Application for Electronic Signatures Using Blockchain Technology to Support Trust, Sovereignty and Privacy

1st Xxxxxxx Xxxxxxxx
Xxxxxxxx Xxxxxxx Xxxxxxxxxx
Xxxxxxxxxx xx xxx Xxxxxxxxxx Xxxxxx
Xxxxxxxxx, Xxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxx

2nd Xxxxxxxx Xxxxxx
Xxxxxxxx Xxxxxxx Xxxxxxxxxx
Xxxxxxxxxx xx xxx Xxxxxxxxxx Xxxxxx
Xxxxxxxxx, Xxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxx

***Abstract*—Electronic signatures are an essential component in the digitalization of processes and communication. When designing signature systems, legal regulations and data protection principles must be taken into account in addition to trust issues. This paper presents a web-based signature application using blockchain technology that also shows how signature information can be verified across providers. This design is intended to strengthen principles such as trust, sovereignty, data protection and data security as well as trust in the implemented system. In addition, new concepts such as URL- and QR code-based placeholders for signatures were implemented and tested, which have the potential to further accelerate and decentralize business processes. The system design is based on a set of criteria relating to trust, usability, privacy and the legal situation. In addition, an iterative cost optimization of the Smart Contracts was applied in order to improve the potential economic viability of the system.**

***Index Terms*—blockchain applications, electronic signatures, data privacy, digital sovereignty, system usability, system trust**

I. Introduction

Electronic signatures play an indispensable role in everyday life as well as in the corporate environment. A declaration of intent must always be reliably traceable to an identity, with the intention of the signer being confirmed by the signature. To electronically sign documents such as contracts, messages, certificates and deeds, the options are fairly limited. Not because they are particularly difficult to implement or because of the lack of suitable tools [1], but because the usability of those systems is limited [2], [3] and innovation in legally binding processes is slow [4].

A. Motivation

One method that promises high usability and is becoming increasingly popular is to use web services for electronic signatures. Usually, the document creator uploads the document, for example a contract, to the signature service website, configures signature fields if necessary, and triggers notification of the signers. The service locks the document to prevent subsequent changes. The signers usually receive the notification by e-mail and click on a link that provides access to the signature submission process. Different authentication and verification methods might be used. Simplicity, relative advantage, and compatibility of cloud-based services have a significant impact on suitability, unlike perceived security, which can be seen as a result of a lack of consumer confidence in the complementarity of cloud services [5]. Another concern is data protection, the confidentiality of documents, and the high level of trust that a company or user of such a service must place in its operators. An organization or individual that shares its documents with such a provider is also sharing a vast amount of confidential information. This is in addition to the fact that this provider is the only source of truth regarding signature and document related information.

In this work, we present a prototypical approach for a web service for electronic signatures that does not require document uploads, while reinforcing its trustworthiness in terms of data validity and consistency by connecting to a public blockchain. The interface to the blockchain is designed in a way that different signing applications and services can implement it and users are able to check the signature data for correctness via different application providers. This is intended to strengthen **trust** and **sovereignty** as well as **independence** from the provider.

The developed prototype serves to test and validate different authentication methods as well as new concepts like URL/QR-

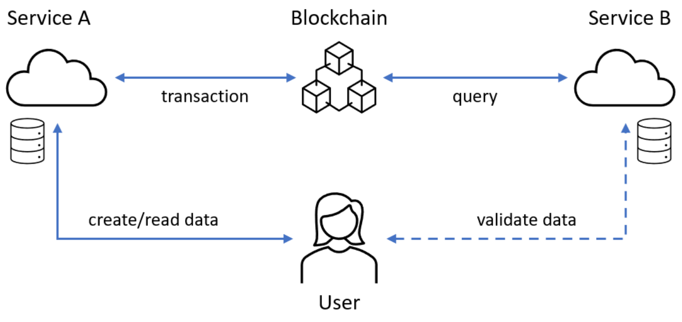


Fig. 1. Signature system architecture.

Identify applicable funding agency here. If none, delete this.

based signatures and represents a reference implementation for blockchain-secured signature information and its verification. Figure 1 shows the general system architecture, in which the user primarily interacts with the service of his choice, but can verify the validity and consistency of the data presented to him at any time via an alternative service from another, possibly competing, provider.

B. Content

This work is structured as follows. The requirements for the system are collected in Section II and then translated into a concept in Section III. Section IV then describes the implementation of the associated components and applications including blockchain cost-optimization, and Section V explains the technical and requirements-based validation of the concept. Finally, the conclusion and future work follow in Sections VI and VII.

C. Method

The first step is to identify the requirements for the intended system. These criteria are mainly derived from literature and consist of legal requirements as well as requirements regarding usability, data privacy and trust and are incorporated into the targeted concept. The system is then implemented and optimized in terms of blockchain costs. Iteration loops run between concept and implementation as well as between implementation and validation. The validation is divided into a technical validation and a validation against the collected criteria, where the technical validation takes place through application tests and functional evaluations. The method of this work is based in part on the design science research methodology (DSRM) by Peffers et. al. [6]. Conceptually, a design research artifact can be any designed object in which a research contribution is embedded in the design [7]. In this case, the artifact consists of a prototype for a web-based application for electronic signatures as well as code for smart contracts targeting public blockchain technology (Ethereum).

II. Requirements

This section compiles the design criteria as a basis for the system design, separated by legal criteria, as well as those of usability, data privacy and trust. The resulting requirements are summarized in Table I.

A. Legal Requirements on Electronic Signatures

Representing a large part of the Western Economic Area, this section considers the legal requirements of the European Union and the United States of America and translates them into basic criteria. For a broader global context, these are complemented by regulations in China as a representative of the East.

1) European Union: The European regulation on electronic identification and trust services for electronic transactions in the internal market (eIDAS) sets basic requirements for electronic signature systems. In this context, a difference is made between simple electronic signatures, advanced electronic signatures (AdvES), and qualified electronic signatures (QES). The latter are, according to the regulation, equal to handwritten signatures in their legal effect. QES require qualified digital certificates (QDC) issued by qualified trust service providers (QTSP), but advanced electronic signatures are sufficient for the vast majority of business processes. Furthermore, the regulation states that neither the legal effect nor the admissibility in court proceedings may be denied on the basis of the electronic form or the lack of qualification of an electronic signature [8]. The eIDAS regulations stipulate that signatures must be clearly assignable to a signatory, the signatory must be identifiable, the signature must be created on the basis of signature data in the signatory's possession, and the signature must be linked to the signed data.

