What Does Not Fit Can be Made to Fit!
Trade-Offs in Distributed Ledger Technology Designs

Niclas Kannengießer
Karlsruhe Institute of Technology,
University of Kassel
niclas.kannengiesser@kit.edu

Sebastian Lins
Karlsruhe Institute of Technology
sebastian.lins@kit.edu

Tobias Dehling
Karlsruhe Institute of Technology
dehling@kit.edu

Ali Sunyaev
Karlsruhe Institute of Technology
sunyaev@kit.edu

Abstract

Distributed ledger technology (DLT), including blockchain, enables secure processing of transactions between untrustworthy parties in a decentralized system. However, DLT is available in different designs that exhibit diverse characteristics. Moreover, DLT characteristics have complementary and conflicting interdependencies. Hence, there will never be an ideal DLT design for all DLT use cases; instead, DLT implementations need to be configured to contextual requirements. Successful DLT configuration requires, however, a sound understanding of DLT characteristics and their interdependencies. In this manuscript, we review DLT characteristics and organize them into six groups. Furthermore, we condense interdependencies of DLT characteristics into trade-offs that should be considered for successful deployment of DLT. Finally, we consolidate our findings into DLT archetypes for common design objectives, such as security, usability, or performance. Our work makes extant DLT research more transparent and fosters understanding of interdependencies and trade-offs between DLT characteristics.

1. Introduction

Distributed ledger technology (DLT), including blockchain, enables secure transactions between untrustworthy parties through algorithm-based consensus. The automated consensus finding eliminates the need for third-party trust enforcement. DLT is promising to automate and speed up information processing while simultaneously decreasing transaction cost. Consequently, organizations from diverse industries have strong interest in the application of DLT in domains such as supply chain management [1], micro and smart grids [2], and internet-of-things (IoT) [3]. Due to differences in application domains, corresponding DLT use cases come with context-depended requirements that necessitate individual configuration of DLT characteristics [4]. Exemplary context-depended requirements are high scalability [3, 5], high throughput [3, 6], fast (micro) transactions, and a high level of anonymity [3, 7] or security [8, 9]. The configuration of DLT characteristics to meet such context-depended requirements is becoming a key challenge for the improvement of DLT concepts and concrete DLT designs.

However, DLT characteristics are highly dependent on each other. Thus, improving certain DLT characteristics will deteriorate other DLT characteristics. For instance, Bitcoin is well known for its high availability and resilience with respect to fraud due to its high number of participating nodes. On the other hand, Bitcoin suffers from low throughput (only seven transactions per second) and from low energy efficiency due to proof-of-work consensus finding. This makes Bitcoin, for instance, unsuitable for the IoT domain, which requires an ever-increasing transaction speed. In contrast, HyperLedger Fabric is capable of 3,500 transactions per second on average but lacks high scalability and supports only a small set of nodes, which reduces availability and resilience compared to Bitcoin. Consequently, DLT developers are currently struggling with the interdependencies of DLT characteristics and resulting trade-offs that need to be tackled when designing and developing DLT.

Extant research on DLT can mainly be distinguished into three streams: First, application of DLT for several use cases [3]; second, classification and (formal) description of DLT designs [10]; third, further development of DLT concepts and designs [11]. Within these streams, extant research focuses mostly on tailoring DLT to specific contexts and employs a particular DLT design to meet the requirements in the chosen context. Trade-offs resulting from interdependencies between DLT characteristics are often neglected and not further investigated. Consequently,

URI: https://hdl.handle.net/10125/60143
ISBN: 978-0-9981331-2-6

interdependencies and resulting trade-offs between DLT characteristics remain unclear. Because of the inseparable and interwoven nature of DLT characteristics, a holistic view on the interaction and mutual interdependencies of DLT characteristics is necessary to understand resulting constraints on the usefulness of specific DLT designs in different domains. The objective of our research is to identify trade-offs between DLT characteristics by answering the following research question:

RQ: What trade-offs result from interdependencies of DLT characteristics?

To answer the research question, we first provide a comprehensive overview of DLT characteristics by surveying prior literature. Then, we identify and discuss trade-offs of DLT characteristics. Finally, we consolidate our findings into archetypes of prominent DLT designs. These DLT archetypes are useful to assess DLT designs with respect to suitability for particular DLT use cases.

Our work makes extant DLT research more transparent and fosters understanding of interdependencies and trade-offs between DLT characteristics. This is useful to select fitting DLT configurations for certain uses cases, to estimate likely limitations and constraints, and to gauge the risks coming with the choice of a particular DLT design.

The remainder of this manuscript is structured as follows. In section 2 related research are presented. Subsequently, the employed methodology is described in section 3. We present an overview of the identified DLT characteristics and trade-offs in section 4. In section 5, we discuss our results, present archetypes for DLT designs, and conclude with implications and directions for future research.

2. Theoretical background

2.1 Distributed ledger technology

DLT facilitates consensus finding in a distributed ledger maintained by a decentralized network. Each node in the decentralized network stores, shares, and synchronizes digital data [12].

Digital data to be included in the ledger is submitted by users in form of transactions. A new transaction is received by a single node of the distributed ledger, which forwards the transaction to each node in the network. Depending on the DLT design, validity of transactions is assessed by a member of nodes participating in the ledger [12]. After a transaction is validated, it is appended to the distributed ledger. As transactions and its corresponding data can only be appended to the distributed ledger, it is hard to alter the data retroactively. Thus, a basic DLT characteristic is its high degree of integrity, which ensures immutability of recorded data.

