Requirement-driven Taxonomy Development – A Classification of Blockchain Technologies for Securities Post-Trading

Benedikt Notheisen
Karlsruhe Institute of Technology
benedikt.notheisen@kit.edu

Sven Willrich
FZI Research Center
willrich@fzi.de

Maximilian Diez
Hasso Plattner Institute
m.diez@posteo.de

Christof Weinhardt
Karlsruhe Institute of Technology
christof.weinhardt@kit.edu

Abstract

In recent years, blockchain and distributed ledger technology (DLT) and its disruptive potential has been one of the most discussed topics in the field of information systems. Driven by the prospect of cost savings and efficiency gains, financial markets are at the core of these discussions. However, in the increasingly convoluted and constantly evolving market of technology providers and platforms, organizations struggle to find a solution that fulfills the specific requirements of their application scenario. To evaluate the suitability of different blockchain-based platforms for securities post-trading, we develop a new methodology to create a technology classification that takes the demands of a specific application context into account. The resulting requirement-based taxonomy sheds light on factors that impede the adoption of blockchain- and DLT-based post-trading, highlights future research challenges, and offers a valuable tool to induce communication between involved stakeholders.

1. Introduction

In recent years, the disruptive potential of distributed ledger (DLT) and blockchain technology [1, 2, 3] and its adoption to practical application contexts [4, 5] were one of the most discussed topics in the field of information systems (IS) [6]. To leverage the disruptive potential, many organizations intensify their blockchain and DLT activities by increasing their investments in external knowledge acquisition [7] as well as internal R&D efforts [8]. However, many organizations fail to find an appropriate technology for their use cases in the rapidly growing and increasingly complex blockchain and DLT market [2].

In the field of post trading, the multitude and diversity of requirements, the involvement of multiple stakeholders (e.g., exchanges, infrastructure providers, banks, professional and retail traders, governments and regulators), and the businesses' global scale complicate this task even further. As a result of this complexity, financial markets struggle to adopt DLT and blockchain technologies despite their immense potential for costs savings and efficiency gains [9].

RQ 1: To which extent can existing blockchain and DLT-solutions fulfill the requirements of securities post-trading?

However, finding the right technology for post-trading constitutes a challenging task that needs to incorporate the constantly evolving blockchain landscape and the functional requirements of a use case alike. While IS research offers a variety of taxonomies and frameworks, such as [10] or [6] that help to structure knowledge about blockchain and DLT, many organizations still fail to connect these abstract concepts to the specific requirements of their use case at hand. In addition, taxonomy development in IS has largely been ad hoc [11] and lacks a structured process to take requirements into account.

RQ 2: How can we create a taxonomy that incorporate the specific requirements of a use case in a structured way?

To resolve these research questions, we propose a new method that combines requirements and technological aspects to create a comparative technology assessment tool (RQ 2). We then apply this method to the use case of securities post-trading to assess and compare different enabling technologies (i.e. DLT platforms) and to identify critical requirements and technology dimensions (RQ 1). The resulting requirement-based taxonomy classifies Ethereum, Hyperledger Fabric, and Corda according to system access, the consensus mechanism, and the characteristics of the underlying database. Based on these dimensions, it evaluates each technology's features with respect to the legal, regulatory, technological, functional requirements of securities' post-trading. While we find that Corda and Hyperledger may be promising solutions, our taxonomy also highlights that no technology is able to meet all requirements yet. More specifically, we identify

URI: https://hdl.handle.net/10125/59899
ISBN: 978-0-9981331-2-6

compatibility issues, the transparency-confidentiality trade-off, and scalability as critical factors for future adoption, and thus starting points for future research.

Consistent with these results, our contribution is twofold: First, we analyze the use case of blockchain-based post-trading in financial markets and thereby shed light on the suitability of three popular DLT platforms (RQ 1). To do so, we create a new method for taxonomy development that takes the technological, socio-economic, and legal environment [6] of our use case into account (RQ 2). We furthermore hope that this method provides a valuable tool for researchers and practitioners to explore technology use cases beyond blockchain and DLT.

The remainder of this paper is structured as follows: Section 2 introduces the taxonomy concept and highlights a research gap. Section 3 establishes our new method. In section 4, we apply our new approach to the use case of blockchain- and DLT-based post-trading. Finally, section 5 discusses and concludes our results.