2) United States of America (USA): In the USA, the Electronic Signatures in Global and National Commerce Act (ESIGN Act) of 2000 and the Uniform Electronic Transactions Act (UETA Act) of 1999 need to be considered. The ESIGN Act, similar to eIDAS, states that a signature, record, or contract signed with it may not be denied validity or the ability to be enforced just because an electronic signature was used. The ESIGN Act defines an electronic signature as an electronic sound, symbol, or process attached to a contract or other record that is executed with the intent to sign the record [9]. The UETA Act defines that the signer must be aware of the legal effect and the fact that he is signing. The intention to sign is what defines the signature. Furthermore, it is described that the signature must be able to be assigned to the signer, even if this is performed by a human or electronic representative [10].

3) China: The chinese Ministry of Information Industry enacted the Administrative Measure on Electronic Certification Service and defines an electronic signature as data in electronic form contained in or attached to a data message, which may be used to identify the signatory in relation to the data message and to indicate the signatory's approval of the information contained in the data message [11]. The requirements for reliable electronic signatures are similar to those of AdvES defined by the eIDAS [11].

B. Privacy Requirements

In addition to giving EU citizens more ability to control their own personal data at organizations inside and outside the EU, the European General Data Protection Regulation (GDPR) has inspired sweeping new laws in the USA and continues to be the most cited privacy regulation when new regulations are considered in the USA and around the world [12]. The GDPR sets strict rules for the processing of personal data, which can pose a problem when using blockchains to manage

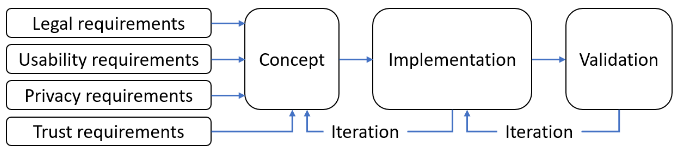


Fig. 2. Iterative design method.

personal data. Article 17 of GDPR that states the *right to be forgotten* requiring the concerned organizations to delete the user data if the user requests it, but since information inside the blockchain cannot be removed, this directly contradicts with Article 17 [13]. When using blockchains, the GDPR consequently requires *privacy by design* principles and a data protection impact assessment [14]. As a consequence for this design, no signature, document or personal data should be stored directly on the blockchain, but this can be replaced by storing a cryptographic proof of their correctness. In addition, the GDPR deals with the responsibilities regarding data processing, which blockchains have their problems with, since here the miners are practically the processing entities [13]. Public blockchains do not have a geographical border for their network, which could also lead to problems regarding the territorial scope defined in the GDPR, and the restrictions regarding territorial scope were considered difficult to implement, which is why permissioned blockchains were recommended instead of public blockchains [15]. Since private blockchains have their own problems, as indicated in the next section, the problem of territorial scope must be solved.

C. Trust Requirements

From the user's point of view the information system and the provider are the two most important targets of trust [16]. Systems should support the exchange of reliable trust cues, and thus allow for correct trust attribution [17]. The studies by Thilesch et al. showed a high relevance of technical system reliability and information quality, in particular the credibility of the system [18]. In our case, trust refers to the correctness of the data, especially the documents and signatures including the relevant metadata, so there should be a way to verify the truth of this data using independent sources of information. Also, social influence as a determinant of trusting beliefs, attitudes, and intentions has a strong effect on trusting beliefs [19]. A decentralized information system approach promises to lower the cost of trust [20]. Integrity of data ensures that data is modified only by those who are authorized to do so, and to maintain data integrity, several cryptographic methods like Secure Hash Algorithm (SHA-2) are used [21]. The trust mechanism that the blockchain enables provides a more transparent, accountable, and controlled handling of verifying competence and experience [22]. Since private and consortium blockchains have low decentralization and less security [23], a public ledger was preferred for our concept to store document and signature proofs.

D. Usability Requirements

Since existing tools do not provide a high level of usability [2], [3] and web or cloud-based signature services are the most accessible to the user, an approach that is the same as or close to a web service should be pursued. The usability of websites can be affected by adapted content, awareness of the target audience, clear navigation and structure, as well as an efficient process [24]. Another important aspect is the barrier-free availability of the system, regardless of the end user's operating system (OS) and type of device. Since desktop and mobile-based blockchain wallets might lack good usability [25] and the end user should not need a blockchain account, the connection to the blockchain should be provided by the service to which the user connects. Further disadvantages of using public blockchains like Ethereum are the monetary price of transactions and scalability issues related to low throughput [22]. As a consequence, the smart contracts and their interfaces should be optimized in terms of cost and transaction performance.

Table I summarizes the requirements derived in this section.

TABLE I
REQUIREMENTS FOR BLOCKCHAIN- AND WEB-BASED ELECTONIC SIGNATURE SYSTEMS.

Name	Category	Required by	Description
LegQDC	Legal	QES	based on QDC from QTSP
LegASG	Legal	QES, AdvES	sign. assignable to signer
LegIDF	Legal	QES, AdvES	identifiable signer
LegOWN	Legal	QES, AdvES	signer owns data to create sign.
LegLNK	Legal	QES, AdvES	linking of sign. w. signed data
LegAWN	Legal	QES, AdvES	awareness of legal effect
LegINT	Legal	QES, AdvES	clear intention to sign
LegDOC	Legal	QES, AdvES	detect document changes
LegPRO	Legal	QES, AdvES	provability of signature
PrivDEL	Privacy	GDPR	right to be forgotten
PrivDSN	Privacy	GDPR	privacy by design
PrivRSP	Privacy	GDPR	responsible processing entity
PrivTER	Privacy	GDPR	territorial scope
PrivNOD	Privacy	GDPR	no document in blockchain
PrivNOS	Privacy	GDPR	no signature in blockchain
PrivDOC	Privacy	(user)	no document upload
TrusREL	Trust	(user)	system reliability
TrusQTY	Trust	(user)	information quality
TrusATT	Trust	(user)	trust attribution
TrusINT	Trust	(user)	integrity of data
TrusVER	Trust	(user)	ability to verify data
TrusIDP	Trust	(user)	independent sources of inform.
TrusSOC	Trust	(user)	social influence
TrusDEC	Trust	(user)	decentralized information
TrusPUB	Trust	(user)	public ledger/blockchain
UsbCON	Usability	(user)	adapted content
UsbAWN	Usability	(user)	awareness of target audience
UsbNAV	Usability	(user)	clear navigation
UsbEFF	Usability	(user)	efficient process
UsbOS	Usability	(user)	OS independency
UsbDEV	Usability	(user)	desktop and mobile devices
UsbACC	Usability	(user)	no blockchain account by user
UsbBCC	Usability	(user)	blockchain transaction costs
UsbBCT	Usability	(user)	blockchain transaction time

III. CONCEPT

In this section, the system concept is designed and explained in detail based on the requirements identified in Section II. The general architecture, the processes for creating signed documents and for verifying identities, documents and signatures are presented.