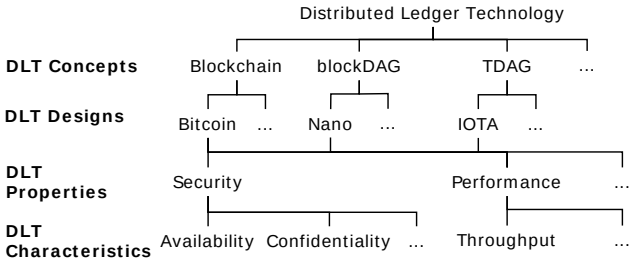

Figure 1. Schematic overview about the DLT terminology

DLT includes different DLT concepts, which mainly differ in the way transactions are validated and stored. Some popular DLT concepts are blockchain [12], block directed acyclic graphs (blockDAG) [3], and transaction-based directed acyclic graphs (TDAG) [3]. The DLT concept blockchain, for example, is employed by Bitcoin and Ethereum, yet they differ from each other in their DLT design, which is a concrete implementation of a DLT concept. Although all DLT designs have DLT properties including security and performance [3, 10, 13, 14, 15], each DLT design has individual configurations of DLT characteristics. A DLT characteristic is a characteristic, which is configurable and crucial to a DLT design's suitability for given use cases.

Figure 1 provides a visualization of the differences between DLT concepts, designs, and characteristics, which are associated with DLT properties. In addition, current development on DLT is illustrated in *Figure 2*, highlighting the relations between DLT concepts, DLT designs, DLT characteristics and the corresponding DLT properties.

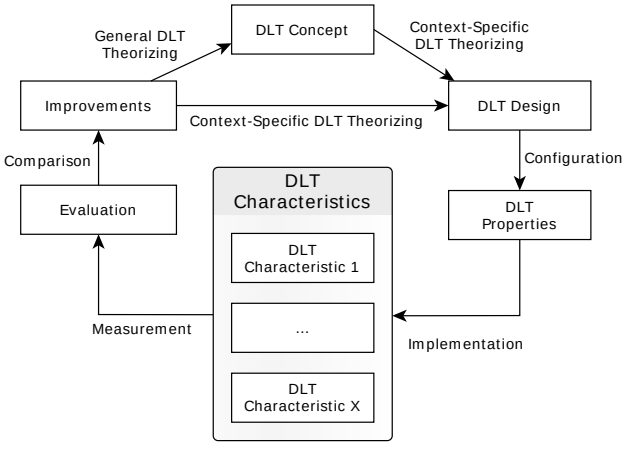

Figure 2. Current DLT development cycle for the improvement of DLT

2.2 Related work

Extant research on DLT can be distinguished into three main streams: Application of DLT in several use cases, classification and (formal) description of DLT designs, and improvements on and further developments of DLT concepts and DLT designs. Use cases and implemented DLT applications offer insights on application requirements and on constraints imposed due to chosen DLT designs [1, 6, 16]. In the second stream, taxonomies, classifications and analyses of DLT designs, which offer an overview about commonalities and differences of DLT designs with respect to DLT characteristics, are developed [9, 10, 17, 18, 19, 20]. Additionally, analysis of DLT designs reveals scenarios for successful attacks such as balance attacks [18] or selfish-mining [17]. However, these analysis focus on single implications and do not provide an overview of other implications. Last, further development focuses further improvement of existing DLT designs and developing evaluation frameworks for DLT designs.

Previous literature focuses on increasing the suitability of DLT designs for specific use cases. Hence, trade-offs between DLT characteristics are often intuitively accepted and not thoroughly examined. Constraints of DLT designs are listed but causes, interdependencies and trade-offs between DLT characteristics are not investigated. These trade-offs strongly influence a DLT design's suitability for a particular use case. It is crucial to investigate (unintended) effects of particular configurations on DLT characteristics on other DLT characteristics. We provide an overview of DLT characteristics that determine the suitability of DLT designs for particular use-cases. Built on the identified DLT characteristics, we then discuss causes that induce trade-offs between DLT characteristics.

3. Methodology

We apply a two-step research approach. We first conduct a literature review to extract DLT characteristics. Our descriptive literature review [21] was guided by recommendations for literature reviews in the information systems domain [22, 23, 24]. We analyze DLT characteristics in detail to identify trade-offs in DLT designs.

3.1 Literature review

To identify publications addressing DLT characteristics, we searched scientific databases that we deemed representative for the identification of DLT characteristics and related interdependencies as they cover a wide range of journals and conferences (i.e., they cover the top computer science and information systems journals and conferences): ACM Digital Library, EBSCOhost, IEEE Xplore, ProQuest, and ScienceDirect. To cover a broad set of publications, we searched each database with the following string in title, abstracts and keywords: (blockchain* OR (distributed AND ledger*)). We limited our search to peer-reviewed articles to ensure high quality of articles. We identified 1,144 articles in this initial search. To identify and filter articles, we first checked the relevance of each article by analyzing title, abstract, and keywords. If any indication for relevance appeared, the article was marked for further analysis. We excluded articles that were duplicates (62), grey literature (i.e., editorials, unfinished manuscripts, dissertations) and books (18), not applicable to our study (56) or not available in English (31). This first relevancy assessment resulted in a sample of 977 articles deemed to be potentially relevant. Afterwards, a fine-grained relevance validation was made by reading the articles in detail, resulting in a final sample of 195 relevant articles. In this second relevance assessment, we excluded articles that do not relate to suitability of DLT characteristics for various use cases (706) or non-research articles (76).