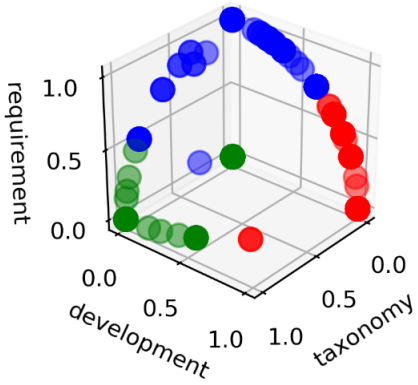
2. Foundations: Taxonomies

To provide a foundation for the methodology development in section 3, we aim to establish a general understanding of taxonomies and ontologies in this section.

A common feature of both concepts - taxonomies and ontologies - is their hierarchical structure. However, a taxonomy can be seen as a subset of ontologies [12] as it only shows an "is a" relationship, whereas an ontology covers various kinds of relationships from multiple perspectives. Moreover, an ontology can be a combination of multiple taxonomies. As a result, ontologies cover a wider range of knowledge, and thus facilitate the representation of different interconnected fields of knowledge in a systematic manner that allows the derivation of inferences between and within those fields. A taxonomy on the other hand, allows going into detail, moving from a high level of abstraction to more granular specializations. In consequence, taxonomies are used to classify objects among various dimensions within potentially complex domains. In both fields - management science and IS - taxonomies gained popularity in recent years and experience a higher priority among researchers [13]. This increased interest builds on their capability to reduce complexity and to identify similarities and differences among objects [14]. In the context of this study, we follow [13] and define a taxonomy T as a set of dimensions $D_i (i = 1, \ldots, n)$ where any dimension contains $k_i \mid k_i \geq 2$ mutually exclusive and collectively exhaustive characteristics $C_{ij} (j = 1, \ldots, k_i)$.

To identify related literature that deals with requirement-driven taxonomy development, we queried the most important scientific databases including Crossref, Elsevier (including Scopus), IEEE, Arxiv, and Springer. Within these queries, we searched for the occurrence of the keywords *requirement*, *development*, and *taxonomy*, sorted the query output by popularity in descending order, and limited the results to $10,000$ hits. In addition, we excluded all hits without an abstract available and a publication date before 1980. Based on the resulting output, we conduct a lemmatization and compute k-means clusters based on the relative importance of a word (*requirement, development, taxonomy*) in the abstract. Figure 1 shows the resulting keyword-based clusters in a 3-dimensional plot and highlights the scarcity of taxonomy development approaches that take the specific requirements of a use case explicitly into account. In addition, it highlights that the classification of objects in a domain of interest is a challenging task and guidance in taxonomy development process is rarely addressed in IS research. We aim to fill this research gap between requirement-driven and classification approaches with the method proposed in section 3.

Figure 1. Common appearance of the keywords "requirement", "development" and "taxonomy" within the identified literature



3. Methodology: Taxonomy Development

3.1. Methodological Approach

To guide the creation of our requirement-driven taxonomy development approach, we follow the design science paradigms of [15] and [16]. The resulting artifact is a method that intends to support researchers and practitioners in their endeavor to structure and organize domain-specific knowledge, while taking the socio-economic, legal, and technological environment

and the functional requirements of a specific application scenario into account. This way, we ensure that our artifact provides a "means to reach desired ends while [taking] the problem environment" into account [16]. This section describes our method as a "set of steps [...] used to perform a task" (i.e. the development of the taxonomy) [15].

3.2. Requirement-driven Taxonomy Development

To develop a classification approach that considers the technological characteristics as well as the requirements of a specific use case, we build on the literature identified in section 2. The resulting method is mainly based on [11] and [13]. In their paper, [11] conduct a comprehensive literature review and propose a structured approach that concentrates on the taxonomy's users' needs. However, their taxonomy development process also includes characteristics irrelevant for the use case at hand.

In contrast to [11] and [13], our method utilizes requirement documents to span an additional perspective and allows the user to alternate between technology and requirement iterations during the creation of the taxonomy. With this requirement-driven approach, we aim to streamline the usability of taxonomies as an analytic tool as dimensions that do not correspond to any requirement or vice versa are excluded.