A. Architecture

Figure 3 shows a signature web service connected to a database, a public blockchain, identity A and identity B. The service is an intelligent website (UsbOS) where, in this example, A has a user account (PrivRSP, PrivTER). This

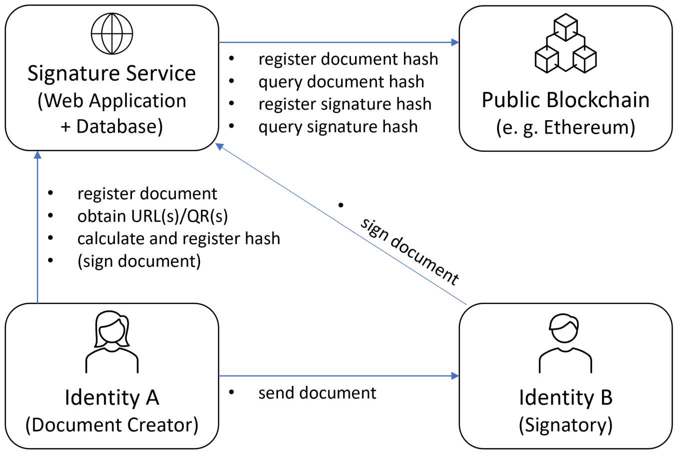


Fig. 3. Workflow and communication architecture.

allows *A* to register documents for signature, eventually for a fee. *A* then sends the document to *B* for signature. The data relating to the document and the signatures is secured against manipulation via the blockchain (TrusSOC). The next subsections explain the process in detail.

B. Document Registration

After the registration of a document, *A* receives one or more combinations of URL and Quick Response (QR) code containing the URL, which can be used to submit a signature and retrieve it once it has been signed (LegLNK, UsbNAV). The user inserts this combination of URL and QR code into the document and saves it permanently. In order to prevent future changes or at least make them recognizable, the fingerprint of the document is calculated as hash via the browser (PrivDOC, PrivNOD) and transmitted to the service (LegDOC). Now *A* can send the document to *B* and potentially other identities via the communication channels of his choice. *B* can then use the URL to submit his/her signature. *B* is then directed to the service, where he/she is lead through the necessary process including identity verification.

C. Identity Verification and Signing

The URL first takes the signatory to a website where information about the document is displayed and the option to check the document is offered. Depending on whether the electronic signature needs to be a QES, various identity verification options are available. The technique for signing then depends on the method chosen.

1) Login: If *B* also has a user account, he can confirm his identity via the login, as he already confirmed his identity when registering the user account (LegIDF, LegASG, LegPRO). As this type of verification does not offer the highest level of security, a manual signature must still be provided using a mouse or finger on the touchscreen, as handwritten signatures indicate a clear intention (LegINT, LegAWN) and have a certain degree of verifiability, e.g. using deep learning-based methods [26] (LegIDF, LegPRO, LegOWN).

2) E-Mail: If *B* does not have a user account and no QES is required, he can identify himself via his e-mail address. In this case, *B* is sent a randomly generated code, which he must then enter (LegIDF, LegASG). In this case, his identity is confirmed purely via the e-mail address and a manual signature is therefore also required here (LegINT, LegAWN, LegIDF, LegPRO, LegOWN).

3) Certificate: A method with greater security and probative value is the signature using an PKCS #12 certificate [27] in PFX or P12 format (LegIDF, LegASG, LegOWN, LegPRO). *B* is asked to enter his personal certificate and the corresponding password. With this method, the signing action consists of a confirming press of a button with a distinct label (LegINT, LegAWN), whereupon the cryptographic signature and a X.509 certificate [28] without private key are filed. If a QDC is used (LegQDC), this is a QES.

4) Certificates App: Since the simultaneous submission of certificate and password might not feel secure for *B*, and in most cases the signatory cannot judge whether the process takes place in the front or back end, we have tried an alternative approach. We have developed a prototypical mobile app that stores one or more certificates (LegIDF, LegASG, LegOWN, LegPRO). A web application can request the app to provide a signature via a QR code containing a URL with a specific scheme (LegINT, LegAWN). The QR code contains the data record to be signed and the endpoint to which the signature and the certificate without private key are to be sent. Apart from the transmission method and the location where the signature is calculated, the procedure is the same as described in the previous paragraph. Here too, a QES can be achieved using a QDC (LegQDC).

D. Data Verification

In the context of this application, verification functions are required to increase confidence in the correctness and integrity of the data. These are based on blockchain technology (TrusDEC, TrusREL) in this design. The relevant data records are the registered documents and the signatures together with the associated information.

1) Verification of Documents: As described in Section III-B, only the hash codes of the documents are transferred to the service. These hashes are registered by the service in the blockchain (UsbACC) via a corresponding smart contract. This protects the time and value of the entry in a trustworthy manner (LegDOC, TrusINT). The recipient of a file could theoretically also check it using third party, possibly competing services or applications using the documents identifier (ID) and hash (TrusVER, TrusIDP). In our prototype, the verification function is offered directly before the signature is submitted.

2) Verification of Signatures: With regard to signatures, it must be possible to check an entire data object consisting of several pieces of information. This object consists at least of the following properties:

- *blockchainMappingID*: the identifier of the signature in the blockchain.