3.2 Data analysis

Our data analysis followed an approach proposed by Lacity et al. [25]. As a first step, we carefully read and analyzed relevant articles to identify the considered DLT characteristics. We recorded for each extracted DLT characteristic a name, a description and the original source [25]. In total, 277 DLT characteristics were extracted. A list of so-called master variables was created to aggregate the identified DLT characteristics [25]. A master variable is an aggregation of similar, DLT characteristics consisting of a name and a description (see the bootstrapping approach in [26]). If an identified DLT characteristic fitted into an existing master variable, we assigned it accordingly; otherwise, a new master variable was created. During this process, we applied the coding rules proposed by Lacity et al. [25]. Since different people often put the same labels on different things and vice versa, it is crucial for the validity of a qualitative analysis to avoid semantic ambiguities (i.e., different terminology for same concepts) [27]. For example, we aggregated the DLT characteristics "immutability" and "tamper-resistance" to the master variable "integrity". To ensure that we identified a reliable set of master variables, we aimed for theoretical saturation [28, 29], that is, the point when no new

findings are gained in further articles. After completing the analysis of 50 articles, randomly selected out of the 195 relevant articles, we noticed that no new master variable emerged in the last 16 articles. Given this high number of articles that did not lead to the identification of any new master variable in our literature review, we were confident to have reached saturation and therefore stopped our literature review. We finalized the list of master variables by reviewing all assignments. To ease understanding, we use the term DLT characteristics for the identified master variables in the following, as they represent an aggregation of similar DLT characteristics.

We applied an inductive approach grouping DLT characteristics by objectives and application contexts. For instance, DLT characteristics were grouped into the DLT property security if they were related to common security topics such as availability and confidentiality.

Subsequently, trade-offs between DLT characteristics were extracted from the literature. Especially constrains on DLT designs and their specific origins were analyzed. Furthermore, interdependencies between characteristics were coded and structured to examine trade-offs between DLT characteristics, even if the underlying interdependencies are not obvious.

4. Results

4.1 DLT characteristics

Our study identified 37 DLT characteristics, which determine the suitability of DLT designs for specific contextual requirements. These DLT characteristics are described in *Table 2*. The inductive grouping of DLT characteristics resulted in the six DLT properties summarized in *Table 1*. These DLT properties are crucial for all examined DLT designs.

4.2 Trade-offs between DLT characteristics

DLT characteristics have interdependencies, and can either be complementary (e.g., a high level of transparency supports auditability) or conflicting (e.g., high availability requires multiple replications of the ledger but comes with the cost of decreased consistency). Interdependencies between DLT characteristics result in trade-offs, which constitute an improvement of one DLT characteristic at the cost of deteriorating another DLT characteristic. Therefore, trade-offs between DLT characteristics result in constraints on the applicability of DLT designs for certain use cases.

Table 1. Identified DLT properties

Security (181 occurrences)
Preservation of confidentiality, integrity, and availability of information [30].
Performance (92 occurrences)
The accomplishment of a given task measured against standards of accuracy, completeness, costs, and speed.
Usability (51 occurrences)
The extent to which a DLT design can be used by specified users to achieve specified goals with respect to effectiveness, efficiency, and satisfaction in a context of use [31].
Development Flexibility (34 occurrences)
The possibilities offered by a DLT design for maintenance and further development.
Level of Anonymity (23 occurrences)
The degree to which individuals are not identifiable within a set of subjects [32].
Institutionalization (16 occurrences)
The emerging embedding of concepts and artifacts (here DLT) in social structures.

Security vs. Institutionalization

Confidentiality vs. Auditability. A high degree of confidentiality comes with granular access rights to saved data, which impedes auditability of transaction contents due to a loss of transparency [35].

Vulnerability Resistance vs. Auditability. In distributed ledgers using equity tokens (e.g., cryptocurrencies) it is crucial to be able to determine the current amount of equity tokens in the system to discern value of the equity tokens. Strong encryption of transactions, which are used to reconstruct the amount of equity tokens owned by users, enables a low level of transparency but impedes auditability. Hence, the total number of equity tokens becomes hard to determine, which makes hacks on the amount of equity tokens also hard to detect.

Security vs. Usability

Availability vs. Costs. High availability is reached by a high number of replications of the ledger on several nodes. Reducing costs by reducing the number of nodes results in a reduction of the number of replications of the distributed ledger and weakens availability. A high number of replications is currently reached in public, unpermissioned blockchains because any node can join the distributed ledger. However, public blockchains also come with higher transaction fees than private or permissioned blockchains [33] resulting in higher overall costs for users.

Security vs. Security

Consistency vs. Availability. Distributed database theory reveals a trade-off between consistency and availability—the CAP Theorem [34]. This trade-off persists in the field of DLT and is caused by latency in block propagation, for example, due to big block sizes or network failures. The larger the number of

Table 2. DLT characteristics grouped by DLT properties

Prop	DLT Characteristics
Security	*Availability*. Availability is the probability that a system can be accessed when needed [15].
	Confidentiality. Prevention of unauthorized information access and release [36, 37, 38].
	Consistency. Strong consistency means that all nodes store the same data in their ledger at the same time [39].
	Integrity. Integrity requires that information is protected against unauthorized modification or deletion as well as irrevocable, accidental, and undesired changes by authorized users [3, 40, 41].
	Level of Encryption. The level of security concerning the application of authentication-related cryptographic primitives in, for example, creation of public/private-key pairs and authentication for transaction authentication [42, 43].
	Level of Decentralization. The number of independent nodes participating in transaction validation and consensus finding [3, 14].
	Level of Trust towards Nodes. The level of how trustworthy each node in the distributed network is [44, 45].
	Likelihood of Forks. A fork is the existence of a branch besides the main branch of a distributed ledger [17, 18].
	Non-Repudiation. Entities involved in a communication cannot deny having participated in all or part of the communication [15, 33, 46].
	Partition Tolerance. The system continues to operate correctly even if an arbitrary number of messages is dropped (or delayed) by the network [40].
	Resilience. The ability to return to a (previous) state after the occurrence of some event or action which may have changed that state [47].
	Vulnerability Resistance. The system's degree of vulnerability to targeted attacks [48].
Performance	*Block Creation Interval*. The time between the creation of consecutive blocks (only in DLT designs using blocks) [15].
	Block Size. The size of data that is stored in a block [7, 15].
	Energy Efficiency. A number expressing the relative efficiency of a tool, such as the number of validated transactions, that is obtained by dividing the tool's output per hour by its energy requirement in watts that is consumed by computation for processes, such as mining and transaction validation [9].
	Propagation Delay. Latency between the submission of a transaction (or block) and the point in time where each node received the transaction [3, 42].
	Required Bandwidth. The bandwidth the DLT design's protocol requires for necessary data exchanges over the decentralized network [49].
	Scalability. The capability of a DLT design to handle an increasing amount of workload or its potential to be enlarged to accommodate that growth [5, 50].
	Throughput. The number of transactions validated and appended to the ledger in a given time interval [5, 8, 39, 51].
	Transaction Validation Speed. Duration required for verifying transaction validity [8, 42].
Usability	*Costs*. Costs related to the implementation and usage of a DLT design, including software development and operational costs [52, 53, 54, 55].
	Ease of Node Adoption. The ease of preparing a new or failed device to be added to the DLT design in the role of a validating node or a consuming terminal device [44, 56, 57].
	Ease of Use. The ability to easily access and work with the DLT design [57, 58].
	Support for Constrained Devices. The extent to which constrained devices, such as those used in the IoT, can participate in a DLT, for instance, by issuing or validating transactions [6, 59, 60].

