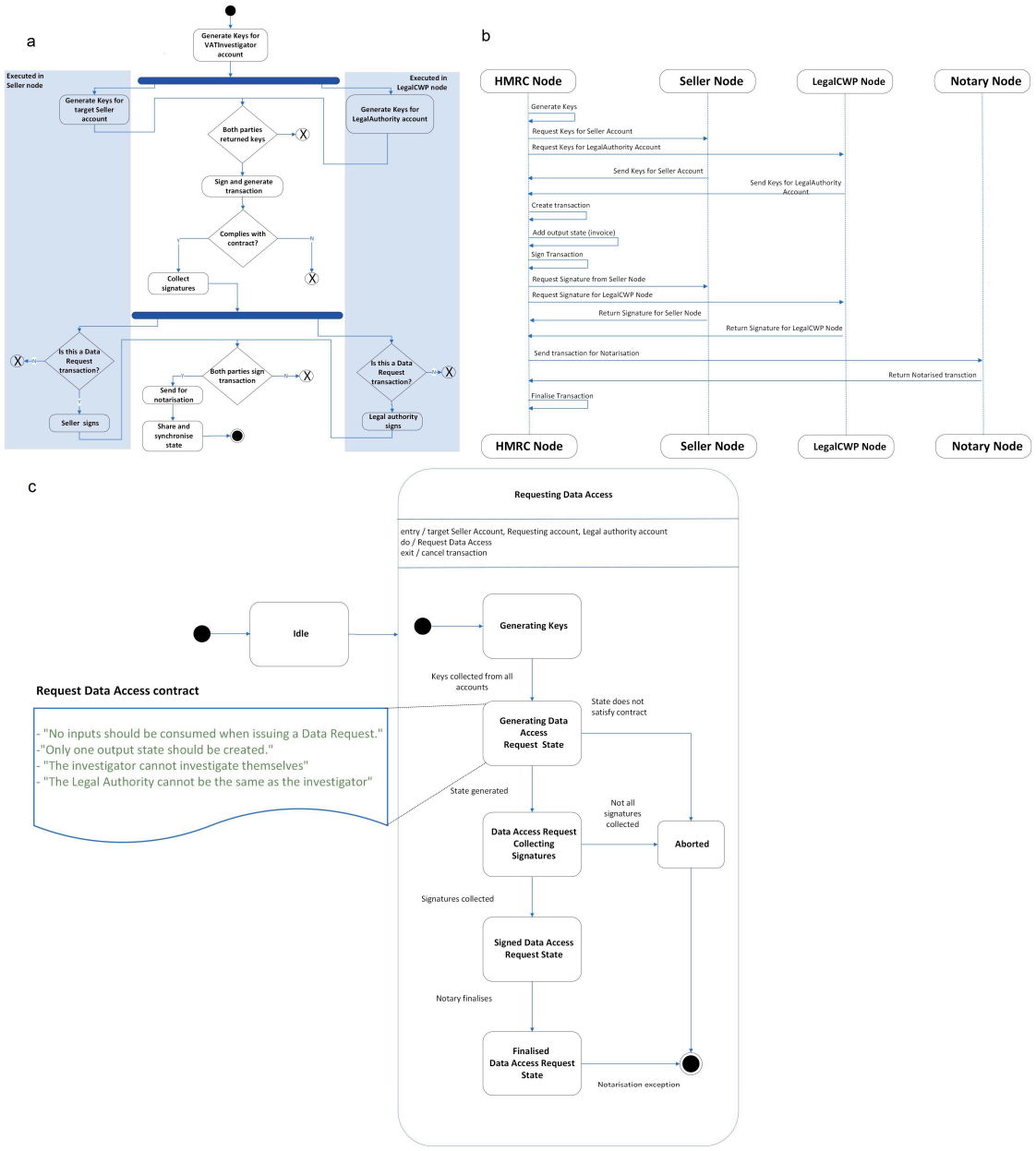

Figure 6. (**a**) Activity), (**b**) sequence, and (**c**) state diagrams for requesting data access.

The warrant requester can execute the warrant when they deem necessary. The smart warrant must be executed by the requester account, otherwise the *HMRC* node rejects the execution. The requester account must provide the ID and *LegalAuthority* that issued the warrant. If these fields do not match with an existing issued warrant, then the *HMRC* node rejects the execution. If the execution passes this test, then a new *DataAccessRequest* state is recorded which uses the existing *DataAccessRequest* state as input. This new state, which contains a timestamp of execution is appended to a transaction which is signed by HMRC and passed on to the *LegalAuthority* node for validation and signing. The *Notary* node is notified, retrieves the data from the investigated account's CWP ledger, returns the data on the requester's user interface, and updates the ledgers of HMRC and *LegalAuthority* with the new confirmed state. Smart warrants provide SupTech capabilities for governments, to ensure that investigations on personal data are performed lawfully and transparently. The functionality of the system with reference to executing a data access request is presented using UML behavioural diagrams, i.e., activity, sequence, and state diagrams, in Figure 7.

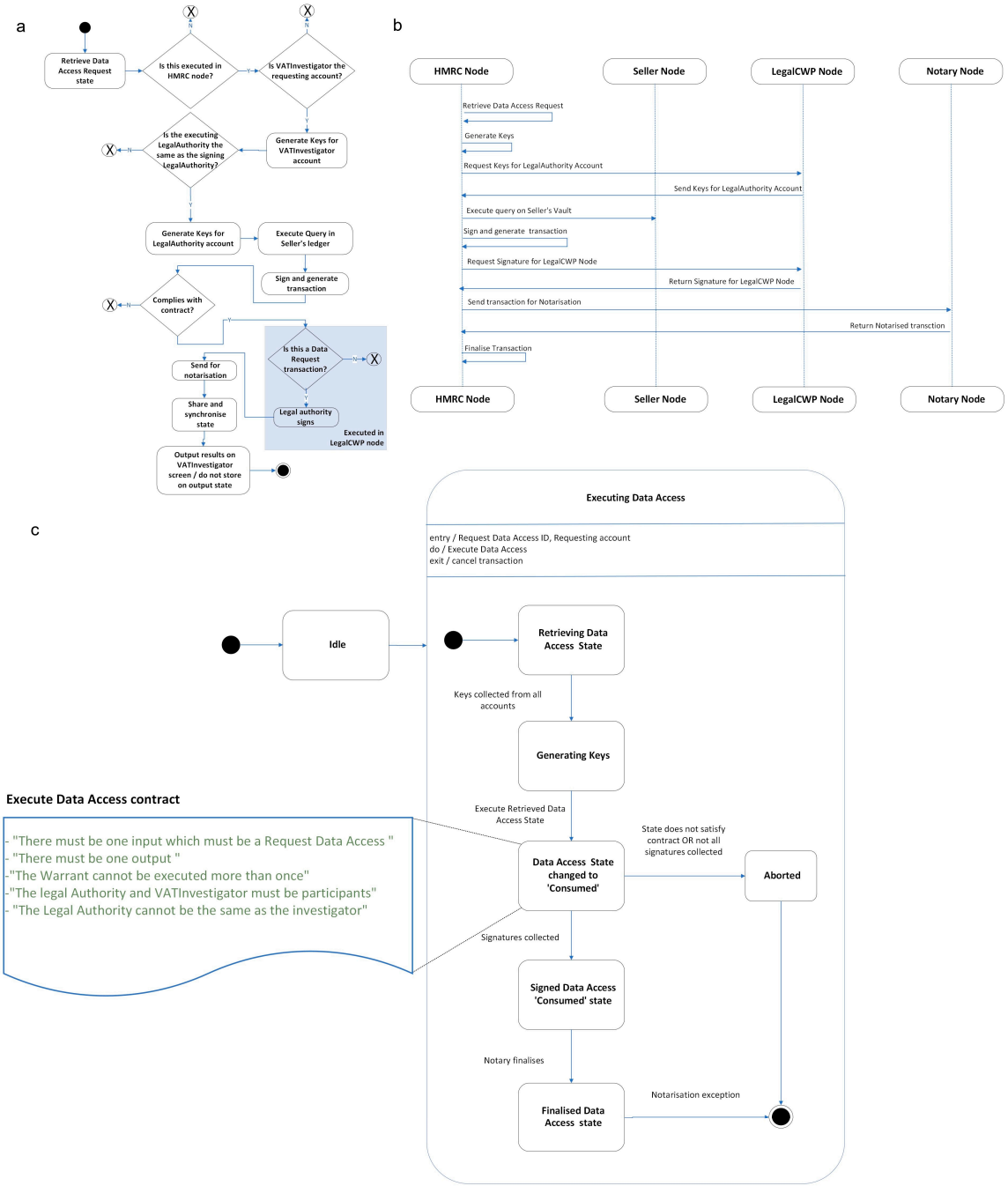

Figure 7. (**a**) Activity, (**b**) sequence, and (**c**) state diagrams for executing data access.

4.6. Issuing Programmable CBDC to Wallets

In addition to operating as an information management layer to the payment systems, the Smart Money DLT can facilitate payments with the use of a native CBDC which resides in the ledger. In this case, the exchange of funds takes place exclusively outside the payments network; however, the VAT for transactions is still paid automatically through MTS. For the purpose of the experiment, the Token SDK has been incorporated into Smart Money to implement financial support along the lines of the Universal Credit scheme [25] which is operating in the UK. This scheme, which is handled by the Department of Work and Pensions, is operated under the HMRC for the purpose of the experiment and it is controlled through the *HMRCCWP* node. The tokens issued by the HMRC can be used in transactions to pay for goods within an allowed goods list. To issue tokens to a wallet,

the HMRC node initiates an *IssueInvoice* flow which executes the necessary steps. This flow can only be executed by the *HMRCCWP*; hence, no other node can issue these tokens within the Smart Money DLT network. The flow assigns new tokens to the specified wallets sequentially through a session that is initiated between *HMRCCWP* and *BuyerCWP*. Effectively, HMRC instructs *BuyerCWP* to update the balance on the user's Corda account, i.e., the receiver's CBDC wallet. Similarly, this process can be executed between *HMRCCWP* and any CBDC wallet provider.

4.7. Security Considerations

While the Smart Money system builds upon the inherent security features of the R3 Corda platform, it is essential to address potential vulnerabilities in order to ensure a robust and trustworthy infrastructure. First, concerns about malicious users, criminals, or unfriendly governments participating in the network are mitigated by Corda's permissioned architecture. Participation is restricted to authenticated and authorized entities through stringent identity verification processes, including comprehensive know your customer (KYC) and anti-money laundering (AML) checks. Nevertheless, the Smart Money CBDC infrastructure would be built in accordance with established industry standards such as the ISO27001 [85], which provides a framework for an effective information security management system, and the National Institute of Standards and Technology (NIST) Cybersecurity Framework [86], which is a comprehensive set of guidelines and best practices to help organizations manage cybersecurity risks.