Eventually, our method yields a taxonomy that consists of a set of requirements and a set of technology dimensions. Each dimension comprises a set of characteristics, while a meta-characteristic specifies the technology and its field of application. Note that we focus on the taxonomy development procedure, and thus do not provide a method to create the requirement documents. Instead, we assume that these requirement documents may originate for instance from experiences from previous projects, interviews with potential users, technology specifications, institutional publications, or requirement analyses. It is also important to keep in mind that the output of our approach depends on its input. Therefore, low quality input results in low quality output. This makes it highly advisable to apply this method in conjunction with requirements engineering approaches such as [17]. Furthermore, the resulting taxonomy is stakeholder-agnostic, and thus any constraint regarding stakeholders must either be incorporated before supplying the input to the method or as part of the ending conditions (detailed in the course of this section).

To initiate the development process, one needs to choose a meta-characteristic. The meta-characteristic is the most comprehensive characteristic that comprises all facets of the use case at hand and forms the basis for the more detailed iterative taxonomy development process. As a result, choosing an appropriate meta-characteristic is crucial in order to create a meaning- and purposeful taxonomy. To ensure an appropriate choice, we follow [11] and recommend to choose the meta-characteristic based on the taxonomy's purpose, its user group, and their expected use. In practice, this includes the technology (domain knowledge) as well as the functional requirements given by the specific application [6]. This bisection of the meta-characteristics ensures the applicability of our method beyond the specific use case of blockchain and DLT-based (domain knowledge) post-trading (requirements).

Similar to the meta-characteristic, the definition of ending conditions is crucial to ensure the return of a useful and valid taxonomy, as the iterative creation of requirements and dimensions is repeated until the return of algorithm 1 meets the specified quality criteria. The ending conditions depend on the meta-characteristic (i.e. the taxonomy's purpose, use, and users) and constitute a crucial driver of the final artifact's conciseness and comprehensiveness. A fundamental objective ending condition to ensure the outputs inherent quality is the mutually exclusive and collectively exhaustive nature of characteristics within all dimensions [11]. In addition, there are other objective and subjective ending conditions that reflect use case- and user-specific factors. [11] provide a list of important objective and an initial set of subjective ending conditions to consider in the taxonomy development process. Ending conditions may also incorporate aspects regarding individual or multiple stakeholders. To ensure a stakeholder-agnostic output, an ending condition may demand to cover requirements of all stakeholders for example.

In combination with the *requirement documents* and the *domain knowledge* (i.e. the meta-characteristic), the *ending conditions* form the core input for our method: The iterative taxonomy development process described in algorithm 1. The first part of each iteration focuses on the *requirement documents* and creates a list of *new requirements* based on one *requirement document* drawn from the stack of *requirement documents*. Based on this list, our method applies the empirical-to-conceptual approach in order to choose a relevant *requirement* from this list and add it to the *requirements* of the taxonomy. Second, the technology phase converts the *domain knowledge* into technology *dimensions* with either the empirical-to-conceptual or the conceptual-to-empirical approach. The inductive empirical-to-conceptual approach generates new

dimensions by using statistical techniques to classify a sample of objects based on common characteristics derived from the meta-characteristic [11, 14]. The deductive conceptual-to-empirical approach on the other hand, aims to conceptualize the dimensions based on the users' notion about the similarities between objects. The success of this approach depends heavily on the expertise, experience, and judgment of the user and may yield inappropriate dimensions. However, the characteristics of a dimension must logically follow from the meta-characteristic. If this is not the case, a dimension can be seen as inappropriate and should be discarded [11, 14]. It is important that the requirements or characteristics considered within a dimension are mutually exclusive and collectively exhaustive, as they would not add any information to the classifying nature of the taxonomy otherwise.

In each iteration, the requirement phase gets a new *requirement document* to create a list of *new requirements* and removes the corresponding *requirement document* from the stack of *requirement documents*. If there is a significant overlap between a *new requirement* and the *requirements* from previous iterations, their consolidation creates a hierarchical order that aggregates overarching concepts. If a *new requirement* affects the technology *dimensions* of our *mapping*, the *dimensions* in question will be revised and mapped to the *new requirement* if applicable. If not, the *requirements* are kept for future iterations, where a *new dimension* may affect them or they may be consolidated with a *new requirement*. Eventually, they will be discarded if none of the other cases applies. When the list is empty, no further requirement iterations are required.