nodes that must receive new transactions, the longer the distributed ledger is in an inconsistent state.

Performance vs. Security

Block Size vs. Consistency. An increased block size comes with a higher block propagation delay resulting in a longer state of inconsistency between nodes in a distributed ledger. This trade-off has also been found in the field of distributed databases [65].

Block Size vs. Integrity. In distributed ledgers using blocks, block propagation delays are strongly influenced by block size [3]. The longer the block propagation delay, the higher is the probability of new forks [15]. Forks increase the probability of immutability breaches [67] caused by attacks such as selfish-mining [61], which impede integrity.

Block Size vs. Vulnerability Resistance. By increasing block size in a block-based distributed ledger, more transactions can be stored in a single block, which causes longer block propagation delays [3] and increased required bandwidth [53]. Highly varying loads on the distributed ledger caused by variations on transaction frequency result in block size variations, which cause variations in block propagation delay in the network [45]. Variations in block propagation delay increase the probability of successful selfish-mining attacks thereby threatening security [17, 61]. Selfish-mining attacks describe a phenomenon where a pool of nodes mines its own branch of a blockchain without publishing their blocks to the main branch until their selfish-mined branch would be chosen as future main branch by the particular fork resolution rules [61]. Thus, the mining pool can revert blockchain contents.

Table 2 continued. DLT characteristics grouped by DLT properties

Prop	DLT Characteristics
Development Flexibility	*Interoperability*. Ability to perform interchain exchanges and the ability to communicate with external services [52].
	Level of Modularity. The extent to which modules of a DLT design can be exchanged (e.g., consensus mechanism) [5, 8, 11, 62].
	Maintainability. Degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers [9, 63].
	Smart Contract Support. The degree of how well smart contracts are supported by a DLT design including expressiveness of supported programming languages [64] and availability of test and development environments [51, 64, 65].
	Transaction Size. The presence of a fixed maximum size for a transaction [15].
LoA[1]	*Transparency*. The ability to publicly view and trace an account's holdings on a distributed ledger [7, 33, 43].
	Unidentifiability. The state of being unidentifiable within a set of subjects [7, 33].
Institutionalization	*Auditability*. The ability of a distributed ledger to be audited regarding technical features and contents by an external party such as a state institution [33, 43].
	Censorship-Resistance. The equal right of any user of the distributed ledger to submit transactions that are not altered or dropped by a third party [33, 45].
	Compliance. Fulfillment of regulatory requirements or best practices [33, 56].
	Development Activity on the DLT design. Amount of code updates for the DLT design and size and activity of the foundation and community associated with the DLT design [9].
	Openness. The extent to which new nodes can join the distributed ledger without being verified [15].
	Responsibility for Functionality. Existence of an enterprise, foundation, or organization that creates a DLT design's underlying code and is responsible for its maintenance and functionality [19, 66].

[1] *Level of Anonymity*

Block Creation Interval vs. Level of Decentralization. In Proof-of-Work, a long block creation interval leads to less often rewards in total and decreased likelihood of rewards for individual miners. This contributes to high variance in received payments for miners. Hence, it is more likely that nodes will join mining pools to increase the probability to receive rewards from mining. This leads to a decreased level of decentralization.

Throughput vs. Integrity. Higher throughput can be reached by a smaller set of verified nodes that validate transactions. Hence, a small number of known nodes makes it easier to have detailed information on the network graph. Access to a detailed network topology facilitates initiation of targeted delays in the communication between nodes because the data flow is known [18]. Thus, the probability for successful balance attacks [18] increases in forkable DLT designs. Balance attack can be defined as the process of transiently disrupting communications between subgroups of miners with equal mining power [18]. During the communications is disrupted, transactions can be submitted to one subgroup while mining new blocks in another subgroup. The attacker's aim is to outweigh the blockchain branch she submitted transactions to with the blockchain branch she participates in the mining process. As a result, the ledger may be rewritten [18]. Balance attacks raise the probability for successful double-spending, which violates a ledger's immutability. Increased vulnerability to immutability violations reduces the integrity of a distributed ledger.

Throughput vs. Partition Tolerance. By decreasing the number of validating nodes in DLT designs using blocks, faster consensus algorithms (e.g., Practical Byzantine Fault Tolerance), which scale well, can be applied instead of slow consensus mechanisms such as Proof-of-Work. On the other hand, decreasing the number of validating nodes in a distributed ledger reduces its partition tolerance because less nodes forward new transactions to foreign nodes.

Development Flexibility vs. Performance.