- *documentID*: the identifier of the document.
- *documentSlot*: the number of the signature placeholder on the document.
- *timeStamp*: the timestamp calculated by the blockchain.
- *signatureData*: the signature itself; a byte array as BASE64 in case of signing with certificates; a string containing polylines representing the handdrawn signature.
- *verificationData*: the e-mail address or certificate depending on the used method.

A hash code is calculated from these values (except for the *blockchainMappingID*) and entered in the blockchain (PrivNOS, TrusINT). The blockchain determines the *timeStamp*, stores it together with the hash code and returns the *blockchainMappingID*. As JavaScript Object Notation (JSON) is one of the most commonly used data formats [29] and is particularly performant in JavaScript environments [30], our prototope offers to download a signature as a JSON file while viewing it. In addition, it offers the option of uploading such a JSON and validating it against the blockchain (TrusVER, TrusIDP). By distributing the signature information as a file and by storing the signature information in the blockchain purely as a hash, verification is independent of the existence of the information in the service's database (PrivDEL), nor does personal information migrate to the blockchain (PrivDEL, PrivDSN). The distribution of the signature information to the identities as JSON together with the possibility of verifying the truth of the information contained ensures independence from the provider and provability remains even if the provider shuts down (TrusREL).

E. Open and Distributed System Approach

Applications such as the prototype developed in this work could be developed and deployed by multiple and different providers, or even by organizations that want to offer their customers a fluid digital process for concluding contracts directly including the process of signing, without the trustworthiness depending on trust in this one organization. To increase trustworthiness, existing signature applications could also secure signature and document information via the blockchain using the same Smart Contracts, as well as provide verification functions according to this scheme (TrusIDP, TrusDEC, TrusVER, TrusATT).

IV. Implementation

The concept presented in Section III was implemented as a web application in conjunction with a local database and connected to an Ethereum-based public (TrusPUB) blockchain.

A. Application

The web application was developed with ASP.NET using the Model View Controller (MVC) design pattern and *Bootstrap* for styling (UsbOS, UsbDEV). SQLite was used to store the application data in this prototypical project, as SQLite is cross-platform, fast, highly reliable and easy to manage as a single file [31].

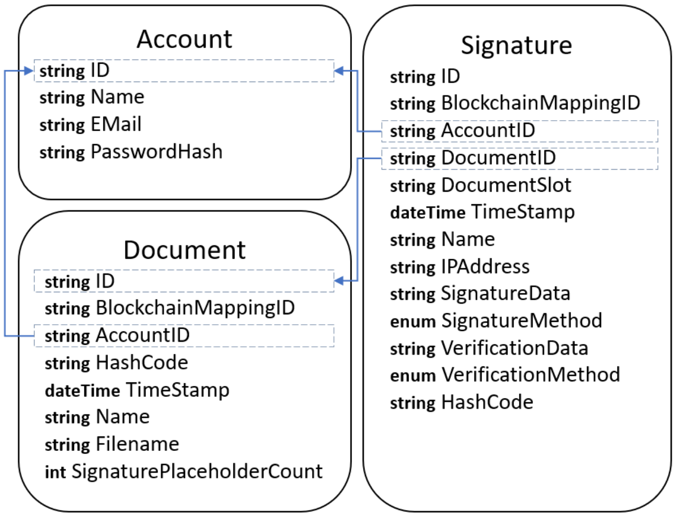
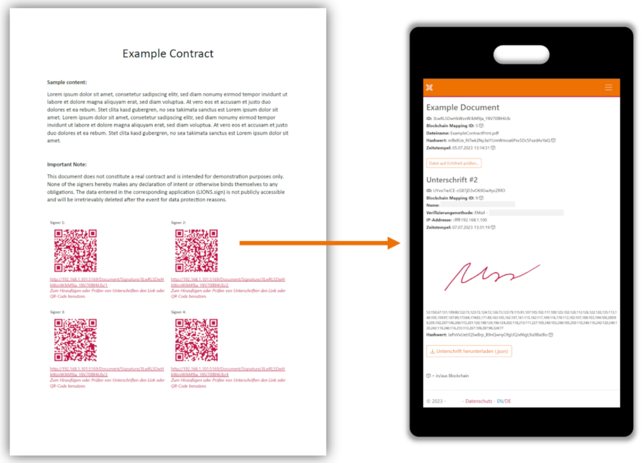


Fig. 4. Application data objects and properties.

1) Data Objects: The application only requires the three object types *Account*, *Document* and *Signature* to store the application information in the database. Figure 4 shows the objects with their properties, including the data type. The *Account* object was limited to the properties necessary for the function of the application and hides information relevant to billing for this prototype. All objects have the *ID* property, which is the unique identifier within the application and database. The objects *Document* and *Signature* have the property *BlockchainMappingID*, which represents the identifier of the reference object within the blockchain.

2) Document Creation: A registered user can create a new document in the application, that is, a new entry for a document file including a hash. This requires several steps, which is why the user is guided through the process step by step in the form of a wizard displaying relevant instructions (UsbNAV, UsbCON, UsbEFF). In the first step, the user creates the local document file in his/her preferred program, such as *MS Word*, and creates a new entry in the signature application, specifying a name and the number of required signature placeholders. The user then receives a URL/QR code combination for each placeholder, which can be copied to the clipboard by pressing a button (UsbEFF). The URL contains the *DocumentID* of the *DocumentSlot*. These URL/QR code combinations are inserted by the user into the file (LegLNK) and adjusted visually if necessary. Next, the user finally saves or exports the file, e.g. as a PDF, and then selects it in the front end of the application. Here, the hash code of the file is calculated using JavaScript (PrivDOC, PrivDSN) and transmitted to the back end, whereby the document is entered in the blockchain. This completes the process and the user can send the document to the recipient(s) via the communication channel of their choice.

3) Signature Creation: If no signature has yet been entered on a placeholder, its URL leads to the submission of the signature (UsbNAV). The displayed page offers a button for checking the file fingerprint, whereby the calculation is again


Example Contract

Sample content:

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Important Note:

This document does not constitute a real contract and is intended for demonstration purposes only. None of the signers hereby makes any declaration of intent or otherwise binds themselves to any obligations. The data entered in the corresponding application (LIONS.sign) is not publicly accessible and will be irretrievably deleted after the event for data protection reasons.

Signer 1:
http://192.168.1.101:5169/Document/Signature/3LwRLSDwHkWzvWJkM9ja_Y8V70lBHiUb/1
Zum Hinzufügen oder Prüfen von Unterschriften den Link oder QR-Code benutzen.