An iteration's technology phase starts with identifying *new dimensions* based on the *domain knowledge*. Again, we establish a hierarchical order of *dimensions* by consolidating overlapping *dimensions*. Similar to the requirement phase, a *new dimension* must be relevant to be added to the list of *dimensions* for further considerations: If a *new dimension* is affected by any previously identified *requirements*, it is added to the *mapping* or kept for future iterations otherwise. Eventually, *dimensions* unaffected by requirements will be discarded.

At the end of each iteration, the user has to check whether the return of algorithm 1 fulfills the initially defined *ending conditions*. If all *ending conditions* are met, the development process terminates and yields a valid *mapping* that spans the taxonomy. Otherwise, another iteration is required. Within each new iteration, the current the *requirements*, *dimensions*, and *mapping* will be updated according to newly identified *requirement documents* and extensions of the *domain knowledge* [14]. Note that after adding new *requirements*, already existing technology *dimensions* and their characteristics have to be revisited and vice versa.

The resulting taxonomy is constructed as a *mapping* from the set of technology *dimensions* to the set of *requirements* that satisfies the validity demands outlined by [11]. Across all iterations, it is important to remember that the taxonomy should not have a descriptive but rather an explanatory nature [11]. Note that algorithm 1 returns not all technology *dimensions* considered but only those that are relevant with respect to at least one *requirement*. To ensure the overall relevance of included *dimensions* and *requirements*, each of them has to comply with the meta-characteristic.

In total, our method reduces the possibility to include arbitrary technology dimensions and characteristics, can be conducted in reasonable time, is straightforward to apply, and leads to a useful result. The resulting taxonomy complies with the formal demands outlined by [11, 13], is concise, robust, comprehensive, extendible, and explanatory [14]. In addition, it consists of one or more dimension and one or more requirement, which comprise mutually exclusive and collectively exhaustive characteristics. In contrast to previous approaches, such as [14] or [11], our method offers a multi-perspective (requirements/technology) approach that utilizes a "generate/test cycle" [16] to search for a meaningful two-sided classification system.

3.3. Interpretation of the Results

A requirement-driven taxonomy is intended to support its user in the assessment of a new technology by structuring information about its characteristics hierarchically and relating them to the specific requirements of a use case. This way, we aim to provide an intuition, which technological design decisions to consider to meet a given requirement and whether the resulting characteristics facilitate or impede compliance. Empty cells indicate the irrelevance of a technology dimension for a requirement. Positive compliance occurs when a specific technology (feature) fulfills a requirement, while negative compliance highlights discrepancies. However, the taxonomy needs to be interpreted carefully, as algorithm 1 does not exclude infeasible system configurations and the quality of the result depends on the quality of the inputs. Especially in young technology fields, such as blockchain technology and DLT, advances in the domain knowledge may render a taxonomy outdated after a short period of time. More generally speaking, whenever we

Algorithm 1: Requirement-driven taxonomy development

Data: sets of {requirement documents, domain knowledge, ending conditions}
Result: taxonomy, map from dimension to set of requirements

$dimensions \leftarrow \emptyset$
$requirements \leftarrow \emptyset$
$mapping \leftarrow \emptyset$
begin
  while *ending conditions met* **do**
    if *requirement documents* $\neq \emptyset$ **then**
      requirement document $\leftarrow$ get and remove *requirement document* from *requirement documents*
      new requirements $\leftarrow$ analyse *requirement document*
      while *new requirements* $\neq \emptyset$ **do**
        new requirement $\leftarrow$ get and remove *new requirement* from *new requirements*
        if *new requirement matches meta-characteristic* **then**
          add *new requirement* to *requirements*
          consolidate *requirements*
          foreach *dimension in dimensions* **do**
            if *dimension affects new requirement* **then**
              add $(dimension, new\ requirement)$ to *mapping*
            end
          end
        end
      end
    else
      new dimension $\leftarrow$ identify technological dimension using *domain knowledge*
      add *new dimension* to *dimensions*
      consolidate *dimensions*
      foreach *requirement in requirements* **do**
        if *new dimension affects requirement* **then**
          add $(new\ dimension, requirement)$ to *mapping*
        end
      end
    end
  end
  return $\{d \in dimensions \mid \exists(d, \_) \in mapping\}, mapping$
end