Smart Contract Support vs. Required Bandwidth. The more complex smart contracts are and the more functionality they must provide, the more likely is an increase in the required size of a transaction data storage. Hence, the required bandwidth must increase to prevent decreased consistency.

Smart Contract Support vs. Transaction Validation Speed. The support for more expressive programming languages enables development of smart contracts providing a broad range of functionality. The more functionality is added to a smart contract, the higher becomes its complexity, ultimately impeding DLT performance because the required execution time and, consequently, the time required for transaction validation increases [49]. A smart contract's complexity may be dramatically increased by using external data sources (e.g., from an oracle). The used compiler further influences the smart contract's runtime. This is because a compiler translates human-readable code into machine code. The resulting machine code should be executable by a computer as fast as possible [56, 68].

Development Flexibility vs. Security

Smart Contract Support vs. Vulnerability Resistance. Greater support of smart contracts enables more flexibility in developing applications on DLT. The more flexibility developers have when developing smart contracts, the more software errors (i.e., bugs) may occur, which harm vulnerability resistance of applications integrating such flawed smart contracts [42].

Smart Contract Support vs. Vulnerability Resistance. A greater support for expressive programming languages (e.g., Java, Go, Python) creates more opportunities for third parties to write exploits that could compromise nodes within a distributed ledger (e.g., through a smart contract). Vulnerabilities in a DLT design's environment (e.g., virtual machines hosting smart contracts) must be regularly secured and maintained to keep the distributed ledger secure.

Level of Anonymity vs. Development Flexibility

Unidentifiability vs. Maintainability. A higher level of anonymity is enabled by a public and unpermissioned DLT design because nodes must not be verified before joining the distributed ledger. On the other hand, updates of DLT code must be updated by the majority of nodes in the whole network to keep compatibility and guarantee up-to-dateness [9]. It is hard to maintain the usually large number of nodes in public, unpermissioned distributed ledgers, which decreases the level of development flexibility. In contrast, permissioned, private DLT designs, are more flexible because each node must be verified and is identified before joining the distributed ledger. Since each participating node is known and the number of validating nodes in the distributed ledger is usually small, maintenance of the nodes is easier, which results in a higher level of development flexibility.

Level of Anonymity vs. Performance

Unidentifiability vs. Throughput. The less a network is controlled by a central authority and the more nodes participate in the network, the higher the possible anonymity level for users. Therefore, public, unpermissioned distributed ledgers promise more anonymity than permissioned ones. In contrast, a smaller network with verified and identifiable nodes is considered as providing higher throughput because faster consensus algorithms can be used (e.g., Byzantine Fault Tolerance). Unidentifiability can also be reached by applying additional processes like mixing and the use of new key-pairs per transaction [69]. These processes create overhead by preprocessing each transaction, which results in decreased transaction validation speed.

Level of Anonymity vs. Usability

Unidentifiability vs. Support for Constrained Devices. To achieve unidentifiability there are two options. First, additional data structures can be used, which require additional storage size [7, 53]. Second, additional processes such as mixing can be applied resulting in additional demand for computational power. Both approaches for achieving unidentifiability come at the cost of additional demands on computational power, which weakens the distributed ledger's support for constrained devices.

Institutionalization vs. Level of Anonymity

Auditability vs. Unidentifiability. Auditability requires readability of transaction contents and the possibility to associate transaction contents with particular actors. As unidentifiability conceals actors, auditability of user activities becomes very difficult or even impossible.

5. Discussion

Our study presents 37 DLT characteristics that determine the suitability of DLT designs for specific use cases. Through the analysis of the DLT characteristics we revealed 18 trade-offs between the DLT characteristics. Our findings shed light on interactions between DLT characteristics that are inherent to DLT designs and elucidate several constraints on DLT designs. Based on our results, improvement of DLT designs can be taught in a more holistic way by both researchers and practitioners. A schematic illustration of our findings can be found in *Figure 3*. Furthermore, the explanation of the identified trade-offs supports practitioners in assessing suitability of DLT designs for use cases and future risks related to the choice of a DLT Design.

Our results indicate that trade-offs between DLT properties are not of equal strength. While there are major conflicts between the DLT properties *security* and *performance* that are present in all DLT designs, there are also conflicts, whose importance varies with the specific use case. For instance, the trade-off between *institutionalization* and *level of anonymity* is only relevant if auditability is required. It is noticeable that security directly conflicts with all DLT properties but *level of anonymity*. However, there are mediated trade-offs via performance, development flexibility, or institutionalization. This implies that a high level of security comes at the cost of all other DLT characteristics.

To make our findings even more comprehensible and easily applicable to DLT designs, we consolidated the identified trade-offs between DLT characteristics into archetypes.

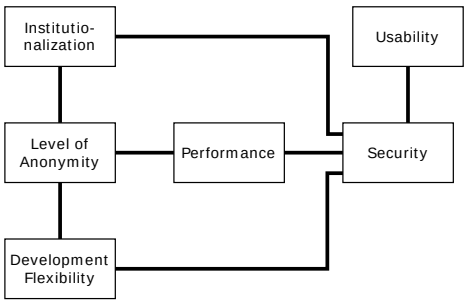

Figure 3. Schematic illustration of trade-offs between DLT properties

DLT Archetypes

Maximum Usability DLT. This type of DLT offers a maximum level of usability to consumers. To make applications integrating DLT capable for constrained devices, a full replication of the ledger on each device should be avoided, for example, because of constrained storage sizes. To make a DLT design suitable for constrained devices, DLT designs can use light nodes. Light nodes do not contribute to increased availability or resilience of the distributed ledger because they do not store a full replication of the ledger and do not validate transactions.