Moreover, the protection of end users against the theft of cryptographic keys is paramount. Implementing robust key management practices has been widely adopted in the banking industry, such as the use of hardware security modules (HSMs), encrypted key storage solutions, and enforcing multi-factor authentication, and can significantly reduce the risk of key compromise. The Smart Money CBDC will rely on existing standard best practices of the UK banking system for the security of cryptographic keys, such as ISO/IEC 27001 [85], ISO/IEC 19790 [87], FIPS 140-2 [88], FIPS 140-3 [89], and PCI DSS [90], and will adhere to national regulations and guidance from bodies like the Financial Conduct Authority [91].

Finally, the system must be resilient against well-known attack vectors like denial of service (DoS), man-in-the-middle (MiM), and side-channel attacks. Network-level security measures, including firewalls, intrusion detection systems, and rate-limiting mechanisms, can mitigate DoS attacks. Secure communication protocols with end-to-end encryption, such as TLS with mutual authentication, help prevent MiM attacks by ensuring data integrity and authenticity between parties. If hardware tokens are adopted for authorization and authentication of end users, side-channel attacks against point-of- sales (PoS) systems would be considered a security risk. To protect against side-channel attacks, especially those targeting cryptographic operations, the system should employ constant-time algorithms and avoid patterns that could leak sensitive information through timing or power analysis. Regular security assessments, code audits, and penetration testing are essential practices to identify and address vulnerabilities proactively, ensuring the ongoing security and integrity of the Smart Money system. Therefore, the Smart Money system employs the same cybersecurity practices, cryptographic protocols, and standards, indicated above, as the existing regulated banking system and can block known attack, thereby maintaining the same level of security.

5. Evaluation Methodology

5.1. Performance Factors and Indicators for DLT Feasibility and Scalability Issues

DLT-based applications for a multitude of domains are currently being researched, with a focus on demonstrating the feasibility and scalability of these solutions. DLTs, which are based on either permissoned or permissionless access models, are assessed with respect to how certain application requirements affect several performance metrics of the system.

The majority of feasibility studies on the application of DLTs are concerned with how several operating factors affect a number of performance metrics.

A literature review has been conducted which shows that the majority of studies emphasise the performance factors such as throughput and latency as there is a common trade-off between data integrity and processing speed for DLT solutions. Of particular interest is the capability of a DLT application to scale with increasing volumes of workload. Workload is assessed in terms of how many transactions are submitted to the ledger per unit time [92–101] as well as the size of the data which are processed for each transaction [95–97,100,102–105]. Other works study the effect of the employed consensus mechanism (proof-of- work, proof-of-stake, and Byzantine fault tolerance [95–99]). The number of participating entities can also have an effect in performance [95–97,99,100,102,103,105], as more nodes and users present a delimiting effect in transaction throughput [92–99,102,104,106], and latency [92–98,100–103]. It also has been found that when the number of transactions becomes too high, some of those transactions might fail. It is therefore useful to measure the loss rate as a performance metric [92–95]. In addition, security factors such as fault tolerance and probability of malicious modification also affect the performance of blockchain solutions such as throughput and latency [98,107]. Table 1 summarises the evaluation metrics for several DLT feasibility and scalability studies found in scientific literature.

Table 1. Factors and metrics in the literature.

	Factors					Performance Metrics		
Source	**Transaction Load**	**Data Volume**	**Consensus Mechanism (*)**	**Security (Fault Tolerance, Probability of Misuse) (*)**	**No of Clients**	**Throughput**	**Latency**	**Transaction Rejection Rate (*)**
[92–94]	X					X	X	X
[95–97]	X	X	X		X	X	X	
[98]	X		X	X		X	X	
[99]	X		X		X	X		
[100]	X	X			X		X	
[101]	X						X	
[102]		X			X	X	X	
[103]		X			X		X	
[104]		X				X		
[105]		X					X	
[106]					X	X		
[107]				X		X	X	

* denotes factors and metrics not considered in this study.

The study presented in this paper investigates the impact of the following factors on scalability: transaction load (*tx*), data volume (*vol*), and number of clients (*cl*). Transaction load is defined as the number of invoices that have been submitted for issue or payment in a run. Data volume refers to the number of items that are exchanged during the transaction and are included in the corresponding invoice. We refer to clients as the number of endpoints that communicate with one of the nodes participating in the Smart Money system and submit transactions to be issued or paid. Within the context of a CBDC, the term "endpoints" may refer to (a) end-user devices which hold an account holder's private key (plastic card, mobile phone) and interact directly with the CWP's Corda node or (b) a special-purpose "aggregator" server which is operated by a CWP and collects transaction requests from end-users and submits them to the Corda node in bulk. By varying each of the factors (which constitute the independent variables of the experiment), the effect on scalability can be measured and analysed. Regarding the effect of the consensus mechanism on scalability, Corda does not use a consensus system. Instead, it provides a reliable witness by means of *Notary* nodes which prevent double-spending [108]. Therefore, the impact of the consensus mechanism is not considered in the experiment. With reference to the performance metrics, results for the variable transaction rejection rate are not presented,

as Corda has consistently completed all the transactions throughout the tests conducted. Therefore, only throughput (*tr*) and latency (*la*) have been selected as the measurement variables for the tests. The performance factors and metrics are summarised in Table 2.

Table 2. Factors and metrics.

Factor	Description
Transaction load *tx*	The number of transactions submitted to the ledger
Data volume *vol*	The number of items on an invoice
Number of clients *cl*	The number of clients that interact with the ledger via a corresponded node
Metric	**Description**
Throughput *tr*	The number of successful transactions completed per unit time. Measured in transactions per second (TPS).
Latency *la*	The time delay between two events: (1) transaction issued and (2) transaction validated. Measured in seconds.

5.2. Experimental Design

The purpose of the experiment is to demonstrate that the Smart Money Corda DLT is a feasible and scalable design for automated VAT split payments with a central bank digital currency. We examine the efficiency of smart contracts to fulfil VAT split payments and programmable money to prove the feasibility of the design. We also measure the impacts to system performance by applying variable levels of stress (through the various factor levels) in order to demonstrate the design's scalability. To assess the effect from each individual factor, tests are conducted with varying values for one factor while the remaining two factors are held constant. This approach allows for evaluating each of the factors separately and understanding how, and to what extent, they affect performance. This is important for informing future iterations of the Smart Money CBDC as well as other, third-party CBDC implementations which are based on R3 Corda and adopt tax split payments.

To investigate the impact of automatic VAT payments (the experiment), the tests will be conducted twice: once with split payments enabled and once with split payments disabled. We consider scalability as the ability to support increasing workloads with reference to the chosen performance factors. Scalability is demonstrated when performance decreases linearly with workload, as in [109–112]. Scalability tests involved with the impact of the Universal Credit scheme have not been conducted. This is because the check for disallowed items in an invoice is made irrespective of the money instrument (account balance or Universal Credit tokens) used for the corresponding payment. Therefore, paying through Universal Credit has no impact on scalability.

5.3. Smart Money Corda DLT Environment

The feasibility and scalability of the Smart Money CBDC is evaluated in a controlled experimental environment which simulates a real-world scenario that involves payments occurring between counterparties as well as the information exchange among all the interested stakeholders. The tests have been conducted using Corda Open Source v4.3 as this version was available at the time of development, and subsequent versions do not provide improvements which impact on the objectives of the experiment. The simulation environment consists of a computer network where each of the Making Tax Smart use case entities participates by means of a Corda node. There are seven Corda nodes in total, which are executed as Java virtual machines from within an equal number of Windows 2012 virtual machines. The hypervisor is VMWare vSphere 6.0. The setup is running on a 16-core Intel Xeon E5-2460 v3 @ 2.6 GHz and a total of 256 GB of RAM, which is allocated to the hypervisor and VMs. Three of the nodes (*HMRCCWP*, *SellerCWP*, and *BuyerCWP*) are allocated more hardware resources (by means of CPU cores and RAM), as preliminary tests demonstrated that this produced performance gains since these nodes execute the most

resource-consuming flows. Moreover, the machines which host *BuyerCWP* and *SellerCWP* also host the RPC clients which generate payments (issue and pay), thus further increasing the resource requirements for those two machines. The remaining nodes, which execute less-demanding tasks, are allocated less hardware resources. Table 3 summarises the list of participating nodes, their description, and their hardware specifications within the Corda experimental network.

Table 3. Corda nodes in the experimental setup.

Node Name	Description	CPU Cores	RAM Size	IP Address
HMRCCWP	Operated by HMRC for validating payments and conducting investigations through smart warrants authorised by LegalCWP.	8	16 GB	192.168.1.110
SellerCWP	Operated by a CWP which hosts accounts of businesses which make sales.	8	64 GB	192.168.1.111
BuyerCWP	Operated by a CWP which hosts accounts of consumers which make purchases.	8	64 GB	192.168.1.112
LegalCWP	Operated by a legal authority for authorising smart warrants to HMRC	4	4 GB	192.168.1.113
Notary	Operated by a legal authority for authorising smart warrants to HMRC	4	4 GB	192.168.1.114

The full functionality of MTS has been implemented in a CorDapp. Each participant node contains the part of the CorDapp which is relevant to their purpose. Similarly, *BuyerCWP* and *SellerCWP* do not execute the code required for requesting smart warrants from node *LegalCWP*. The functionality of each node is presented in Table 4.

Table 4. Corda nodes.

Node Name	Functionality
HMRCCWP	Validates payments for invoices issued from accounts registered to SellerCWP and paid by accounts registered to BuyerCWP. Issues UC money (tokens) to accounts registered to BuyerCWP. Requests smart warrants from LegalCWP for accounts registered to SellerCWP. Executes smart warrants authorised by LegalCWP. Creates accounts to BuyerCWP, SellerCWP, and LegalCWP and shares those accounts with other nodes. Hosts VATPayments account (for payment validation). Hosts VATInvestigator account (for smart warrant execution)
BuyerCWP	Pays invoices on behalf of accounts registered to BuyerCWP which are issued by accounts registered to SellerCWP. Hosts buyer accounts which hold balances (in Current and UC accounts).
SellerCWP	Issues invoices on behalf of accounts registered to SellerCWP which are to be paid by accounts registered to BuyerCWP. Hosts seller accounts which hold balances (in Current accounts).
Legal CWP	Authorises smart warrants issued from the account registered to HMRCCWP.
Notary	Ensures uniqueness to transactions through validation.