use an incorrect, outdated, or incomplete data as input, the resulting taxonomy might fail to fulfill its purpose. The same issues arise, when new requirements emerge. However, algorithm 1 also allows to update a taxonomy via additional iterations, if new information on either side becomes available. In addition, the result is restricted to the specific requirements and technology domain of the use case under question. As a result, for each application scenario (i.e. the combination of requirements and a technology domain), a new taxonomy needs to be developed. Lastly, while creating a foundation to initiate action and facilitate communication among stakeholders, a requirement-based taxonomy only provides a structured overview over technology characteristics that neither guides nor recommends any direct action. To derive direct implications however, one must consider the priorities and perspectives of individual stakeholders, which goes beyond the scope of our approach.

4. The Case of Post-Trading Services in Financial Markets

To evaluate the capability of different blockchain and DLT platforms to enable post-trading services, identify critical requirements, and highlight impeding factors for adoption, we apply the method proposed in section 3 to the use case of post-trading services

in financial markets. More specifically, we create a requirement-driven taxonomy of three popular smart contract platforms with a publicly available knowledge base: Ethereum [18], Corda [19], and Hyperledger Fabric [20, 21]. We consider the possibility to use smart contracts as an important prerequisite to implement the functional scope and complexity of modern post-trading infrastructures. In addition, we focus on more general and popular platforms to ensure the quality of our inputs and to create a broad domain knowledge that enables us to differentiate between the technologies appropriately.

4.1. Post-Trade Requirements

Post-trading services ensure that a trade is completed according to the agreement concluded between buyer and seller. This includes the assessment of a trade's details, the approval of the transaction, changing the ownership records, and eventually the exchange of securities for cash [22]. In total, the core functions of post-trading services are the clearing and settlement of transactions, the custody and safekeeping of securities, and the provision of notary services [23]. Globally, this generates costs of more than $ 10 BN per year [24]. Driven by the promise of efficiency gains, market participants strive to leverage the blockchain's potential to replace current centralized infrastructures that create these costs [9]. However, the post-trading industry faces a variety of legal, regulatory, technological, functional requirements, and thus comparing different solutions with respect to their suitability for post-trading is a challenging task.

To elicit these requirements, we gather multiple requirement documents from institutions [25, 9, 26, 27, 28, 29, 30, 31], academics [32, 24, 33, 34, 35], and market participants [23]. Especially, the access to technological documents from Boerse Stuttgart - the second largest stock exchange in Germany - helped us to take the specific infrastructure requirements into account. Table 1 summarizes the extracted post-trading requirements and provides a simplified overview. Note that the requirement documents and not this list form the initial input for the requirement-driven taxonomy development procedure introduced in section 3.

Table 1. Summary of post-trading requirements

Requirement	Description
Governance	Ability to exert control over the system (e.g. in crisis).
Mutability	Retrospective changes of database updates to correct mistakes.
Scalability	Capability to process a specific number of transactions per time interval.
Data controls	Confidentiality on the individual and transparency on the regulatory level.
Reliability	Proper functioning and continuous availability.
Security	Correctness of the stored data and robustness towards corruption.
Finality	Delay after which transactions can be considered as final.
Identification	Provision of unique and certified identifiers for each entity.
Compatibility	Standardized interaction with legacy systems, ancillary services, and other infrastructures.
Effectiveness	Implementation of all relevant process steps and features.

4.2. Domain Knowledge

4.2.1. Blockchain & DLT Basics The blockchain was first introduced as the underlying technology of Bitcoin and provides a means to manage and update a shared transactional database within a network of interacting parties that utilize a consensus algorithm and cryptographic security measures to replace a central authority [6]. In consequence, the fundamental innovation that comes with blockchain technology is the decentralization of the consensus authority and the resulting revolution of the role of trust between peers and between peers and blockchain-based platforms [36].