Maximum Development Flexibility DLT. Great development flexibility can be achieved through support of smart contracts that can be individually deployed. The more expressive supported programming languages are, the more bugs may be exhibited by smart contracts. Smart contract code is hard to review and to test, especially, with respect to chained execution. Hence, smart contracts increase the risks for security breaches (e.g., The DAO attack [70]).

Maximum Performance DLT. High performance requires a maximum number of transactions per seconds. To achieve that goal, a minimal complexity of the employed consensus algorithm and encryption approaches is necessary. Additionally, a smaller number of validating nodes speeds up system throughput. Yet, a smaller number of nodes decreases security of a DLT and requires a higher level of trust for the nodes. This requires verification of nodes and, hence, the loss of their anonymity. Furthermore, participating nodes do not need to contain a whole replication of the ledger. This type of DLT design can be used by even strongly-constrained devices, which lowers the number of replications of the system and consequently the system's resilience.

Maximum Anonymity DLT. To reach a high level of anonymity, additional processing of transactions is necessary (e.g., mixing, heavier encryption). These processes are time-consuming and require additional computational power, which slows down performance. Great network size comes with a higher level of anonymity, but performance is deteriorated in block-based distributed ledgers. Furthermore, auditability is limited or even impossible because transactions cannot be traced back to the issuing user.

Maximum Security DLT. A maximum of security in DLT is reached by increasing network size, excluding possibly fraudulent nodes, and reducing development flexibility for smart contracts. These approaches contradict. For instance, the nodes' level of anonymity is decreased to prevent attacks by fraudulent nodes.

Maximum Institutionalization DLT. Institutionalization requires a high level of auditability and compliance. Auditability comes with high transparency, which results in a low level of anonymity. The level of compliance of any system always depends on current standards and regulations. Standards and regulations can be changed, and systems must adapt to changes to retain their level of compliance. A basic characteristic of DLT designs is immutability. Hence, the later adaptation to changes concerning compliance becomes hard. Therefore, a high level of immutability rules out adaptations towards a high level of compliance without a decrease in integrity.

6. Conclusion

Our research aims to better understand trade-offs in DLT designs by providing a comprehensive list of DLT characteristics, a discussion on causes for emerging trade-offs, and the proposition of six archetypes. We contribute to research and practice in several ways. First, by discussing potential trade-offs in DLT designs, we provide a holistic view on the interaction and mutual interdependencies of DLT characteristics. This holistic view is valuable to understand resulting constraints on the usefulness of specific DLT designs in different domains. This supports research and practice in assessing DLT limitations and to gauge the risks resulting from the choice of a particular DLT design. Second, the presented six archetypes can be used as a reference to obtain an impression of technical implications of a chosen DLT design and to ease the assessment of a DLT design's suitability.

However, current approaches for measuring DLT characteristics must be further developed to integrate the identified interdependencies and trade-offs to quantify their strength. Therefore, future research should examine methods for measuring impacts of DLT characteristics and the implications of different configurations of DLT characteristics.

7. References

[1] Chen, S., R. Shi, Z. Ren, J. Yan, Y. Shi, and J. Zhang, "A Blockchain-Based Supply Chain Quality Management Framework", *IEEE 14th International Conference on e-Business Engineering*, (2017), 172–176.

[2] Goranović, A., M. Meisel, L. Fotiadis, S. Wilker, A. Treytl, and T. Sauter, "Blockchain Applications in Microgrids an Overview of Current Projects and Concepts", *43rd Annual Conference of the IEEE Industrial Electronics Society*, (2017), 6153–6158.

[3] Yeow, K., A. Gani, R.W. Ahmad, J.J.P.C. Rodrigues, and K. Ko, "Decentralized Consensus for Edge-Centric Internet of Things: A Review, Taxonomy, and Research Issues", *IEEE Access 6*, 2018, pp. 1513–1524.

[4] Labazova, O., T. Dehling, and A. Sunyaev, "From Hype to Reality: A Taxonomy of Blockchain Applications", *52th Hawaii International Conference on System Sciences*, (2019).

[5] Li, W., A. Sforzin, S. Fedorov, and G.O. Karame, "Towards Scalable and Private Industrial Blockchains", *ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, (2017), 9–14.

[6] Huh, S., S. Cho, and S. Kim, "Managing IoT devices using blockchain platform", *19th International Conference on Advanced Communication Technology*, (2017), 464–467.

[7] Khalilov, M.C.K., and A. Levi, "A Survey on Anonymity and Privacy in Bitcoin-like Digital Cash Systems", *IEEE COMMUN SURV TUT*, 2018, pp. 1–44.

[8] Dinh, T.T.A., J. Wang, G. Chen, R. Liu, B.C. Ooi, and K.-L. Tan, "BLOCKBENCH: A Framework for Analyzing Private Blockchains", *ACM International Conference on Management of Data*, (2017), 1085–1100.

[9] Yli-Huumo, J., D. Ko, S. Choi, S. Park, and K. Smolander, "Where Is Current Research on Blockchain Technology?—A Systematic Review", *PLOS ONE 11*(10), 2016, pp. 1–27.

[10] Glaser, F., and L. Bezzenberger, "Beyond Cryptocurrencies - A Taxonomy of Decentralized Consensus Systems", *23rd European Conference on Information Systems*, (2015), 1–18.

[11] Androulaki, E., A. Barger, V. Bortnikov, et al., "Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains", *13th EuroSys Conference*, (2018), 1–15.

[12] Nakamoto, S., "Bitcoin: A Peer-to-Peer Electronic Cash System", 2008. https://bitcoin.org/bitcoin.pdf

[13] Kaushik, A., A. Choudhary, C. Ektare, D. Thomas, and S. Akram, "Blockchain – Literature survey", *2nd IEEE International Conference on Recent Trends in Electronics, Information Communication Technology*, (2017), 2145–2148.