Corda nodes are not configured to communicate when they are built from source, i.e., the IP addresses of the remaining nodes are not included in their configuration files. Corda's native network bootstrapper is used for updating the node configuration with the list of IPs of subsequent nodes. After that process is completed, the nodes are ready to be started up as JVMs. Users map to Corda accounts; therefore, any request submitted to a node (e.g., for issuing a payment, for paying an invoice, for executing a smart warrant, etc.) is made on behalf of an account. Accounts from different nodes interact with each other

only through the nodes they are registered on. An account's parent node is responsible for transmitting the message to the destination node which in turn, notifies the destination account. An RPC (remote procedure call) client application is used for interfacing a user/account with a node. That RPC is submitted to the target node through the network; hence, the user can reside in any network station that can communicate with the node through TCP/IP. For the purpose of this experiment, the users reside within the node they are registered on, i.e., requests for payment are submitted from clients executed within the same VM which hosts the *SellerCWP*. Similarly, payments are submitted from clients executed within the same VM which hosts the *BuyerCWP*.

6. Feasibility Study

The feasibility of the Smart Money system was evaluated using real-life scenarios where the transactions involve buyers who purchase goods from sellers and where HMRC utilises the Smart Money system for enforcing tax collection and compliance. There are seven classes of goods sold in these transactions, and each class incurs a VAT rate, i.e., adult clothing: 20%, alcohol: 20%, books: 0%, children's clothing: 0%, electrical: 20%, energy: 5%, groceries: 0%. A transaction involves the following entities:

- A seller named MegaCompany, which is a retailer that sells all the available classes of goods. This seller takes part in MTS through their account, which is held in a Corda node named *SellerCWP*. This node hosts Corda accounts for sellers.
- A buyer named Alice, which is a consumer that may purchase any of the available classes of goods. This buyer takes part in MTS through their account, which is held in a Corda node named *BuyerCWP*. This node hosts Corda accounts for buyers.
- HMRC, which collects VAT for all VAT-incurring transactions that take place through MTS. HMRC takes part in MTS through an account named VATPayments, which is held in a Corda node named *HMRCCWP*.
- The *Notary* node which, by design, participates in every transaction which takes place in R3 Corda for validation purposes.

Each core functionality of Smart Money is implemented using Corda flows (see Section 4). The feasibility is evaluated in the form of assertion tests [113] which check the correctness of the flows, i.e., whether the flows produce the correct results. The assertion tests presented in the remainder of this section verify that it is feasible to implement the MTS core functionality with distributed ledger technology.

6.1. Issuing and Paying an Invoice

This Section covers the scenarios for testing the feasibility of issuing and paying invoices with VAT split payments within MTS. These scenarios cover the cases where an invoice is being paid by an account which has or has not sufficient funds to cover the total value of the goods purchased.

In this test (see Algorithm A1), a payment is created when a seller (MegaCompany) issues an invoice to a buyer (Alice) (see row 8 in Algorithm A1). This invoice contains information which is found in a regular invoice, such as the class of goods purchased, the quantity of items per class, the net price, the VAT rate for each class of goods, and the total price for the invoice. In addition, each invoice contains the type of money that will be used for the payment, which can be one of two types: Current and Token. If the type is Current, then the invoice can only be paid with funds from Alice's 'regular' account. This is because certain item classes (e.g., alcohol) do not belong in the allowed goods (see Algorithm A1). For a complete list of fields, see the invoice state constructor in Figure 3.

After the invoice is issued, it is appended in the unpaid invoice queue for both Alice and MegaCompany. HMRC, via its *VATPayments* account, should not have visibility of any unpaid invoices; therefore, is not aware of the transaction between Alice and MegaCompany. The first assertion test (AT1) verifies that the (yet unpaid) transaction is only visible by the accounts Alice and MegaCompany (see rows 9–12 in Algorithm A1). The invoice is paid by Alice, which is followed by the next assertion test (AT2) which verifies that all

counterparties see the invoice as paid (rows 14–17 in Algorithm A1). The final assertion test (AT3), verifies that all account balances are now updated according to the transactions (payment for goods and VAT split payments; see rows 18–21 in Algorithm A1).

A subsequent test (see Algorithm A2) was conducted to verify that a transaction cannot be completed if Alice does not have sufficient funds in their account to pay for the invoice's total amount. In this case, the unpaid invoice is created first, and the payment fails with an error "insufficient funds available" when Alice tries to pay for the invoice. The invoice, according to Assertion Test 4 (AT4), must remain in the unpaid state (see rows 10–13 in Algorithm A2). The account balances for all counterparties should also remain unchanged (see rows 14–17 in Algorithm A2), which is verified by Assertion Test 5 (AT5).

The next series of assertion tests (as listed in Table 5) were conducted to verify that goods can be purchased by spending tokens. In this test, a different shopping list was used (see Algorithm A3), where only goods from the allowed goods list were to be purchased. According to the Assertion Test (AT6), the invoice state must change to paid (see Algorithm A3 rows 10–13). Assertion Test 7 (AT7) was conducted to check whether the balances were correct after the tokens were spent (see Algorithm A3 rows 14–17). One more test (see Algorithm A3) was conducted to verify that a transaction cannot be completed if Alice pays for goods that are not in the allowed goods list, e.g., alcohol, with money from their token account (see Figure A1 for the shopping list). In this case, the payment for the invoice fails with the error "you cannot pay for invalid goods with money from your token account". The invoice remains in the unpaid state (see rows 10–13 in Algorithm A4). The account balances for all counterparties should also remain unchanged (see rows 14–17 in Algorithm A4). The assertion tests returned positive results for all the test cases, thereby proving that the split VAT payment functionality is feasible with the MTS use case for the Smart Money system. This has been found to work properly for variable VAT rates in the same invoice, e.g., an invoice with goods of all three VAT rates (0%, 5%, and 20%).

Table 5. Assertion tests.

Test ID	Description	Test Result
AT1	Unpaid invoice is stored in MegaCompany ledger Unpaid invoice is stored in Alice's ledger Unpaid invoice is stored in Alice's ledger	True
AT2	Invoice state in MegaCompany's ledger is Paid Invoice state in Alice's ledger is Paid Invoice state in VATPayment's ledger is Paid MegaCompany's account balance is credited with Netamount	True
AT3	MegaCompany's account balance is credited with Netamount Alice's account balance is debited with Totalamount VATPayment's account balance is credited with VAT	True
AT4	Invoice state in MegaCompany's ledger is not Paid Invoice state in Alice's ledger is not Paid Invoice state in VATPayment's ledger is not Paid	True
AT5	MegaCompany's account balance remains unchanged Alice's account balance remains unchanged VATPayment's account balance remains unchanged	True
AT6	Invoice state in MegaCompany's ledger is Paid Invoice state in Alice's ledger is Paid Invoice state in VATPayment's ledger is Paid	True
AT7	MegaCompany's account balance is credited with Netamount Alice's account balance is debited with Totalamount VATPayment's account balance is credited with VAT	True

Table 5. *Cont.*

Test ID	Description	Test Result
AT8	Invoice state in MegaCompany's ledger is not Paid Invoice state in Alice's ledger is not Paid Invoice state in VATPayment's ledger is not Paid	True
AT9	MegaCompany's account balance remains unchanged Alice's account balance remains unchanged VATPayment's account balance remains unchanged	True
AT10	Signed DAR is created and unexecuted	True
AT11	Signed DAR is executed	True
AT12	Fetched data is MegaCompany's actual transactions	True
AT13	VATInvestigator can only query MegaCompany's transactions once	True

6.2. Requesting and Issuing Smart Warrants

This Section covers the scenarios for testing the feasibility of establishing controlled access to payments data for government agencies after authorisation by a legal authority. These scenarios cover the case where HMRC conducts an investigation into a retailer and requests access to payments that have been completed through MTS. In this test case (see Algorithm A5), a smart warrant is created when an investigator who works for HMRC (*VATInvestigator*) submits a data access request to a legal entity (*LegalAuthority*) so that they gain access to the payments data of a seller account (MegaCompany). The *LegalAuthority* signs the request (this process is conducted automatically in the feasibility tests) and returns a signed Data Access Request (DAR) which can only be executed by the *VATInvestigator* account and can only be used for accessing the payments data of MegaCompany's account. The first assertion test (AT10) verifies that the signed DAR has been created and is a validated, unexecuted request authorised by the *LegalAuthority* (see rows 5–7 in Algorithm A5). When the *VATInvestigator* executes the warrant, the warrant's state should first become executed (for the second assertion test, AT11, see rows 9–10 in Algorithm A5). This account can then execute the warrant, which runs a vault query on the MegaCompany's database of transactions and fetches all results on the *VATInvestigator*'s screen (see row 12 in Algorithm A5). This concludes the third assertion test (AT12). The final assertion test for smart warrants (AT13), verifies that the *VATInvestigator* account can only execute this query once (row 13 in Algorithm A5). The feasibility tests confirm that the functionality required for a CBDC with MTS capabilities is feasible using the Smart Money system, particularly implemented with the R3 Corda DLT.

6.3. Findings of Feasibility Study

The results presented in this Section have confirmed that designing and developing a CBDC with VAT split-payment capabilities is feasible using distributed ledger technology and smart contracts, particularly the R3 Corda open-source blockchain platform. It has also been shown that the Smart Money DLT provides the required capabilities for investigations on behalf of HMRC by way of smart warrants, with the necessary safeguards in place to assure citizens' rights to privacy. The Smart Money system can even act as an enabler of smart policies through a CBDC, where social services such as the UK's Universal Credit scheme can be offered more efficiently and transparently, which makes it more beneficial for citizens, government, as well as for businesses.

7. Scalability Study

This Section presents the results of the scalability study for the Smart Money CBDC which tests the effects of three factors on system performance. In particular, it has to be shown that the performance decrease is tractable, as the factors shift from lower values to higher values. There is a question about what the upper bound is for these factors within the context of this scalability study. The selection of the value ranges for each of the factors in the experiment takes into account hardware limitations of the employed testbed

and real-life use cases. Given that a single RPC client may utilise significant portions of RAM (between 128 MB and 1 GB), experimental runs employing more than 100 clients become infeasible, since the VM's RAM resources will be exhausted (see Section 5.3 for the use of RPC clients as transaction generators). It therefore applies that $10 < cl < 100$. In accordance with the basket of goods and services study published by the Office for National Statistics [114], the value 100 is used as the maximum number for the items per invoice performance factor, hence $10 < vol < 100$.

Transaction load (tx), i.e., the number of invoices issued and paid, is a factor which depends on the number of clients (cl) because the total number of invoices is equal to the number of clients multiplied by the number of invoices (issued or paid) per client. The effective upper bound for transaction load in the experiments was set to 10,000 invoices (e.g., 100 clients $\times$ 100 invoices per client and 10 clients $\times$ 1000 invoices per client) for all client transactions. Preliminary tests with higher transactions loads were conducted, and the results were consistent with those illustrated in subsequent sections. These results are omitted for the sake of brevity. Every data point in the results presented in Figures 8–10 represents an average of 10 runs for each test case. The experimental data can be found in Tables A1–A4.