To achieve such a trust-free environment the consensus algorithm moderates database updates, thereby allowing potentially conflicting parties to achieve an agreement on the validity of new information [37]. More specifically, these changes constitute atomic updates (i.e. they are either fully applied or not applied at all) that result from transactions between peers [38]. The resulting append-only database is an ordered list of all updates, which enables all network participants to determine the system's current state and fully reproduce historic transactions [38]. The log itself is immutable, and thus removing or tampering with agreed and stored data becomes infeasible [39]. From a network perspective the ledger is stored and replicated across every participant in the network providing transparency over historical transactions, while individual participants can leave and join the network arbitrarily [1, 37]. From a formal perspective, the term blockchain is usually associated with the underlying data structure - a chain of data blocks linked by cryptographic hashes - and often

refers to a public and pseudonymous setup. The term DLT in contrast comprises a broader, more general spectrum. DLT includes any technology that allows the trust-free management of data in a decentralized way. Within either system - DLT and blockchain - smart contracts offer a way to implement software logic, which allows complex interactions to happen without a central governing authority [1].

4.2.2. Design Decisions On the global DLT and blockchain market, many different blockchain systems are available [5]. While each of them share the same building blocks, their the individual designs reflect the requirements imposed by a specific application context [6]. This section builds on [40, 41, 38, 42] and summarizes fundamental DLT and blockchain design decisions, thereby spanning the domain knowledge for the upcoming taxonomy development.

The system's *Access Scope* determines whether access to the system is *permissionless* or *permissioned*. The degree of access depends on an individual's rights to initiate and view updates, the ability to verify newly proposed updates, as well as the possibility to freely join a network. In a public setup, new participants can join without permission, while private networks may lay down specific requirements or restrict access to a limited set of entities.

The *consensus algorithm* defines the way participants exert control over data and agree on updates. As a result, its robustness towards the malicious behavior of individuals determines safety, security, and performance [43]. In permissionless setups, algorithms such as Proof-of-Work or Proof-of-Stake randomize the costly privilege to append new data. At the same time transaction fees and rewards for appending data facilitate the competitiveness of the consensus process. This concept, aims to prevent the corruption of data and to minimize the risk of compromising attacks [44]. In permissioned systems on the other hand, byzantine fault tolerant (BFT) algorithms, such as Proof-of-Authority or Practical and Federate BFT, offer cheaper alternatives to vote on the correctness of new data [45].

Within the consensus process, the verification of new transactions comprises two steps, *validation* and *ordering* [30]. The first step verifies each transaction's formal correctness and checks the signature and ownership of the sender. If the new transactions are formally correct, the second step establishes a sequence of transactions for the database update. Depending on the *Consensus Type*, the validation of transactions can be performed either separately by each node or in conjunction with their ordering on a network level. Separating these two steps can increase performance, as each participant only validates the transactions he or she is involved in, while centralized *notaries* are in charge of ordering and certify past transactions' correctness [19].

After consensus on the validity and order of transactions is achieved, the actual database update is conducted. This update can follow two design paradigms that affect performance and practical usability of the system. The first and more common *Update Type* relies on the *UTXO model* [42]. In this model, an update incorporates a set of unspent transaction outputs (UTXO) of the sender that generates a set of new UTXOs for the recipient of a transaction. The UTXO model is stateless in the sense that there is no need to keep anything except UTXOs to reproduce the current ledger state. The second type is based on the *account model* [42]. In the account model, any update is either a transaction between participants' accounts or a request to a smart contract, which changes its state. In contrast to the UTXO model, the account model is stateful and data stored in a smart contract must be kept by all participants until it is overwritten. Note, that neither model is technically more powerful and - although impractical - a conversion between them is easy to achieve [18].

Independent of their type, database updates are stored in one of three *Data Structures*. The first and most common structure is a *single chain*, where updates are grouped into blocks and then linked to secure their order. Alternatively, a blockchain or DLT system can also comprise *multiple chains* that are verified and updated independently. However, an update that concerns more than one chain can lead to problems. The third and fundamentally different structure is a *directed acyclic graph* (DAG) [46]. In contrast to a single or multi-chain structure, a DAG block can be linked to more than one preceding block. This way, non-conflicting updates can be verified at the same time. However, while improving scalability, this structure may impose security threats.