Signer 2:
http://192.168.1.101:5169/Document/Signature/3LwRLSDwHkWzvWJkM9ja_Y8V70lBHiUb/2
Zum Hinzufügen oder Prüfen von Unterschriften den Link oder QR-Code benutzen.

Signer 3:
http://192.168.1.101:5169/Document/Signature/3LwRLSDwHkWzvWJkM9ja_Y8V70lBHiUb/3
Zum Hinzufügen oder Prüfen von Unterschriften den Link oder QR-Code benutzen.

Signer 4:
http://192.168.1.101:5169/Document/Signature/3LwRLSDwHkWzvWJkM9ja_Y8V70lBHiUb/4
Zum Hinzufügen oder Prüfen von Unterschriften den Link oder QR-Code benutzen.

Example Document

ID: 3LwRLSDwHkWzvWJkM9ja_Y8V70lBHiUb
Blockchain Mapping ID: 5
Dateiname: ExampleContractPrint.pdf
Hashwert: mfle8Ue_RiTwk2Ny3el1UmWmoa6Psv5Dc5FxzdAvYaQ
Zeitstempel: 05.07.2023 13:14:31

Unterschrift #2

ID: UYvo7ieJCE-cG87jD3vO69GwJtycZRfO
Blockchain Mapping ID: 9
Name: [redacted]
Verifizierungsmethode: EMail - [redacted]
IP-Addresse: ::ffff:192.168.1.100
Zeitstempel: 07.07.2023 13:31:19

53:150,67:131,109:80,122:73,123:72,124:72,126:73,123:79,115:91,107:105,102:117,100:125,102:128,112:126,122:120,135:113,148:105,159:97,167:89,173:84,174:83,171:89,163:103,162:107,161:115,162:117,169:116,176:112,182:107,188:103,194:100,200:99,205:102,207:106,206:113,201:120,198:124,196:124,202:118,213:111,227:105,240:103,248:105,250:110,246:116,242:120,240:120,242:119,246:116,255:113,267:109,287:99,324:77
Hashwert: JaPxVxUet/Q5wBrp_B9nQwnyOfgUQJxNigL9u9BaJBo

Datei auf Echtheit prüfen...
Unterschrift herunterladen (.json)
= in/aus Blockchain
© 2023 - - Datenschutz - EN/DE

Fig. 5. Signature placeholders as QR code and URL.

performed in JavaScript and only the hash is transmitted. In addition, the user can now choose between the identification methods described in Section III-C using radio buttons. This process is also implemented as a wizard (UsbNAV, UsbCON, UsbAWN, UsbEFF). If a manual signature is required, this is done via a canvas object, whereby the polyline is recorded in the form of coordinates and transferred as string.

4) Signature Information: If a signature has already been provided on a placeholder, its URL leads to a page that presents the signature with all relevant information and functions (UsbNAV). This page shows all the relevant properties of the document and signature (TrusQTY), with the information secured via the blockchain marked with a block symbol. The manual signature, if available, or otherwise the digital signature is also displayed. There are also download buttons for the signature as JSON and, if applicable, for the signatory's certificate. Figure 5 shows a rough illustration of an example document with four placeholders and the signature information with a handwritten signature, displayed on a smartphone.

5) Blockchain Interface: Since our approach is based on Ethereum (see Section IV-B) and the web application was implemented using .NET, we used *Nethereum*. Nethereum is an open source .NET integration library for Ethereum that simplifies smart contract maintenance and interaction [32].

B. Blockchain

As explained in Section III-A, this work targets a public blockchain. Ethereum addresses several limitations of the Bitcoin's scripting language and supports all types of computations, including loops, as well as several other improvements over the blockchain structure [33]. In order to develop and test the Smart Contracts easily and free of charge, a local Ethereum network was set up on the development machine using the tool *Ganache*. This also allows accounts, contracts, transactions and events to be viewed and evaluated in a convenient way.

C. Smart Contracts

Smart Contracts for Ethereum are developed with the programming language *Solidity*. In order to apply the concept, the following functions were implemented:

1) createDocument: This function is used to register a document, even if no file fingerprint is yet known. It does not require any input parameters, stores an empty object for the document in the blockchain and returns the *BlockchainMappingID* via the event.

2) updateDocument: This function is used to assign the hash code to a document object and the blockchain then sets the timestamp on the object. This is only possible if a timestamp has not yet been set. I takes the *BlockchainMappingID* and *HashCode* as input and returns the timestamp via the corresponding event.

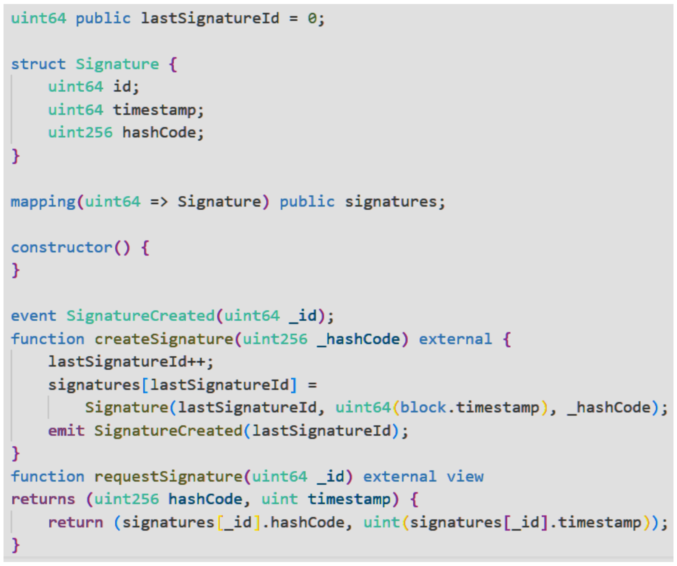
3) requestDocument: This function is used to directly obtain the hash code and timestamp for a document passing the *BlockchainMappingID* as parameter.

4) createSignature: This function is used to store a reference object for a signature data record in the blockchain. The hash of the data record is transmitted as a parameter and the *BlockchainMappingID* and timestamp are returned via the event. Figure 6 shows the Solidity code for this function and the required stucture, storage variables and event.