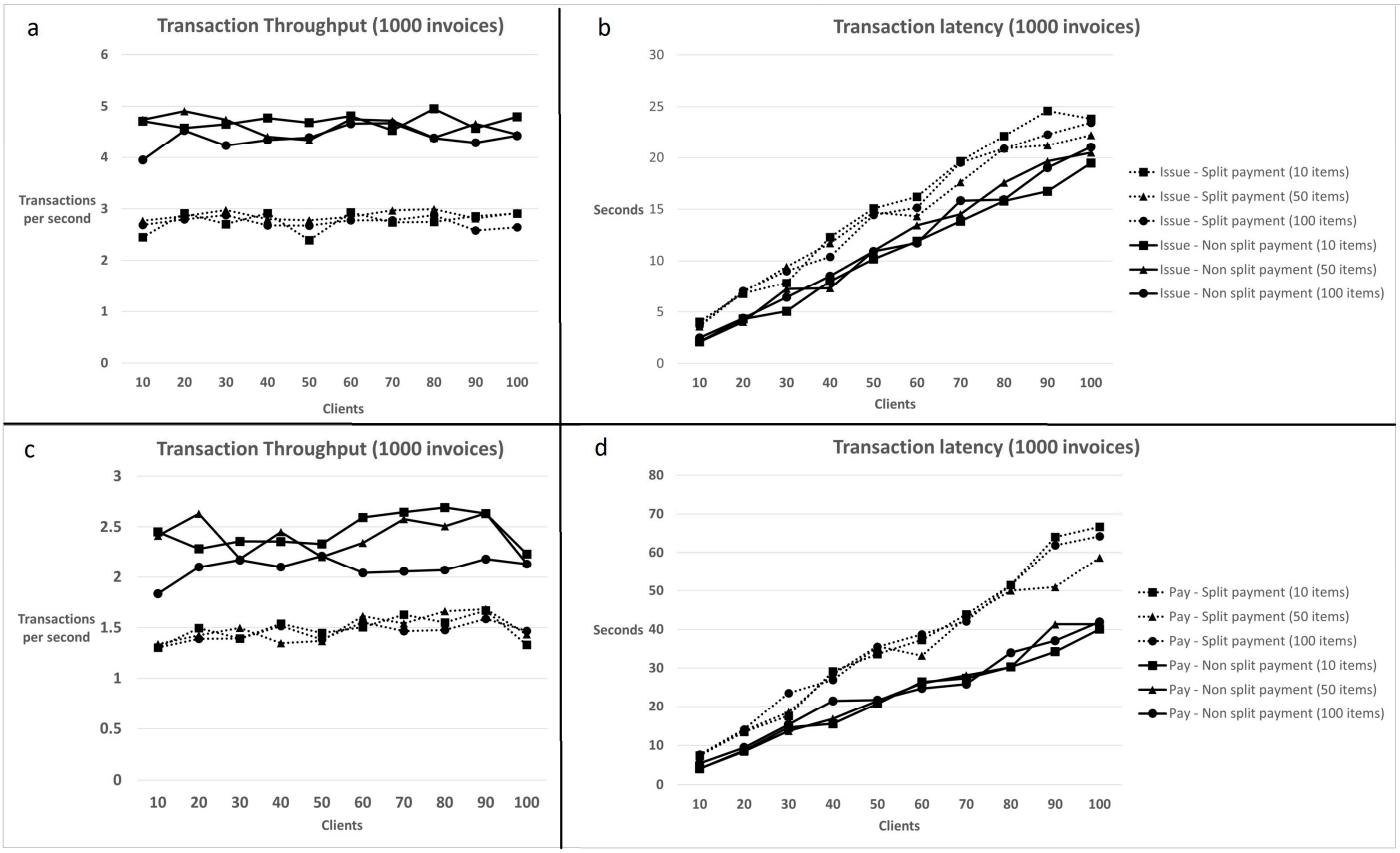

Figure 8. (**a**) Transaction throughput—issue [1000 invoices]; (**b**) transaction latency—issue [1000 invoices]; (**c**) transaction throughput—payment [1000 invoices]; (**d**) transaction latency—payment [1000 invoices].

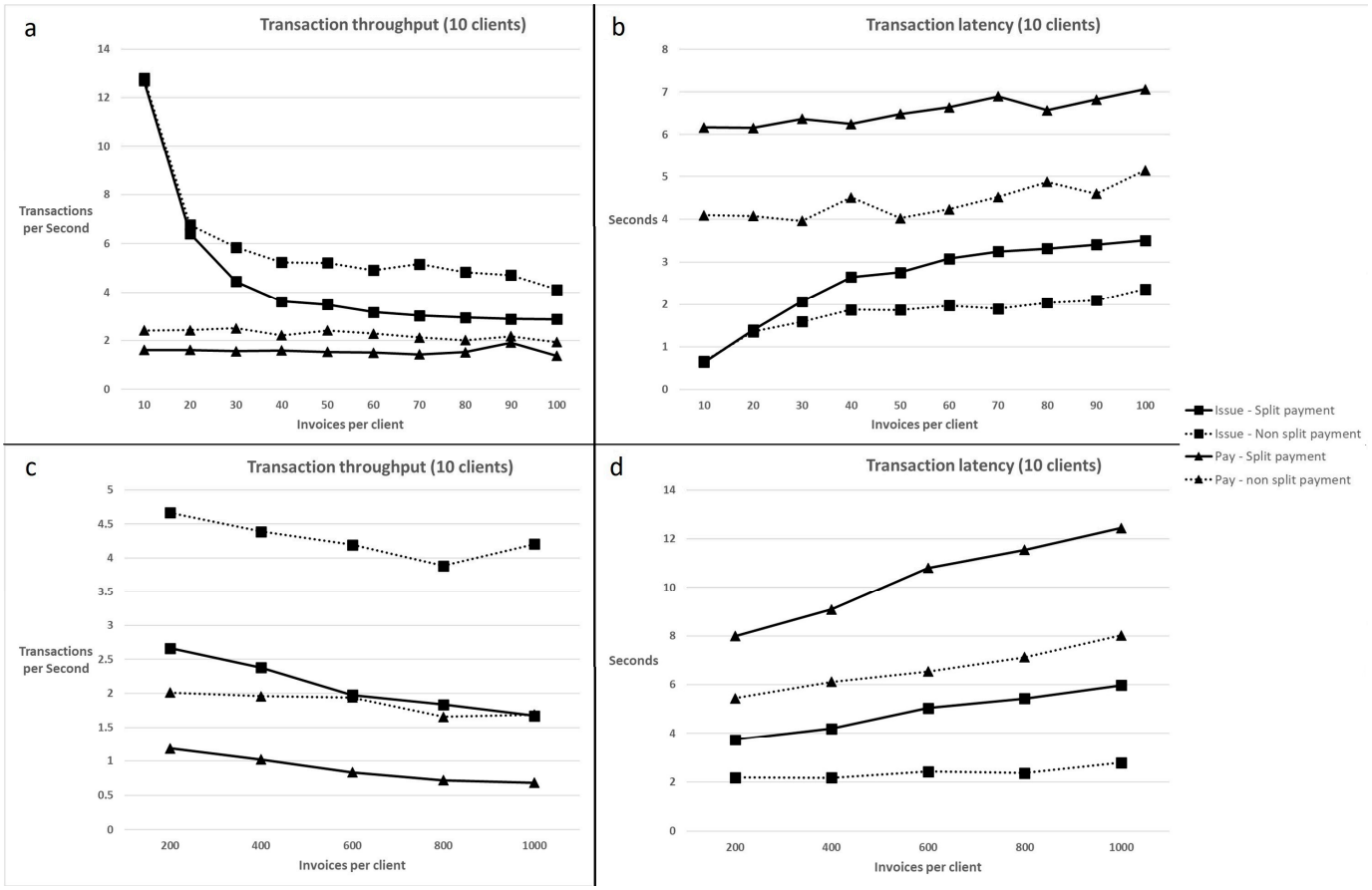

Figure 9. (**a**) Transaction throughput [10 clients]; (**b**) transaction latency [10 clients]; (**c**) transaction throughput [10 clients]; (**d**) transaction latency [10 clients].

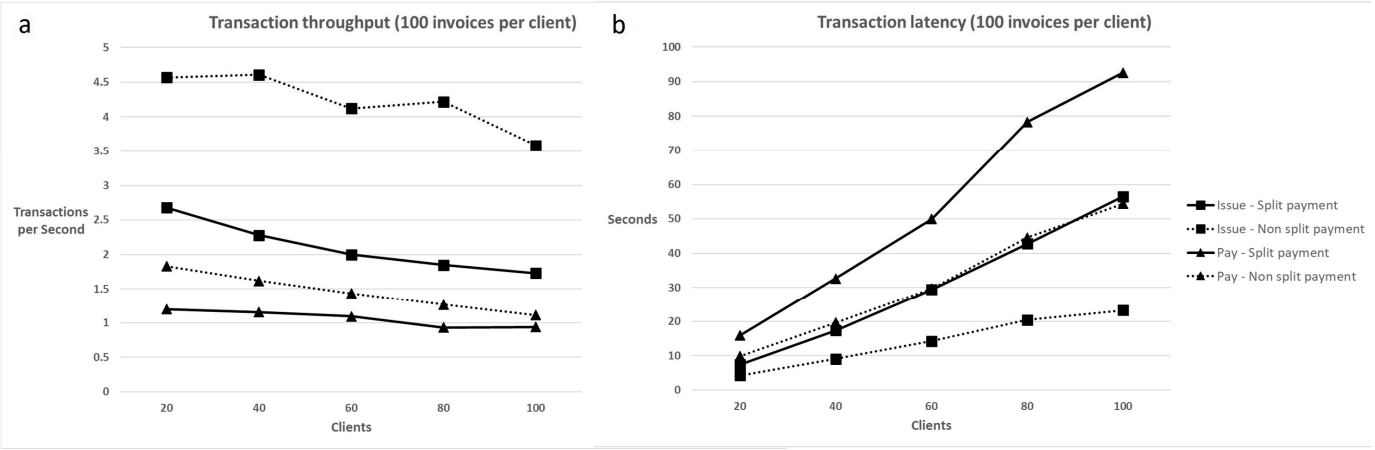

Figure 10. (**a**) Transaction throughput [100 invoices per client]; (**b**) transaction latency [100 invoices per client].

7.1. Impact of Data Volume and Number of Clients

A series of experiments was conducted to assess the impact of increasing data volume on the Smart Money DLT. A three-level scale of data volume with reference to the number of items included within an invoice was adopted, i.e., 10 items per invoice, 50 items per invoice, and 100 items per invoice. The total number of invoices was fixed to the value of 1000. The experiments were repeated for various numbers of clients [10, 20, . . ., 100], and the transaction load, i.e., the total number of invoices, was spread out amongst the clients so that the following Formula was satisfied:

$$cl \times tx \text{ per client} = 1000.$$

This design made it possible to evaluate the impact of increasing numbers of clients on the network in addition to the effects of increasing data volume. Furthermore, all tests were executed twice: one set of runs with split payments enabled and one set of runs without split payments, where the full amount was paid directly to the seller with no involvement from HMRC.