The *confidentiality* of the stored data is a critical point for many applications and is in sharp contrast to the transparency paradigm inherent in most blockchain and DLT systems. One way to achieve *full* confidentiality is to refrain from storing sensitive data *on-chain*. In addition, *Partial* confidentiality can be reached by encrypting or hashing data or limiting access to a specific set or type of nodes. However, increasing confidentiality comes at the cost of transparency, the robustness of consensus, and performance (e.g. in the context of zero knowledge proofs [47]).

4.3. A Taxonomy of Blockchain-based Post-Trade Solutions

With the rise of blockchain technology and DLT, several related taxonomies have been proposed. [41] for instance categorize DLTs based on technological features. However, their resulting taxonomy does neither fulfill the conditions outlined by [11] nor focus on the specific use case of post-trading. Similarly, [10] centers his taxonomy on decentralized consensus mechanisms while lacking a specific application focus. In consequence, we propose a taxonomy of blockchain and DLT with respect to post-trading in financial markets.

4.3.1. Taxonomy Development The meta-characteristic guiding the development process is the feasibility of blockchain technology and DLT for securities post-trading. This comprises the requirements represented by the requirement documents elicited in section 4.1 as well as the blockchain features introduced in section 4.2 (i.e. the domain knowledge). With the resulting taxonomy, we aim to make the considered technologies comparable with respect to their use in the field of securities post-trading (purpose). This helps all users involved in the post-trading process (e.g., exchanges, brokers, and other service providers) to identify the best suited basis for a blockchain- or DLT-based post-trading (use).

To define the ending conditions for the development procedure, we follow [11]. The resulting objective ending conditions require that all dimensions and requirements have to be examined, while every requirement, dimension, and characteristic is unique, mutually exclusive, collectively exhaustive, and possible. In addition, to ensure the explanatory power beyond a pure requirements and technology perspective, we demand that every technology dimension is associated with at least one requirement. Subjectively, we require the number of requirements, dimensions, and characteristics to allow for a understandable but precise distinction, thereby providing a sufficient level of differentiation. This ensures that the resulting taxonomy is concise and robust. Furthermore, all relevant requirements and dimensions should be covered in order to enable a comprehensive classification of all considered solutions.

Within each iteration phase, we follow algorithm 1 to gradually extract all requirements from the requirement documents highlighted in section 4.1 and consolidate them in each iteration, if necessary. Similarly, we extract a new dimension from the domain knowledge summarized in section 4.2 in each iteration. This way, we refine, consolidate, and discard possible blockchain design decisions with respect to their relevance in the context of securities post-trading. Within the reciprocal creation of the taxonomy's mapping, we integrate mutability in governance as well as finality in security on the requirement side, while we discard compatibility. We do so, because to this date none of the considered technologies explicitly implements compatibility features to establish a standardized connection to other (legacy) systems. In consequence, dropping the compatibility requirement does not harm comprehensiveness, as it does not provide any explanatory power yet. With an analogous reasoning, we furthermore drop the parallel characteristic within data structure, and full within confidentiality. To account for the blockchain's trade-off between confidentiality and transparency, the data controls requirement is furthermore split into these two requirements. Note that when the considered technologies implement new features an update of the taxonomy may be required.

4.3.2. The Taxonomy After 15 iterations, algorithm 1 yields a valid taxonomy that classifies Ethereum, Corda, and Hyperledger Fabric according to their access scope, the consensus algorithm (Cons Algo) and type (Cons Type), data structure, update type, and confidentiality. The taxonomy is depicted in table 2 and complies with all objective and subjective ending conditions. Note that while the considered technologies also allow for other specifications, we focused on the baseline configuration introduced in the related technical documents. Due to the iterative design of the method, the taxonomy can be enhanced by adding new requirement documents, extending the domain knowledge, or updating the meta-characteristic if more data, new requirements, new technologies, or new features become available.

Table 2 depicts the resulting requirement-based taxonomy of blockchain- and DLT-based post-trading solutions. While the permissionless Ethereum protocol struggles with scalability, privacy, and identity issues, the permission-based specification of the Hyperledger Fabric or Corda offers a more promising foundation for potential implementations. However, the absent compatibility with legacy systems, the trade-off between transparency and confidentiality, as well as the finality of transactions still constitute limitations that impede adoption. In total, our taxonomy shows that none of the considered technologies is already able to fulfill all requirements demanded by post-trading in modern

Table 2. Taxonomy of Blockchain-based Post-trading Solutions: E stands for Ethereum, H for Hyperledger Fabric, and C for Corda. Empty cells indicate the irrelevance of a technology dimension and ✓ (×) indicates positive (negative) compliance with a requirement.