5) requestSignature: This function is used to directly obtain the hash code and timestamp for a signature passing the *BlockchainMappingID* as parameter.

D. Cost Optimization

As costs incur for the use of smart contracts, i.e. for registering and signing documents, these must be as low as possible in order to be bearable or profitable for the potential signature service provider. This section therefore looks at the iterations carried out to optimize costs (UsbBCC). The costs for the execution of smart contracts in Ethereum are


```
uint64 public lastSignatureId = 0;

struct Signature {
    uint64 id;
    uint64 timestamp;
    uint256 hashCode;
}

mapping(uint64 => Signature) public signatures;

constructor() {
}

event SignatureCreated(uint64 _id);
function createSignature(uint256 _hashCode) external {
    lastSignatureId++;
    signatures[lastSignatureId] =
        Signature(lastSignatureId, uint64(block.timestamp), _hashCode);
    emit SignatureCreated(lastSignatureId);
}
function requestSignature(uint64 _id) external view
returns (uint256 hashCode, uint timestamp) {
    return (signatures[_id].hashCode, uint(signatures[_id].timestamp));
}
```

Fig. 6. Smart Contract code for signatures.

measured in **gas**, whose real value depends on the *gas price*. The gas price is not fixed but set by the blockchain user and miners prefer transactions with higher fees, meaning that transactions whose fees are too low may never be included in the blockchain (UsbBCT), while setting a very high gas fee guarantees that the transaction will be executed quickly [34]. In this cost optimization, we measure the gas consumption, as this variable is independent of exchange rate and price and reflects the necessarily resulting total costs as a factor. Table II shows the exact gas consumption for all versions. The minimum, maximum, average and median costs signature costs and the costs for contract creation are shown for each version. The signature function was called ten times for each calculation.

1) Version 1: In the first version, all functions were combined in one contract and the hash codes were provided as strings to support different hash lengths. All functions were implemented with return values.

2) Version 2: For the first step in the optimization process, the return values were removed for version 2 and unnecessary local variables within the functions were removed, since each instruction and defining variables costs gas [34]. This step brought an improvement in gas consumption, but only a very small one.

3) Version 3: In Solidity, fixed size variables are cheaper than variables with variable size like strings [34]. Since data storage in the contract has the greatest impact on costs [35], we have optimized the object structure for the signature. We have therefore changed the data type of the hash code from *string* to *uint256*, which means that only a SHA256 is supported, but a very high saving is achieved.

4) Version 4: Another pattern is tight packing structures by assigning small data types to consecutive variables to save them into a single storage slot [36]. We therefore adjusted the order of the structure variables, but this had no effect on the results and is an indication that either the compiler had already packed tightly beforehand, or the change in order was not sufficient.

5) Version 5: To save unnecessary parameters, the timestamp was removed from the signature creation event in this iteration step. This only had a minimal impact on the costs.

6) Version 6: Since for reasons of cost-efficient maintainability of smart contract code, separable functions should be divided into multiple smart contracts [34] [36], and because the size of a contract could have an impact on transaction costs, we moved the functions for the documents to another smart contract. As shown in Table II, this also had only small impact on costs.

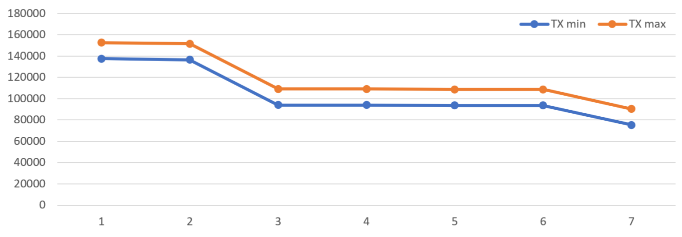


Fig. 7. Gas cost of creating signatures.

7) Version 7: Since external functions consume less gas than public ones [34], and our functions do not have to be called by others, we have changed the functions from *public* to *external*. Since tight packing did not work in Version 4 and only variables of the same type can be packed into a storage segment [34], we changed the data type of the timestamp from *uint* to *uint64*. These changes have drastically reduced the costs.

Figure 7 shows the cost reduction using the minimum and maximum transaction costs for the creation of a signature. The maximum costs were actually only incurred for the first execution of a smart contract, which is why the average and median are quite different. Table II shows this difference very clearly and it is noticeable that the median actually corresponds to the minimum and this can be used as the gas consumption incurred. For versions 6 and 7, the provision costs for both contracts are given in total, with the costs for the signature contract in brackets.

TABLE II
SIGNATURES GAS COST.

Version	contract(s)	TX min	TX max	TX avg	TX med
1	902.362	137.687	152.687	139.187	137.687
2	889.857	136.536	151.536	138.036	136.536
3	592.174	94.042	109.042	95.542	94.042
4	592.174	94.042	109.042	95.542	94.042
5	592.174	93.684	108.684	95.184	93.684
6	748.662 (330.857)	93.661	108.661	94.161	93.661
7	769.596 (351.791)	75.378	90.378	76.878	95.378

V. VALIDATION

In this work, validation consists on the one hand of technical validation based on practical tests of the implemented prototype and on the other hand of verifying whether all requirements of Section II were taken into account.

A. Technical Validation

The practical validation consisted of various software tests with different documents, verification and signature procedures (see Section III-C). This involved monitoring data at runtime using the step-by-step debugging feature of Visual Studio, checking the SQLite database content and reconciling blockchain transactions using *Ganache*. The verification functions built into the prototype, which also make use of the blockchain, were tested using corrupted and unmodified documents and signature JSON files, after which the expected results were displayed. Although the web application, the database and the blockchain network were operated or executed on the same machine, the application was also accessed by other devices, including mobile devices, via the local network.

B. Requirements Validation

In addition to the practical validation, it must be checked whether the legal requirements (Section II-A) and the requirements regarding privacy (Section II-B), trust (Section II-C) and usability (Section II-D) have been taken into account. The criteria are listed in Table III, indicating by which aspects they were taken into account and in which section of this work this information can be found.

TABLE III
REQUIREMENTS VALIDATION.