The results in Figure 8 show that the Smart Money DLT achieves similar performance for all data volume levels. By inspecting the families of curves (issue—split, issue—non-split, pay—split, and pay—non-split) there is no clear within-family separation, which shows that the number of items per invoice has no effect on performance when invoices are issued or paid. Additional findings can be made with regard to the number of clients in transactions. In Figure 8a, the curves follow almost a straight line, which shows that throughput is not affected by increasing the number of clients. In the case of payments, Figure 8c, throughput appears to slightly increase when the transaction load is spread out amongst more clients, which demonstrates high scalability. Transaction latency, Figure 8b,d, increases with the number of clients, albeit linearly. Overall, the results show that the Smart Money DLT demonstrates scalability when the data volume of transactions increases. Similar results are observed when increasing the number of clients in the network. The results also show that with split payments enabled, performance (in terms of transaction throughput and latency) drops by a factor of 1.5 to 2. This is caused by the extra computational burden required for calculating and transferring the VAT of a payment to HMRC for either issuing or paying an invoice.

7.2. Impact of Transaction Load

There are two ways to vary *tx*, i.e., (a) increase the number of invoices per client while keeping the client population constant or (b) increase the number of clients while keeping the number of invoices per client constant. A separate series of tests was conducted for each of the two approaches. The results from Section 7.1 confirm that the data volume has no impact on performance; therefore, it was deemed unnecessary to control this factor for the remainder of the experiments. The number of items per invoice was set to 10 and was held constant across all tests.

7.2.1. Varying Invoices per Client and Keeping Clients Constant

In this set of experiments, there were 10 clients issuing and paying invoices at the same time. The results illustrated in Figure 9a,b show that performance drops instantly from 10 to 20 invoices per client. From there, performance (in terms of throughput and latency) drops smoothly as the number of invoices increases. This shows that the Smart Money DLT (i.e., the MTS) scales appropriately with increasing numbers of payments (issued and paid) by holding the number of clients constant and increasing the number of transactions per client. This is also observed when the number of invoices increases further, from 200 to 1000 per client, see Figure 9c,d. Enabling VAT split payments impacts performance in this case as well, since transaction throughput with split payments is always lower with an associated higher latency.

7.2.2. Varying Number of Clients and Keeping Invoices per Client Constant

In this set of experiments, each client was configured to submit 100 invoices both for issue and payment. The results presented in Figure 10a,b show that throughput drops and latency increases linearly with increasing numbers of clients (and consequently, the total number of invoices). This is observed for all types of payments, either with or without split payment. It is therefore shown that the MTS is scalable for increasing transaction load when the increase is caused by adding more clients to the network.

7.2.3. Findings of the Scalability Study

The results of the scalability study have shown that the volume of data contained in an invoice has no discernible effect on the performance of the Smart Money system; therefore, the design and implementation demonstrate scalability. This finding indicates that there is no restriction as to which businesses can participate in this scheme as sellers with respect to the volume of goods that can be sold in a single transaction. For example, a shopping list in a supermarket is commonly larger compared to a shopping list from a clothing store. It also proves that MTS can be used not just for retail but also for wholesale transactions, where the quantity of goods exchanged can be much higher. Moreover, the Smart Money system can cope with increasing numbers of clients. The system can retain a certain level of transaction throughput, but the latency of each transaction increases with the number of clients, albeit linearly, and, therefore, in a scalable manner. This shows that MTS is applicable to existing payment systems which have to support a very big number of endpoints (e.g., point of sales systems).

From inspection of the results, it is observed that increasing the data volume generated by each client has much less of an effect compared to increasing the number of clients. Since clients are external and communicate via RPC calls, each additional client introduces more network communication overhead. RPC calls involve serialization/deserialization, network latency, and processing time on the server side, all of which contribute to increased latency. Also, as the number of clients increases, DLT nodes may experience resource contention (CPU, memory, and I/O). Handling more simultaneous RPC calls can lead to queuing delays and increased processing times per transaction.

The results also show that the Smart Money system scales well when the transaction load is increased, either when increasing the number of clients or when increasing the volume of transactions generated by each client. This is a requirement of modern payment systems which incorporate new forms of money such as CBDC, which will take up an even larger proportion of daily payments as physical cash and cheques are being abandoned. Issuing an invoice is faster than paying for an invoice. This is caused by the extra computational overhead for the additional calculations and necessary checks, e.g., check if it is an un-paid invoice, check if the funds required are available, check if this invoice can be paid (see Figures 4 and 5). These steps are executed serially and are mandated for a CBDC payment infrastructure; thus, they cannot be eliminated. However, a technical solution for mitigating the delays associated with paying an invoice compared to issuing, would be the parallelisation of the checks that are currently implemented serially. This parallelisation would be manifested within the Corda flow that implements the payment; however, this feature is not provided by the R3 Corda framework.

Overall, the design has been demonstrated to be scalable within the simulation environment and the constraints of its hardware as described in Section 5.3.

8. Conclusions

As CBDCs are being presented and experimented with by central banks across the globe, it becomes apparent that the existing CBDC blueprints do not fully exploit the opportunities offered by DLT and smart contracts to aid central banks and OGDs in overcoming governance frictions and achieving their objectives. The research presented herein is based on an approach which integrates tax compliance to a CBDC without disrupting the existing money system. The proposed Smart Money CBDC, which utilises

the R3 Corda DLT infrastructure, brings together for the first time the financial system and OGDs in a manner which is feasible, scalable, and practical for creating smart policies such as automated tax collection from payments and controlled data access for policing. From a computer science perspective, a novel function has been developed that uniquely triggers two payments concurrently as a single transaction on the distributed ledger. The inclusion of the programmable money concept in the Smart Money CBDC further amplifies the impact of the approach by extending the application to conditional purchases to support responsible spending of funds, hence preventing errors. These novel capabilities can only be realised with DLT-enabled smart contracts, which embody trust in multiparty agreements in a way that necessitates the revision of money's standard definition and its role in the economy. Money can now act as a policy sensor and policy actuator by mobilising enhanced data contained within the payment messages. Smart Money, which is an enabler of Smart Policy, is the first practical CBDC application of a DLT which promises new capabilities for government with benefits for the economy and society as a whole.

Author Contributions: P.L.: Conceptualization, Methodology, Investigation, Software, Writing—Original Draft, Writing—Review and Editing, Validation and Funding Acquisition. G.I.: Investigation, Methodology, Data Curation, Software, Writing—Original Draft, Writing—Review and Editing, Validation, and Visualization. G.W.: Investigation, Software, Writing—Review and Editing, and Validation. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the UK's Engineering and Physical Sciences Research Council [Grant number: EP/P032001/1]. Additional funding support was provided by EPSRC's Impact Acceleration Account (IAA).

Data Availability Statement: The scalability study dataset is available upon request by the authors.

Acknowledgments: The authors would like to thank project partners the Bank of England, Office for National Statistics, Ministry of Justice and His Majesty's Revenue and Customs for contributing to advisory board meetings and project workshops.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Assertion Tests

Algorithm A1. Pay invoice with sufficient funds

```
Require: Totalamount < abalance
Shoppinglist = {. . .} ▷ See Figure A1 for full list
Totalamount = Shoppinglist.amount
Netamount = Shoppinglist.netamount()
VAT = Shoppinglist.vatamount()
mcbalance = MegaCompany.getBalance()
abalance = Alice.getBalance()
vbalance = VATPayments.getBalance()
Inv = MegaCompany.issueOrder(Alice,Shoppinglist)
assert{▷ Assertion Test 1
Inv ∈ MegaCompany.getInvoices()
Inv ∈ Alice.getInvoices()
Inv < VATPayments.getInvoices()}
Alice.payInvoice(Inv)
assert{▷ Assertion Test 2
MegaCompany.getState(Inv) == isPaid
Alice.getState(Inv) == isPaid
VATPayments.getState(Inv) == isPaid}
assert{▷ Assertion Test 3
MegaCompany.getAccountbalance(v) == mcbalance + Netamount
Alice.getAccountbalance() == abalance − Totalamount
VATPayments.getAccountbalance() == vbalance + VAT}
```

Algorithm A2. Pay invoice with goods within the allowed goods list with tokens

Require: Totalamount > abalance
1: Shoppinglist = {. . .} ▷ See Figure A1 for full list
2: Totalamount = Shoppinglist.amount
3: Netamount = Shoppinglist.netamount()
4: VAT = Shoppinglist.vatamount()
5: mcbalance = MegaCompany.getBalance()
6: abalance = Alice.getBalance()
7: vbalance = VATPayments.getBalance()
8: Inv = MegaCompany.issueOrder(Alice,Shoppinglist)
9: Alice.payInvoice(Inv)
10: assert{▷ **Assertion Test 4**
11: MegaCompany.getState(Inv) ! = isPaid
12: Alice.getState(Inv) ! = isPaid
13: VATPayments.getState(Inv) ! = isPaid}
14: assert{▷ **Assertion Test 5**
15: MegaCompany.getAccountbalance(v) == mcbalance
16: Alice.getAccountbalance() == abalance
17: VATPayments.getAccountbalance() == vbalance}

Algorithm A3. Pay invoice with insufficient funds

Require: Totalamount < atokenbalance
1: Shoppinglist = {. . .} ▷ See Figure A2 for full list
2: Totalamount = Shoppinglist.amount
3: Netamount = Shoppinglist.netamount()
4: VAT = Shoppinglist.vatamount()
5: mcbalance = MegaCompany.getBalance()
6: atokenbalance = Alice.getTokenbalance()
7: vbalance = VATPayments.getBalance()
8: Inv = MegaCompany.issueOrder(Alice,Shoppinglist)
9: Alice.payInvoicewithTokens(Inv)
10: assert{▷ **Assertion Test 6**
11: MegaCompany.getState(Inv) == isPaid
12: Alice.getState(Inv) == isPaid
13: VATPayments.getState(Inv) == isPaid}
14: assert{▷ **Assertion Test 7**
15: MegaCompany.getAccountbalance(v) == mcbalance + Netamount
16: Alice.getAccountbalance() == atokenbalancebalance − Totalamount
17: VATPayments.getAccountbalance() == vbalance
+ VAT}

Algorithm A4. Pay invoice with goods outside the allowed goods list with tokens

Require: Totalamount > atokenbalance
1: Shoppinglist = {. . .} ▷ See Figure A1 for full list
2: Totalamount = Shoppinglist.amount
3: Netamount = Shoppinglist.netamount()
4: VAT = Shoppinglist.vatamount()
5: mcbalance = MegaCompany.getBalance()
6: atokenbalance = Alice.getTokenbalance()
7: vbalance = VATPayments.getBalance()
8: Inv = MegaCompany.issueOrder(Alice,Shoppinglist)
9: Alice.payInvoicewithTokens(Inv)
10: assert{▷ **Assertion Test 8**