[14] Li, X., P. Jiang, T. Chen, X. Luo, and Q. Wen, "A survey on the security of blockchain systems", *FUTURE GENER COMP SY*, 2017, pp. 1–13.

[15] Xu, X., I. Weber, M. Staples, et al., "A Taxonomy of Blockchain-Based Systems for Architecture Design", *IEEE International Conference on Software Architecture*, (2017), 243–252.

[16] Zupan, N., K. Zhang, and H.-A. Jacobsen, "Hyperpubsub: a decentralized, permissioned, publish/subscribe service using blockchains: demo", *18th ACM/IFIP/USENIX Middleware Conference: Posters and Demos*, (2017), 15–16.

[17] Göbel, J., H.P. Keeler, A.E. Krzesinski, and P.G. Taylor, "Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay.", *Performance Evaluation 104*, 2016, pp. 23–41.

[18] Natoli, C., and V. Gramoli, "The Balance Attack or Why Forkable Blockchains are Ill-Suited for Consortium", *47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, (2017), 579–590.

[19] Tschorsch, F., and B. Scheuermann, "Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies", *IEEE COMMUN SURV TUT 18*(3), 2016, pp. 2084–2123.

[20] Watanabe, H., S. Fujimura, A. Nakadaira, Y. Miyazaki, A. Akutsu, and J. Kishigami, "Blockchain contract: Securing a blockchain applied to smart contracts", *IEEE International Conference on Consumer Electronics*, (2016), 467–468.

[21] Paré, G., M.C. Trudel, M. Jaana, and S. Kitsiou, "Synthesizing information systems knowledge: A typology of literature reviews", *Information and Management 52*(2), 2015, pp. 183–199.

[22] Kitchenham, B., O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic Literature Reviews in Software Engineering - A Systematic Literature Review", *INFORM SOFTWARE TECH 51*(1), 2009, pp. 7–15.

[23] vom Brocke, J., A. Simons, K. Riemer, B. Niehaves, and R. Platfaut, "Standing on the Shoulders of Giants: Challenges and Recommendations of Literature Search in Information Systems Research.", *Communications of the Association for Information Systems 37*(9), 2015, pp. 205–224.

[24] Webster, J., and R.T. Watson, "Analyzing the Past to Prepare for the Future: Writing a Literature Review", *MIS Q. 26*(2), 2002, pp. 13–23.

[25] Lacity, M.C., S. Khan, A. Yan, and L.P. Willcocks, "A review of the IT outsourcing empirical literature and future research directions", *J INF TECHNOL 25*(4), 2010, pp. 395–433.

[26] Jankowicz, D., *The easy guide to repertory grids*, Wiley, Chichester, West Sussex, England ; Hoboken, N.J, 2004.

[27] Shaw, M.L.G., and B.R. Gaines, "Comparing conceptual structures: consensus, conflict, correspondence and contrast", *Knowledge Acquisition 1*(4), 1989, pp. 341–363.

[28] Corbin, J.M., and A.L. Strauss, *Basics of qualitative research: techniques and procedures for developing grounded theory*, Los Angeles, 2015.

[29] Glaser, B.G., and A.L. Strauss, *The discovery of grounded theory: strategies for qualitative research*, Aldine, New Brunswick, 2009.

[30] ISO/IEC, "Information technology – Security techniques – Information security management systems – Overview and vocabulary", 2009.

[31] ISO/DIS, "Ergonomics of human-system interaction – Part 11: Usability: Definitions and concepts", *ISO 9241-11*.

[32] Pfitzmann, A., M. Hansen, and M. Köhntopp, "Anonymity, Unlinkability, Unobservability, Pseudonymity, and Identity Management – A Consolidated Proposal for Terminology", 2002. https://www.researchgate.net/publication/228622491_Anonymity_Unlinkability_Undetectability_Unobservability_Pseudonymity_and_Identity_Management-A_Consolidated_Proposal_for_Terminology

[33] Neisse, R., G. Steri, and I. Nai-Fovino, "A Blockchain-based Approach for Data Accountability and Provenance Tracking", *12th International Conference on Availability, Reliability and Security*, (2017), 1–10.

[34] Abadi, D., "Consistency Tradeoffs in Modern Distributed Database System Design: CAP is Only Part of the Story", *Computer 45*(2), 2012, pp. 37–42.

[35] Kim, H., and M. Laskowski, "A Perspective on Blockchain Smart Contracts: Reducing Uncertainty and Complexity in Value Exchange", *26th International Conference on Computer Communication and Networks*, (2017), 1–6.

[36] Cherdantseva, Y., and J. Hilton, "A Reference Model of Information Assurance & Security", *International Conference on Availability, Reliability and Security*, (2013), 546–555.

[37] Moubarak, J., E. Filiol, and M. Chamoun, "Comparative analysis of blockchain technologies and TOR network: Two faces of the same reality?", *1st Cyber Security in Networking Conference*, (2017), 1–9.

[38] Wang, J., M. Li, Y. He, H. Li, K. Xiao, and C. Wang, "A Blockchain Based Privacy-Preserving Incentive Mechanism in Crowdsensing Applications", *IEEE Access 6*, 2018, pp. 17545–17556.