Name	Addressed by	Section(s)
LegQDC	options too use QDC	III-C3, III-C4
LegASG	linked with email or cert	III-C1, III-C2, III-C3, III-C4
LegIDF	email or cert subject	III-C1, III-C2, III-C3, III-C4
LegOWN	password or private key	III-C1, III-C2, III-C3, III-C4
LegLNK	link on document	III-B, IV-A2
LegAWN	manual sign. or button	III-C1, III-C2, III-C3, III-C4
LegINT	manual sign. or button	III-C1, III-C2, III-C3, III-C4
LegDOC	store document hash	III-B, III-D1
LegPRO	email, man. sign, priv. key	III-C1, III-C2, III-C3, III-C4
PrivDEL	offchain database	III-D2
PrivDSN	offchain, no doc. upload	III-D2, IV-A2
PrivRSP	service is responsible	III-A
PrivTER	service provider	III-A
PrivNOD	hash only	III-B
PrivNOS	hash only	III-D2
PrivDOC	not document upload	III-B, IV-A2
TrusREL	decetralized / blockchain	III-D
TrusQTY	detailed information view	IV-A4
TrusATT	decentralized system	III-E
TrusINT	hashes and timestamps	III-D1, III-D2
TrusVER	compare hashes	III-D1, III-D2, III-E
TrusIDP	decentralized system	III-D1, III-D2, III-E
TrusSOC	blockchain based	III-A
TrusDEC	blockchain based	III-D, III-E
TrusPUB	Ethereum as blockchain	IV
UsbCON	wizards for workflows	IV-A2, IV-A3
UsbAWN	signature wizard	IV-A3
UsbNAV	QR links and wizards	III-B, IV-A2, IV-A3
UsbEFF	wizards, valid. buttons	IV-A2, IV-A3
UsbOS	web application	III-A, IV-A
UsbDEV	responsive web app	IV-A
UsbACC	service as blockchain user	III-D
UsbBCC	cost optimization	IV-D
UsbBCT	control via gas price	IV-D

This table verifies that all criteria have been taken into account and that the legal requirements have been met for all identity verification methods.

VI. Conclusion

A. Summary

In this work, a set of legal criteria for electronic signature systems and criteria for privacy, trust and usability were collected and summarized as a basis for concept development. These criteria were incorporated into a concept that includes a web application with a local database for off-chain data and a connection to an Ethereum blockchain with specially developed smart contracts. Only hashes of the information are stored in the blockchain in order to meet data protection requirements. The smart contracts were optimized in several iterations with a view to the cost-effectiveness of such a system.

B. Contribution

As a partial result, the criteria catalog contributes a solid basis for the development of further systems. The system concept shows how data can be processed and secured against falsification or modification via a blockchain without coming into conflict with data protection requirements and thus serves as a reference or blueprint for similar projects. In particular, the concept opens up an idea of how the signature data in new, but also in existing signature systems could be secured among competitors in a trustworthy but data-saving manner, increasing trust and sovereignty. The cost optimization of our smart contracts also contributes information on which steps are particularly relevant and have a great impact on blockchain related costs.

C. Considerations

Depending on how expensive the transactions are with Ethereum, an alternative public blockchain with lower costs could be considered. Another option would be to use a private blockchain, which would eliminate transaction costs but raises governance issues. The governance in a private blockchain assigns authority and responsibility among the consortium members and determines nodes that will be able to create blocks, to read/write data and to contribute in the consensus mechanism [37]. It would then remain to be examined what difference this makes in terms of trust by the user.

VII. Future Work

At the time of this writing, we have an empirical study in progress that is looking at the usability and perceived trustworthiness of existing signature systems and is also investigating the acceptance of concepts such as QR codes as placeholders for signatures and the influence of a blockchain connection on trustworthiness. Depending on the results, the findings of the study may then be incorporated back into this concept. We will also investigate other identity systems and thus methods for identifying the signatory and evaluate them using this signature system. Furthermore, this system with its applications and components will serve as a reference for further and other implementations and will therefore be made available in a public repository.

Acknowledgment

xxxx xxxx xxxxxxxxxxx xxxx xxx xxxxx xxxxxxxx xxxxxxx. xxxxx xx xxxxxx xx xxxxxxx — xxxxxxxxxxxxxx xxx xxxxxxxxxx xxxxxxxx xxxxxx xx xxx xxxxxxxxxx, xxxxx xx xxxxxxxxxx xxxxxxxxxxx. xxxxxxx xx xxxxxx xx xxx xxxxxxxx xxxxx — xxxxxxxxxxxxxxxx.

References

[1] S. Blythe, "Digital Signature Law of the United Nations, European Union, United Kingdom and United States: Promotion of Growth in E-Commerce With Enhanced Security," *Richmond Journal of Law & Technology*, vol. 11, p. 6, Jan. 2005.

[2] F. H. Monzón, M. Tupia, and M. Bruzza, "Security Versus Usability in E-Government: Insights from the Literature," in *Developments and Advances in Defense and Security* (Á. Rocha, M. Paredes-Calderón, and T. Guarda, eds.), Smart Innovation, Systems and Technologies, (Singapore), pp. 29–42, Springer Nature, 2020.

[3] T. Zefferer, V. Krnjic, K. Stranacher, and B. Zwattendorfer, "Measuring Usability to Improve the Efficiency of Electronic Signature-Based e-Government Solutions," in *Measuring E-government Efficiency: The Opinions of Public Administrators and Other Stakeholders* (M. P. Rodríguez-Bolívar, ed.), Public Administration and Information Technology, pp. 45–74, New York, NY: Springer, 2014.

[4] H. Roßnagel, "On Diffusion and Confusion – Why Electronic Signatures Have Failed," in *Trust and Privacy in Digital Business* (S. Fischer-Hübner, S. Furnell, and C. Lambrinoudakis, eds.), Lecture Notes in Computer Science, (Berlin, Heidelberg), pp. 71–80, Springer, 2006.

[5] K. W. Chong, Y. S. Kim, and J. Choi, "A study of factors affecting intention to adopt a cloud-based digital signature service," *Information*, vol. 12, no. 2, p. 60, 2021.

[6] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of management information systems*, vol. 24, no. 3, pp. 45–77, 2007.

[7] A. Hevner, S. Chatterjee, A. Hevner, and S. Chatterjee, "Design science research in information systems," *Design research in information systems: theory and practice*, pp. 9–22, 2010.

[8] The European Parliament and the Council of the European Union, "European Regulation on Electronic Identification and Trust Services for Electronic Transactions in the internal market." Official Journal of the European Union, 2014.