Algotithm A4. *Cont.*

11: MegaCompany.getState(Inv) ! = isPaid
12: Alice.getState(Inv) ! = isPaid
13: VATPayments.getState(Inv) ! = isPaid}
14: assert{▷ **Assertion Test 9**
15: MegaCompany.getAccountbalance(v) == mcbalance
16: Alice.getAccountbalance() == abalance
17: VATPayments.getAccountbalance() == vbalance}

Algorithm A5. Data Access (Smart Warrant)

1: mcDAR = VATInvestigator.DAR(MegaCompany, LegalAuthority)
2: SignedmcDAR = LegalAuthority.signDAR(mcDAR, VATInvestigator)
3: assert{▷ **Assertion Test 10**
4: SignedmcDAR ∈ VATInvestigator.getDARs()
5: VATInvestigator.getState(SignedmcDAR) == unexecuted}
6: mcdata = VATInvestigator.executeDAR(SignedmcDAR)
7: assert{▷ **Assertion Test 11**
8: VATInvestigator.getState(SignedmcDAR) ! = unexecuted}
9: mcinvoices == vaultQuery(MegaCompany,invoices)
10: assert{
mcinvoices == mcdata} ▷ **Assertion Test 12**
12: assert{▷ **Assertion Test 13**
13: NULL == vaultQuery(MegaCompany,invoices)}

Appendix B. Shopping Lists


```
[
  {
    "amount": 60626,
    "buyer": "Alice",
    "shoppingList": [
      {
        "item": "Groceries",
        "price": 3627,
        "quantity": 1,
        "vatRate": 0
      },
      {
        "item": "Energy",
        "price": 8040,
        "quantity": 1,
        "vatRate": 5
      },
      {
        "item": "Groceries",
        "price": 923,
        "quantity": 1,
        "vatRate": 0
      },
      {
        "item": "Books",
        "price": 9781,
        "quantity": 1,
        "vatRate": 0
      },
      {
        "item": "Groceries",
        "price": 4620,
        "quantity": 1,
        "vatRate": 0
      },
      {
        "item": "Children's Clothing"
        "price": 9159,
        "quantity": 2,
        "vatRate": 0
      },
      {
        "item": "Alcohol",
        "price": 9448,
        "quantity": 1,
        "vatRate": 20
      },
      {
        "item": "Adult Clothing",
        "price": 835,
        "quantity": 1,
        "vatRate": 20
      }
    ],
    "whoAmI": "MegaCompany"
  }
]
```


Figure A1. Shopping List 1.

```
[
  {
    "amount": 26009,
    "buyer": "Alice",
    "shoppingList": [
      {
        "item": "Groceries",
        "price": 837,
        "quantity": 1,
        "vatRate": 0
      },
      {
        "item": "Groceries",
        "price": 1004,
        "quantity": 1,
        "vatRate": 0
      },
      {
        "item": "Energy",
        "price": 5090,
        "quantity": 1,
        "vatRate": 5
      },
      {
        "item": "Groceries",
        "price": 392,
        "quantity": 1,
        "vatRate": 0
      },
      {
        "item": "Books",
        "price": 3556,
        "quantity": 1,
        "vatRate": 0
      },
      {
        "item": "Groceries",
        "price": 8921,
        "quantity": 1,
        "vatRate": 0
      },
      {
        "item": "Children's Clothing",
        "price": 6209,
        "quantity": 2,
        "vatRate": 0
      }
    ],
    "whoAmI": "MegaCompany"
  }
]
```


Figure A2. Shopping List 2.

Appendix C. Experimental Data

Table A1. Throughput for split payment.

Throughput					
Issue			Pay		
10 Items	50 Items	100 Items	10 Items	50 Items	100 Items
2.45	2.77	2.69	1.30	1.34	1.30
2.91	2.86	2.80	1.49	1.42	1.39
2.71	2.78	2.58	1.39	1.50	1.39
2.51	2.60	2.78	1.44	1.35	1.42
2.39	2.78	2.68	1.45	1.37	1.38
2.63	2.85	2.78	1.60	1.61	1.45
2.94	2.67	2.68	1.63	1.64	1.56
2.55	2.79	2,58	1.65	1.46	1.48
2.66	3.02	2.78	1.47	1.68	1.59
2.91	2.92	2.65	1.33	1.43	1.47

Table A2. Latency for split payment.

Throughput					
Issue			Pay		
10 Items	50 Items	100 Items	10 Items	50 Items	100 Items
4.03	3.59	3.67	742	7.43	7.65
6.79	6.90	7.05	13.43	13.80	14.08
7.84	8.42	10.01	22.55	21.50	17.51
12.30	10.72	12.39	29.13	22.41	26.91
15.09	14.71	14.46	33.61	35.65	35.51
16.21	15.32	16.15	38.29	38.23	37.71
19.67	16.62	17.53	50.87	41.14	47.10
20.12	18.92	21.91	52.46	49.05	52.50
22.58	21.22	22.29	60.04	53.97	58.90
23.81	22.22	23.44	66.71	58.67	64.25

Table A3. Throughput for non-split payment.

Throughput					
Issue			Pay		
10 Items	50 Items	100 Items	10 Items	50 Items	100 Items
4.71	4.74	3.95	2.45	2.41	1.84
4.58	4.91	4.53	2.28	2.63	2.10
4.45	4.54	4.53	2.16	2.18	2.17
4.67	4.61	4.64	2.45	2.25	2.20
4.68	4.34	4.39	2.33	2.20	2.21
4.51	4.55	4.56	2.39	2.44	2.14
4.53	4.32	4.27	2.65	2.48	2.06
4.75	4.49	4.28	2.49	2.41	2.27
4.87	4.36	4.49	2.53	2.53	2.28
4.80	4.45	4.43	2.23	2.14	2.13

Table A4. Latency for non-split payment.

Throughput					
Issue			Pay		
10 Items	50 Items	100 Items	10 Items	50 Items	100 Items
2.10	2.08	2.48	4.06	4.13	5.45
4.31	4.04	4.38	8.70	8.45	9.45
7.06	5.24	7.42	10.58	14.68	15.33
7.00	9.31	9.52	17.53	16.79	13.46
10.17	10.97	10.92	20.71	21.50	21.69
11.90	12.45	11.73	27.41	25.01	21.72
14.85	13.52	13.83	30.35	29.11	27.85
15.79	16.58	15.93	32.29	29.22	30.97
17.74	19.65	20.03	33.24	33.33	36.10
19.48	20.50	21.05	40,07	41.37	41.93

References

1. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. *SSRN* **2008**, 1–9. [CrossRef]
2. Hayes, A.S. Cryptocurrency Value Formation: An Empirical Study Leading to a Cost of Production Model for Valuing Bitcoin. *Telemat. Inform.* **2017**, *34*, 1308–1321. [CrossRef]
3. Eyal, I. Blockchain Technology: Transforming Libertarian Cryptocurrency Dreams to Finance and Banking Realities. *Computer* **2017**, *50*, 38–49. [CrossRef]
4. Fry, J.; Cheah, E.-T. Negative Bubbles and Shocks in Cryptocurrency Markets. *Int. Rev. Financ. Anal.* **2016**, *47*, 343–352. [CrossRef]
5. Ammous, S. Can Cryptocurrencies Fulfil the Functions of Money? *Q. Rev. Econ. Finance* **2018**, *70*, 38–51. [CrossRef]
6. Vandezande, N. Virtual Currencies under EU Anti-Money Laundering Law. *Comput. Law Secur. Rev.* **2017**, *33*, 341–353. [CrossRef]
7. Dai, M.; Zhang, S.; Wang, H.; Jin, S. A Low Storage Room Requirement Framework for Distributed Ledger in Blockchain. *IEEE Access* **2018**, *6*, 22970–22975. [CrossRef]
8. Hawlitschek, F.; Notheisen, B.; Teubner, T. The Limits of Trust-Free Systems: A Literature Review on Blockchain Technology and Trust in the Sharing Economy. *Electron. Commer. Res. Appl.* **2018**, *29*, 50–63. [CrossRef]
9. Yeow, K.; Gani, A.; Ahmad, R.W.; Rodrigues, J.J.P.C.; Ko, K. Decentralized Consensus for Edge-Centric Internet of Things: A Review, Taxonomy, and Research Issues. *IEEE Access* **2017**, *6*, 1513–1524. [CrossRef]
10. Sullivan, C.; Burger, E. E-Residency and Blockchain. *Comput. Law Secur. Rev.* **2017**, *33*, 470–481. [CrossRef]
11. Ho, G.T.S.; Tang, Y.M.; Tsang, K.Y.; Tang, V.; Chau, K.Y. A Blockchain-Based System to Enhance Aircraft Parts Traceability and Trackability for Inventory Management. *Expert Syst. Appl.* **2021**, *179*, 115101. [CrossRef]
12. Hu, J.; Zhu, P.; Qi, Y.; Zhu, Q.; Li, X. A Patent Registration and Trading System Based on Blockchain. *Expert Syst. Appl.* **2022**, *201*, 117094. [CrossRef]
13. Muzumdar, A.; Modi, C.; Madhu, G.M.; Vyjayanthi, C. A Trustworthy and Incentivized Smart Grid Energy Trading Framework Using Distributed Ledger and Smart Contracts. *J. Netw. Comput. Appl.* **2021**, *183–184*, 103074. [CrossRef]
14. Collomosse, J.; Bui, T.; Brown, A.; Sheridan, J.; Green, A.; Bell, M.; Fawcett, J.; Higgins, J.; Thereaux, O. Archangel: Trusted Archives of Digital Public Documents. In Proceedings of the ACM Symposium on Document Engineering 2018, Halifax, NS, Canada, 28–31 August 2018; DocEng 2018. Association for Computing Machinery, Inc.: New York, NY, USA, 2018; pp. 1–4.