[39] Luu, L., V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A Secure Sharding Protocol For Open Blockchains",

ACM SIGSAC Conference on Computer and Communications Security, (2016), 17–30.
[40] Dehling, T., and A. Sunyaev, "Secure provision of patient-centered health information technology services in public networks—leveraging security and privacy features provided by the German nationwide health information technology infrastructure", *Electronic Markets 24*(2), 2014, pp. 89–99.
[41] Dennis, R., G. Owenson, and B. Aziz, "A Temporal Blockchain: A Formal Analysis", *2016 International Conference on Collaboration Technologies and Systems (CTS)*, (2016), 430–437.
[42] Anh, D.T.T., M. Zhang, B.C. Ooi, and G. Chen, "Untangling Blockchain: A Data Processing View of Blockchain Systems", *IEEE T KNOWL DATA EN 30*(7), 2018, pp. 1–20.
[43] Kaaniche, N., and M. Laurent, "A blockchain-based data usage auditing architecture with enhanced privacy and availability", *IEEE 16th International Symposium on Network Computing and Applications*, (2017), 1–5.
[44] Li, C., and L.J. Zhang, "A Blockchain Based New Secure Multi-Layer Network Model for Internet of Things", *IEEE International Congress on Internet of Things*, (2017), 33–41.
[45] Min, X., Q. Li, L. Liu, and L. Cui, "A Permissioned Blockchain Framework for Supporting Instant Transaction and Dynamic Block Size", *IEEE Trustcom/BigDataSE/ISPA*, (2016), 90–96.
[46] ISO, "Information processing systems — Open Systems Interconnection — Basic Reference Model — Part 2: Security Architecture", 1989.
[47] Dearnley, P.A., "An Investigation into Database Resilience", *The Computer Journal 19*(2), 1976, pp. 117–121.
[48] Porru, S., A. Pinna, M. Marchesi, and R. Tonelli, "Blockchain-oriented Software Engineering: Challenges and New Directions", *39th International Conference on Software Engineering Companion*, (2017), 169–171.
[49] Dai, M., S. Zhang, H. Wang, and S. Jin, "A Low Storage Room Requirement Framework for Distributed Ledger in Blockchain", *IEEE Access 6*, 2018, pp. 22970–22975.
[50] Bondi, A.B., "Characteristics of Scalability and their Impact on Performance", *2nd International Workshop on Software and Performance*, (2000), 195–203.
[51] Dickerson, T., P. Gazzillo, M. Herlihy, and E. Koskinen, "Adding Concurrency to Smart Contracts", *ACM Symposium on Principles of Distributed Computing*, (2017), 303–312.
[52] Lundqvist, T., A. de Blanche, and H.R.H. Andersson, "Thing-to-thing electricity micro payments using blockchain technology", *Global Internet of Things Summit*, (2017), 1–6.
[53] Pustišek, M., and A. Kos, "Approaches to Front-End IoT Application Development for the Ethereum Blockchain", *PROCEDIA COMPUT SCI 129*, 2018, pp. 410–419.
[54] Rimba, P., A.B. Tran, I. Weber, M. Staples, A. Ponomarev, and X. Xu, "Comparing Blockchain and Cloud Services for Business Process Execution", *IEEE International Conference on Software Architecture*, (2017), 257–260.
[55] Subramanian, H., "Decentralized Blockchain-based Electronic Marketplaces", *Communications of the ACM 61*(1), 2017, pp. 78–84.
[56] Mencias, A.N., D. Dillenberger, P. Novotny, et al., "An optimized blockchain solution for the IBM z14", *IBM J RES DEV*, 2018, pp. 1–11.
[57] Frey, D., M.X. Makkes, P.-L. Roman, F. Taïani, and S. Voulgaris, "Bringing Secure Bitcoin Transactions to Your Smartphone", *15th International Workshop on Adaptive and Reflective Middleware*, (2016), 1–6.
[58] Anjum, A., M. Sporny, and A. Sill, "Blockchain Standards for Compliance and Trust", *IEEE Cloud Computing 4*(4), 2017, pp. 84–90.
[59] Buccafurri, F., G. Lax, S. Nicolazzo, and A. Nocera, "Overcoming Limits of Blockchain for IoT Applications", *12th International Conference on Availability, Reliability and Security*, (2017), 1–6.
[60] van der Heijden, R.W., F. Engelmann, D. Mödinger, F. Schönig, and F. Kargl, "Blackchain: Scalability for Resource-constrained Accountable Vehicle-to-x Communication", *1st Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers*, (2017), 1–5.
[61] Eyal, I., and E.G. Sirer, "Majority is not Enough: Bitcoin Mining is Vulnerable", In N. Christin and R. Safavi-Naini, eds., *Financial Cryptography and Data Security*. 2014.
[62] Vukolić, M., "Rethinking Permissioned Blockchains", *ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, (2017), 3–7.
[63] ISO/IEC, "Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models", *ISO/IEC 25010:2011*, 2011.
[64] Luu, L., D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making Smart Contracts Smarter", *ACM SIGSAC Conference on Computer and Communications Security*, (2016), 254–269.
[65] Unterweger, A., F. Knirsch, C. Leixnering, and D. Engel, "Lessons Learned from Implementing a Privacy-Preserving Smart Contract in Ethereum", *9th IFIP International Conference on New Technologies, Mobility and Security*, (2018), 1–5.
[66] Stanciu, A., "Blockchain Based Distributed Control System for Edge Computing", *21st International Conference on Control Systems and Computer Science*, (2017), 667–671.
[67] Hofmann, F., S. Wurster, E. Ron, and M. Böhmecke-Schwafert, "The immutability concept of blockchains and benefits of early standardization", *ITU Kaleidoscope: Challenges for a Data-Driven Society*, (2017), 1–8.
[68] Lemieux, V.L., "A Typology of Blockchain Recordkeeping Solutions and some Reflections on their Implications for the Future of Archival Preservation", *IEEE International Conference on Big Data*, (2017), 2271–2278.
[69] Ziegeldorf, J.H., F. Grossmann, M. Henze, N. Inden, and K. Wehrle, "CoinParty: Secure Multi-Party Mixing of Bitcoins", *5th ACM Conference on Data and Application Security and Privacy*, (2015), 75–86.
[70] Zhao, X., Z. Chen, X. Chen, Y. Wang, and C. Tang, "The DAO attack paradoxes in propositional logic", *4th International Conference on Systems and Informatics*, (2017), 1743–1746.