[9] "Electronic Signatures in Global and National Commerce Act." 15 U.S.C. Chapter 96, 2000.

[10] "Uniform Electronic Transactions Act." National Conference of Commissioners on Uniform State Laws, 1999.

[11] M. Wang, "Do the regulations on electronic signatures facilitate international electronic commerce? a critical review," *Computer Law & Security Review*, vol. 23, no. 1, pp. 32–41, 2007.

[12] R. N. Zaeem and K. S. Barber, "The effect of the gdpr on privacy policies: Recent progress and future promise," *ACM Transactions on Management Information Systems (TMIS)*, vol. 12, no. 1, pp. 1–20, 2020.

[13] A. B. Haque, A. N. Islam, S. Hyrynsalmi, B. Naqvi, and K. Smolander, "Gdpr compliant blockchains–a systematic literature review," *IEEE Access*, vol. 9, pp. 50593–50606, 2021.

[14] S. Schwerin, "Blockchain and privacy protection in the case of the european general data protection regulation (gdpr): a delphi study," *The Journal of the British Blockchain Association*, vol. 1, no. 1, 2018.

[15] R. Belen-Saglam, E. Altuncu, Y. Lu, and S. Li, "A systematic literature review of the tension between the gdpr and public blockchain systems," *Blockchain: Research and Applications*, p. 100129, 2023.

[16] M. Söllner, A. Hoffmann, and J. M. Leimeister, "Why different trust relationships matter for information systems users," *European Journal of Information Systems*, vol. 25, no. 3, pp. 274–287, 2016.

[17] M. A. Sasse, "Usability and trust in information systems," in *Trust and Crime in Information Societies*, Edward Elgar, 2005.

[18] M. T. Thielsch, S. M. Meeßen, and G. Hertel, "Trust and distrust in information systems at the workplace," *PeerJ*, vol. 6, p. e5483, 2018.

[19] X. Li, T. J. Hess, and J. S. Valacich, "Why do we trust new technology? a study of initial trust formation with organizational information systems," *The Journal of Strategic Information Systems*, vol. 17, no. 1, pp. 39–71, 2008.

[20] M. J. Casey and P. Vigna, "In blockchain we trust," *MIT Technology Review*, vol. 121, no. 3, pp. 10–16, 2018.

[21] R. Kumar and R. Sharma, "Leveraging blockchain for ensuring trust in iot: A survey," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 10, pp. 8599–8622, 2022.

[22] A. Hasselgren, J.-A. Hanssen Rensaa, K. Kralevska, D. Gligoroski, and A. Faxvaag, "Blockchain for increased trust in virtual health care: Proof-of-concept study," *Journal of Medical Internet Research*, vol. 23, no. 7, p. e28496, 2021.

[23] U. Majeed, L. U. Khan, I. Yaqoob, S. A. Kazmi, K. Salah, and C. S. Hong, "Blockchain for iot-based smart cities: Recent advances, requirements, and future challenges," *Journal of Network and Computer Applications*, vol. 181, p. 103007, 2021.

[24] D. Lawrence and S. Tavakol, "Website usability," *Balanced Website Design: Optimising Aesthetics, Usability and Purpose*, pp. 37–58, 2007.

[25] M. Moniruzzaman, F. Chowdhury, and M. S. Ferdous, "Examining usability issues in blockchain-based cryptocurrency wallets," in *Cyber Security and Computer Science: Second EAI International Conference, ICONCS 2020, Dhaka, Bangladesh, February 15-16, 2020, Proceedings 2*, pp. 631–643, Springer, 2020.

[26] E. Alajrami, B. A. Ashqar, B. S. Abu-Nasser, A. J. Khalil, M. M. Musleh, A. M. Barhoom, and S. S. Abu-Naser, "Handwritten signature verification using deep learning," *International Journal of Academic Multidisciplinary Research*, 2020.

[27] K. Moriarty, M. Nystrom, S. Parkinson, A. Rusch, and M. Scott, "Pkcs# 12: Personal information exchange syntax v1. 1," tech. rep., Internet Engineering Task Force, 2014.

[28] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet x. 509 public key infrastructure certificate and certificate revocation list (crl) profile," tech. rep., Internet Engineering Task Force, 2008.

[29] P. Siriwardena and P. Siriwardena, "Message-level security with JSON web signature," *Advanced API Security: OAuth 2.0 and Beyond*, pp. 157–184, 2020.

[30] H. K. Dhalla, "A Performance Analysis of Native JSON Parsers in Java, Python, MS. NET Core, JavaScript, and PHP," in *2020 16th International Conference on Network and Service Management (CNSM)*, pp. 1–5, IEEE, 2020.

[31] K. P. Gaffney, M. Prammer, L. Brasfield, D. R. Hipp, D. Kennedy, and J. M. Patel, "Sqlite: past, present, and future," *Proceedings of the VLDB Endowment*, vol. 15, no. 12, pp. 3535–3547, 2022.

[32] D. P. Bauer, "Nethereum," in *Getting Started with Ethereum: A Step-by-Step Guide to Becoming a Blockchain Developer*, pp. 119–123, Springer, 2022.

[33] D. Vujičić, D. Jagodić, and S. Randić, "Blockchain technology, bitcoin, and ethereum: A brief overview," in *2018 17th international symposium infoteh-jahorina (infoteh)*, pp. 1–6, IEEE, 2018.

[34] L. Marchesi, M. Marchesi, G. Destefanis, G. Barabino, and D. Tigano, "Design patterns for gas optimization in ethereum," in *2020 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, pp. 9–15, IEEE, 2020.

[35] E. Albert, J. Correas, P. Gordillo, G. Román-Díez, and A. Rubio, "Gasol: Gas analysis and optimization for ethereum smart contracts," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pp. 118–125, Springer, 2020.

[36] B. Severin, M. Hesenius, F. Blum, M. Hettmer, and V. Gruhn, "Smart money wasting: Analyzing gas cost drivers of ethereum smart contracts," in *2022 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pp. 293–304, IEEE, 2022.

[37] O. Dib, K.-L. Brousmiche, A. Durand, E. Thea, and E. B. Hamida, "Consortium blockchains: Overview, applications and challenges," *Int. J. Adv. Telecommun*, vol. 11, no. 1, pp. 51–64, 2018.