15. van Engelenburg, S.; Janssen, M.; Klievink, B. Design of a Software Architecture Supporting Business-to-Government Information Sharing to Improve Public Safety and Security: Combining Business Rules, Events and Blockchain Technology. *J. Intell. Inf. Syst.* **2019**, *52*, 595–618. [CrossRef]
16. Olnes, S.; Ubacht, J.; Janssen, M. Blockchain in Government: Benefits and Implications of Distributed Ledger Technology for Information Sharing. *Gov. Inf. Q.* **2017**, *34*, 355–364. [CrossRef]
17. Koning, J.P. *Fedcoin: A Central Bank-Issued Cryptocurrency*. R3 Reports. 2016. Available online: https://www.r3.com/wp-content/uploads/2018/04/Fedcoin_Central_Bank_R3.pdf (accessed on 11 September 2024).
18. Bech, M.; Garratt, R. Central Bank Cryptocurrencies. In *BIS Quarterly. Review*; Bank of International Settlements: Basel, Switzerland, 2017; pp. 55–70, ISBN 1683-013X.
19. Deutsche Bundesbank. *Distributed Ledger Technologies in Payments and Securities Settlement: Potential and Risks*; Deutsche Bundesbank: Frankfurt, Germany, 2017; pp. 35–49.
20. Engert, W.; Fung, B.S.C. *Central Bank Digital Currency: Motivations and Implications*; Staff Discussion Paper; Bank of Canada: Ottawa, OT, Canada, 2017.
21. Qian, Y. Central Bank Digital Currency: Optimization of the Currency System and Its Issuance Design. *China Econ. J.* **2019**, *12*, 1–15. [CrossRef]
22. Bank of Canada. Contingency Planning for a Central Bank Digital Currency. Digital Currencies Fintech. 2020. Available online: https://www.bankofcanada.ca/2020/02/contingency-planning-central-bank-digital-currency/ (accessed on 20 January 2020).
23. Sveriges Riksbank, S. *E-Krona Pilot Phase 1*; Sveriges Riksbank: Stockholm, Sweden, 2021.
24. Bank of England. *Central Bank Digital Currency Opportunities, Challenges and Design*; Bank of England: London, UK, 2020.
25. Royston, S. Understanding Universal Credit. *J. Poverty Soc. Justice* **2012**, *20*, 69–86. [CrossRef]
26. Romanova, I.; Grima, S.; Spiteri, J.; Kudinska, M. The Payment Services Directive II and Competitiveness: The Perspective of European Fintech Companies. *Eur. Res. Stud. J.* **2018**, *21*, 3–22. [CrossRef]
27. Wang, H.; Ma, S.; Dai, H.N.; Imran, M.; Wang, T. Blockchain-Based Data Privacy Management with Nudge Theory in Open Banking. *Future Gener. Comput. Syst.* **2019**, *110*, 812–823. [CrossRef]
28. *ISO 20022*; Universal Financial Industry Message Scheme. International Organisation for Standardization: Geneva, Switzerland, 2020.
29. Pay.UK. New Payments Architecture Programme. Available online: https://www.wearepay.uk/npa/ (accessed on 17 September 2020).
30. Society for Worldwide Interbank Financial Telecommunications SWIFT GPI 2022. Available online: https://www2.swift.com/knowledgecentre/rest/v1/publications/s_comp_app_gpi_fnc_instit_lbl_crtria_2022/1.0/s_comp_app_gpi_fnc_instit_lbl_crtria_2022.pdf?logDownload=true (accessed on 5 February 2022).
31. Bank of England. *New Forms of Digital Money*; Discussion Paper; Bank of England: London, UK, 2021.
32. Rahman, A.J. Deflationary Policy under Digital and Fiat Currency Competition. *Res. Econ.* **2018**, *72*, 171–180. [CrossRef]
33. Sompolinsky, Y.; Zohar, A. Bitcoin's Underlying Incentives. *Commun. ACM* **2018**, *61*, 46–53. [CrossRef]
34. Paech, P. The Governance of Blockchain Financial Networks. *Mod. Law Rev.* **2017**, *80*, 1073–1110. [CrossRef]
35. Khan, C.; Lewis, A.; Rutland, E.; Wan, C.; Rutter, K.; Thompson, C. A Distributed-Ledger Consortium Model for Collaborative Innovation. *Computer* **2017**, *50*, 29–37. [CrossRef]
36. Vukolic, M. Rethinking Permissioned Blockchains. In Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts—BCC '17, Abu Dhabi, United Arab Emirates, 2 April 2017; pp. 3–7.
37. Armknecht, F.; Karame, G.O.; Mandal, A.; Youssef, F.; Zenner, E. Ripple: Overview and Outlook. In *Trust and Trustworthy Computing, Proceedings of the 8th International Conference, TRUST 2015, Heraklion, Greece, 24–26 August 2015*; Conti, M., Schunter, M., Askoxylakis, I., Eds.; Springer International Publishing: Berlin/Heidelberg, Germany, 2015; Volume 9229, pp. 163–180.
38. Niranjanamurthy, M.; Nithya, B.N.; Jagannatha, S. Analysis of Blockchain Technology: Pros, Cons and SWOT. *Clust. Comput.* **2018**, *22*, 14743–14757. [CrossRef]
39. Sankar, L.S.; Sindhu, M.; Sethumadhavan, M. Survey of Consensus Protocols on Blockchain Applications. In Proceedings of the 2017 4th International Conference on Advanced Computing and Communication Systems ICACCS, Coimbatore, India, 6–7 January 2017; pp. 1–5. [CrossRef]
40. Lee, J.-H. BIDaaS: Blockchain Based ID As a Service. *IEEE Access* **2018**, *6*, 2274–2278. [CrossRef]
41. Aggarwal, S.; Chaudhary, R.; Aujla, G.S.; Kumar, N.; Choo, K.-K.R.; Zomaya, A.Y. Blockchain for Smart Communities: Applications, Challenges and Opportunities. *J. Netw. Comput. Appl.* **2019**, *144*, 13–48. [CrossRef]
42. Dyson, B.; Hodgson, G. *Digital Cash: Why Central Banks Should Start Issuing Electronic Money*; Positive Money: Broomfield, CO, USA, 2016.
43. Grym, A.; Heikkinen, P.; Kauko, K.; Takala, K. Central Bank Digital Currency. *BoF Econ. Rev.* **2017**, *4*, 5.
44. Sveriges Riksbank. *The Riksbank's e-Krona Project: Report 1*; Sveriges Riksbank: Stockholm, Sweden, 2017; p. 44.
45. Prasad, E. *Central Banking in a Digital Age: Stock-Taking and Preliminary Thoughts*; Hutchins Center on FIscal & Monetary Policy at BROOKINGS: Washington, DC, USA, 2018; pp. 1–52.
46. Ahmat, N.; Bashir, S. *Central Bank Digital Currency: A Monetary Policy Perspective*; Central Bank of Malaysia: Kuala Lumpur, Malaysia, 2017.
47. Meaning, J.; Dyson, B.; Barker, J.; Clayton, E. *Broadening Narrow Money: Monetary Policy with a Central Bank Digital Currency*; Staff Working Paper No. 724; Bank of England: London, UK, 2018.

48. Barrdear, J.; Kumhof, M. *The Macroeconomics of Central Bank Issued Digital Currencies*; Staff Working Paper No. 605; Bank of England: London, UK, 2016.
49. Alfonso, V.; Kamin, S.; Zampolli, F. *Central Bank Digital Currencies (CBDCs) in Latin America and the Caribbean*; BIS Working Paper 989; Bank for International Settlements: Basel, Switzerland, 2022.
50. Bank of England. The Digital Pound. Available online: https://www.bankofengland.co.uk/the-digital-pound (accessed on 7 December 2023).
51. Shiona McCallum. *Digital Pound Plans Should Proceed with Caution, Say MPs*. BBC.COM 2023. Available online: https://www.bbc.com/news/technology-67590468 (accessed on 5 February 2023).
52. European Central Bank. Exploring Anonymity in Central Bank Digital Currencies. *Focus* **2019**, *4*, 1–11.
53. European Central Bank. Eurosystem Launches Digital Euro Project. Press Release. 2021, pp. 1–2. Available online: https://www.ecb.europa.eu/press/pr/date/2021/html/ecb.pr210714~d99198ea23.en.html (accessed on 5 February 2023).
54. BIS Innovation Hub. Project Rosalind: Developing Prototypes for an Application Programming Interface to Distribute Retail CBDC 2022. Available online: https://www.bis.org/about/bisih/topics/cbdc/rosalind.htm (accessed on 10 August 2023).
55. BIS Innovation Hub. Project Icebreaker: Central Banks of Israel, Norway and Sweden Team up with the BIS to Explore Retail CBDC for International Payments. 2022. Available online: https://www.bis.org/about/bisih/topics/cbdc/icebreaker.htm (accessed on 10 August 2023).
56. BIS Innovation Hub. Project mBridge: Connecting Economies Through CBDC. 2022. Available online: https://www.bis.org/publ/othp59.htm (accessed on 10 August 2023).
57. Bank of England. A New RTGS Service for the United Kingdom: Safeguarding Stability, Enabling Innovation. 2016. Available online: https://www.bankofengland.co.uk/news/2016/september/a-new-rtgs-service-for-the-uk-safeguarding-stability-enabling-innovation (accessed on 17 November 2019).
58. Bordo, M.; Levin, A. Central Bank Digital Currency and the Future of Monetary Policy. *Hoover Inst. Econ. Work. Pap.* **2017**. Working paper 23711. [CrossRef]
59. Kumhof, M.; Noone, C. *Central Bank Digital Currencies—Design Principles and Balance Sheet Implications*; Staff Working Paper No. 725; Bank of England: London, UK, 2018.
60. Committee on Payments and Market Infrastructures. *Central Bank Digital Currencies for Cross-Border Payments: Report to the G20*; Bank for International Settlements: Basel, Switzerland, 2021.
61. Bhawana; Kumar, S. Permission Blockchain Network Based Central Bank Digital Currency. In Proceedings of the 2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON), Kuala Lumpur, Malaysia, 24–26 September 2021; pp. 1–6.
62. Han, J.; Kim, J.; Youn, A.; Lee, J.; Chun, Y.; Woo, J.; Hong, J.W.-K. Cos-CBDC: Design and Implementation of CBDC on Cosmos Blockchain. In Proceedings of the 2021 22nd Asia-Pacific; Network Operations and Management Symposium (APNOMS), Tainan, Taiwan, 8–10 September 2021; pp. 303–308.
63. Zhang, J.; Tian, R.; Cao, Y.; Yuan, X.; Yu, Z.; Yan, X.; Zhang, X. A Hybrid Model for Central Bank Digital Currency Based on Blockchain. *IEEE Access* **2021**, *9*, 53589–53601. [CrossRef]
64. Bamakan, S.M.H.; Motavali, A.; Babaei Bondarti, A. A Survey of Blockchain Consensus Algorithms Performance Evaluation Criteria. *Expert Syst. Appl.* **2020**, *154*. [CrossRef]
65. Elsden, C.; Feltwell, T.; Lawson, S.; Vines, J. Recipes for Programmable Money. In Proceedings of the Conference on Human Factors in Computing Systems, Glasgow, UK, 4–9 May 2019.
66. Swan, M. Anticipating the Economic Benefits of Blockchain. *Technol. Innov. Manag. Rev.* **2017**, *7*, 6–13. [CrossRef]
67. Ali, R.; Narula, N. *Redesigning Digital Money: What Can We Learn from a Decade of Cryptocurrencies?* Digital Currency, Initiative, MIT Media Lab.: Stamford, CO, USA, 2019; pp. 1–13.
68. IBM Institute for Business Value Expert Insights—Charting the Evolution of Programmable Money. 2019. Available online: https://www.ibm.com/downloads/cas/GDKQKR6O (accessed on 11 September 2024).
69. Rikken, O. Blockchain Real Time Tax. *LinkedIn*. 22 August 2017. Available online: https://www.linkedin.com/pulse/blockchain-real-time-tax-olivier-rikken (accessed on 19 January 2023).
70. Søgaard, J.S. A Blockchain-Enabled Platform for VAT Settlement. *Int. J. Account. Inf. Syst.* **2021**, *40*, 100502. [CrossRef]
71. Campbell-verduyn, M. Bitcoin, Crypto-Coins, and Global Anti-Money Laundering Governance. *Crime Law Soc. Change* **2018**, *69*, 283–305. [CrossRef]
72. Callegaro, M.; Yang, Y. The Role of Surveys in the Era of "Big Data". In *The Palgrave Handbook of Survey Research*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 175–192. [CrossRef]
73. Whitaker, S.D. Big Data versus a Survey. *Q. Rev. Econ. Finance* **2017**, *67*, 285–296. [CrossRef]
74. Haldane, A. *Will Big Data Keep Its Promise?* 2018. Available online: https://www.bankofengland.co.uk/speech/2018/andy-haldane-centre-for-data-analytics-for-finance-and-macro (accessed on 5 July 2019).
75. Mihaylov, B.; Onea, L.; Hansen, K.M. Architecture-Based Regulatory Compliance Argumentation. *J. Syst. Softw.* **2016**, *119*, 1–30. [CrossRef]
76. Adams, S.; Bowers, L.; Foster, R. *The UK Flow of Funds Project: Comprehensive Review of the UK Financial Accounts*; Office for National Statistics: Newport, UK, 2015; pp. 1–15.

77. Bank for International Settlements. Irving Fisher Committee on Central Bank Statistics. 2017. Available online: https://www.bis.org/ifc/publ/ifc_ar2017.pdf (accessed on 5 July 2019).
78. Hileman, G.; Rauchs, M. Global Blockchain Benchmarking Study. *SSRN* **2017**, 122. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3040224 (accessed on 10 July 2019).
79. Löber, K.; Houben, A. *Central Bank Digital Currencies*; Bank for International Settlements: Basel, Switzerland, 2018; p. 28.
80. HMRC. Preliminary Estimate of the VAT Gap (Tax Year 2020 to 2021). 2021. Available online: https://www.gov.uk/government/statistics/announcements/preliminary-estimate-of-the-vat-gap-tax-year-2020-to-2021 (accessed on 5 April 2022).
81. Valenta, M.; Sandner, P. *Comparison of Ethereum, Hyperledger Fabric and Corda*; Frankfurt School Blockchain Center: Frankfurt, Germany, 2017.
82. R3, Tools and Add-Ons: Corda Accounts Library. 2024. Available online: https://docs.r3.com/en/tools/accounts/accounts-index.html (accessed on 26 April 2023).
83. Omar, A.; Weerakkody, V.; Sivarajah, U. Digitally Enabled Service Transformation in UK Public Sector: A Case Analysis of Universal Credit. *Int. J. Inf. Manag.* **2017**, *37*, 350–356. [CrossRef]
84. VISA, Enhanced Merchant Information. 2020. Available online: https://developer.visa.com/solutions/merchant_information (accessed on 11 September 2024).
85. *ISO/IEC 27001:2022*; Information Security, Cybersecurity and Privacy Protection—Information Security Management Systems—Requirements. ISO: Geneva, Switzerland, 2022. Available online: https://www.iso.org/standard/27001 (accessed on 11 September 2024).
86. National Institute of Standards and Technology. Cybersecurity Framework. 2024. Available online: https://www.nist.gov/cyberframework (accessed on 11 September 2024).
87. *ISO/IEC 19790:2012*; Information Technology—Security Techniques—Security Requirements for Cryptographic Modules. ISO: Geneva, Switzerland, 2012. Available online: https://www.iso.org/standard/52906.html (accessed on 11 September 2024).
88. National Institute of Standards and Technology. *FIPS-140-2. Security Requirements for Cryptographic Modules*; U.S. Department of Commerce: Washington, DC, USA, 2002.
89. National Institute of Standards and Technology. *FIPS-140-3. Security Requirements for Cryptographic Modules*; U.S. Department of Commerce: Washington, DC, USA, 2019.
90. Official PCI Security Standards Council Site. Available online: https://www.pcisecuritystandards.org/ (accessed on 11 September 2024).
91. FCA. Article 8 Requirements of Devices and Software Linked to Elements Categorised as Inherence. In *FCA Handbook*; FCA: London, UK, 2021. Available online: https://www.handbook.fca.org.uk/techstandards/PS/2021/2021_01/chapter-ii/011.html (accessed on 11 September 2024).
92. Jiang, L.; Chang, X.; Liu, Y.; Mišić, J.; Mišić, V.B. Performance Analysis of Hyperledger Fabric Platform: A Hierarchical Model Approach. *Peer--Peer Netw. Appl.* **2020**, *13*, 1014–1025. [CrossRef]
93. Huang, K.; Zhang, X.; Mu, Y.; Rezaeibagha, F.; Du, X. Scalable and Redactable Blockchain with Update and Anonymity. *Inf. Sci.* **2020**, *546*, 25–41. [CrossRef]
94. Melo, W.S.; Bessani, A.; Neves, N.; Santin, A.O.; Carmo, L.F.R.C. Using Blockchains to Implement Distributed Measuring Systems. *IEEE Trans. Instrum. Meas.* **2019**, *68*, 1503–1514. [CrossRef]
95. Khan, K.M.; Arshad, J.; Khan, M.M. Investigating Performance Constraints for Blockchain Based Secure E-Voting System. *Future Gener. Comput. Syst.* **2020**, *105*, 13–26. [CrossRef]
96. Sharma, P.K.; Park, J.H. Blockchain Based Hybrid Network Architecture for the Smart City. *Future Gener. Comput. Syst.* **2018**, *86*, 650–655. [CrossRef]
97. Novo, O. Blockchain Meets IoT: An Architecture for Scalable Access Management in IoT. *IEEE Internet Things J.* **2018**, *5*, 1184–1195. [CrossRef]
98. Hao, Y.; Li, Y.; Dong, X.; Fang, L.; Chen, P. Performance Analysis of Consensus Algorithm in Private Blockchain. In Proceedings of the IEEE Intelligent Vehicle Symposium, Changsu, China, 26–30 June 2018; pp. 280–285. [CrossRef]
99. Kuzlu, M.; Pipattanasomporn, M.; Gurses, L.; Rahman, S. Performance Analysis of a Hyperledger Fabric Blockchain Framework: Throughput, Latency and Scalability. In Proceedings of the 2019 2nd IEEE International Conference on Blockchain, Atlanta, GA, USA, 14–17 July 2019; pp. 536–540. [CrossRef]
100. Wang, H.; Zhang, J. Blockchain Based Data Integrity Verification for Large-Scale IoT Data. *IEEE Access* **2019**, *7*, 164996–165006. [CrossRef]
101. Pal, S.; Rabehaja, T.; Hitchens, M.; Varadharajan, V.; Hill, A. On the Design of a Flexible Delegation Model for the Internet of Things Using Blockchain. *IEEE Trans. Ind. Inform.* **2020**, *16*, 3521–3530. [CrossRef]
102. Si, H.; Sun, C.; Li, Y.; Qiao, H.; Shi, L. IoT Information Sharing Security Mechanism Based on Blockchain Technology. *Future Gener. Comput. Syst.* **2019**, *101*, 1028–1040. [CrossRef]
103. Lu, Q.; Xu, X.; Liu, Y.; Weber, I.; Zhu, L.; Zhang, W. uBaaS: A Unified Blockchain as a Service Platform. *Future Gener. Comput. Syst.* **2019**, *101*, 564–575. [CrossRef]
104. Chen, J.; Lv, Z.; Song, H. Design of Personnel Big Data Management System Based on Blockchain. *Future Gener. Comput. Syst.* **2019**, *101*, 1122–1129. [CrossRef]
105. Li, M.; Weng, J.; Yang, A.; Lu, W.; Zhang, Y.; Hou, L.; Liu, J.N.; Xiang, Y.; Deng, R.H. CrowdBC: A Blockchain-Based Decentralized Framework for Crowdsourcing. *IEEE Trans. Parallel Distrib. Syst.* **2019**, *30*, 1251–1266. [CrossRef]

106. Huang, J.; Li, H.; Zhang, J. Blockchain Based Log System. In Proceedings of the 2018 IEEE International Conference on Big Data, Big Data, Seattle, WA, USA, 10–13 December 2018; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2019; pp. 3033–3038.
107. Li, Y.; Liang, L.; Jia, Y.; Wen, W.; Tang, C.; Chen, Z. Blockchain for Data Sharing at the Network Edge: Trade-Off Between Capability and Security. *IEEE ACM Trans. Netw.* **2024**, *32*, 2616–2630. [CrossRef]
108. R3. Corda Enterprise 4.8. Notaries. 2022. Available online: https://docs.r3.com/en/platform/corda/4.8/enterprise/key-concepts-notaries.html (accessed on 11 September 2024).
109. Manzoor, A.; Braeken, A.; Kanhere, S.S.; Ylianttila, M.; Liyanage, M. Proxy Re-Encryption Enabled Secure and Anonymous IoT Data Sharing Platform Based on Blockchain. *J. Netw. Comput. Appl.* **2021**, *176*, 102917. [CrossRef]
110. Luu, L.; Chu, D.-H.; Olickel, H.; Saxena, P.; Hobor, A. Making Smart Contracts Smarter. In Proceedings of the ACM Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 254–269.
111. Hafid, A.; Hafid, A.S.; Samih, M. Scaling Blockchains: A Comprehensive Survey. *IEEE Access* **2020**, *8*, 125244–125262. [CrossRef]
112. Zhou, Q.; Huang, H.; Zheng, Z.; Bian, J. Solutions to Scalability of Blockchain: A Survey. *IEEE Access* **2020**, *8*, 16440–16455. [CrossRef]
113. Rosenblum, D.S. A Practical Approach to Programming with Assertions. *IEEE Trans. Softw. Eng.* **1995**, *21*, 19–31. [CrossRef]
114. Gooding, P. Consumer Price Inflation Basket of Goods and Services: 2022—Inflation and Price Indices. 2022. Available online: https://www.ons.gov.uk/economy/inflationandpriceindices/articles/ukconsumerpriceinflationbasketofgoodsandservices/2022 (accessed on 5 February 2023).