		Technology												
	Dimension	Access Scope		Cons Algo		Cons Type		Data Structure		Update Type			Confidentiality	
	Characteristic	Pmls	Pmd	PoX	BFT	Uni	Iso	Single	Multi	UTXO	Acc	Hyb	None	Part
Requirement	Governance	$E_{\times}$	$H_{\checkmark}C_{\checkmark}$	$E_{\times}$	$H_{\checkmark}, C_{\checkmark}$									
	Scalability	$E_{\times}$	$H_{\checkmark}C_{\checkmark}$	$E_{\times}$	$H_{\checkmark}C_{\checkmark}$	$E_{\times}H_{\times}$	$C_{\checkmark}$	$E_{\times}$	$H_{\checkmark}C_{\checkmark}$	$C_{\checkmark}$	$E_{\times}$	$H_{\times}$	$E_{\checkmark}$	$H_{\checkmark}C_{\checkmark}$
	Confidentiality	$E_{\times}$	$H_{\checkmark}C_{\checkmark}$			$E_{\times}H_{\times}$	$C_{\checkmark}$	$E_{\times}$	$H_{\checkmark}C_{\checkmark}$	$C_{\checkmark}$	$E_{\times}$	$H_{\checkmark}$	$E_{\times}$	$H_{\checkmark}C_{\checkmark}$
	Transparency	$E_{\checkmark}$	$H_{\times}C_{\times}$			$E_{\checkmark}H_{\checkmark}$	$C_{\times}$	$E_{\checkmark}$	$H_{\times}C_{\times}$	$C_{\times}$	$E_{\checkmark}$	$H_{\times}$	$E_{\checkmark}$	$H_{\times}C_{\times}$
	Reliability			$E_{\checkmark}$	$H_{\times}C_{\times}$									
	Security	$E_{\times}$	$H_{\checkmark}C_{\checkmark}$	E	$H_{\checkmark}C_{\checkmark}$									
	Identity	$E_{\times}$	$H_{\checkmark}C_{\checkmark}$											
	Effectiveness					$E_{\checkmark}H_{\checkmark}$	$C_{\times}$			$C_{\times}$	$E_{\checkmark}$	$H_{\checkmark}$		
	Total	E	HC	E	HC	EH	C	E	HC	C	E	H	E	HC

financial markets. Nonetheless, it allows us to direct the focus towards Corda and Hyperledger as promising solutions and highlights compatibility, confidentiality, and scalability as starting points for future R&D activities.

5. Conclusion

In total, we investigate the suitability of existing blockchain and DLT platforms to conduct post-trading services in modern financial markets. To do so, we develop a new methodology that allows its user to integrate the technology characteristics and demands of a specific application scenario to create a use case-specific classification of potential solutions (RQ 2). We then apply this approach to the use case of blockchain and DLT-based post-trading in securities markets. The resulting taxonomy highlights that the considered blockchain technologies and DLTs still struggle to meet all requirements demanded for post-trading services. Namely, the issues that impede adoption are the absent compatibility with existing systems, the trade-off between transparency and confidentiality, and scalability concerns (RQ 1). By the means of our taxonomy, we hope to offer a tool to create a baseline to prioritize R&D efforts and to facilitate communication among post-trading stakeholders.

However, it also is important to note that the our approach does not exclude infeasible technology configurations and the quality of the output depends on the quality of the input. In addition, we do not provide any direct recommendations but rather aim to facilitate the transparency over the connection between requirements and technological features to create a foundation for further analyses. In consequence, future methodical research may include the development of measures to control for input quality and a way to exclude infeasible configurations. Eventually, applying our approach to other cases would strengthen its validity, usefulness, and efficiency [16] beyond the formal quality criteria of [11, 13] and highlight possibilities for further improvements.

References

[1] L. W. Cong and Z. He, "Blockchain disruption and smart contracts," Working Paper 24399, National Bureau of Economic Research (NBER), March 2018.

[2] M. Friedlmaier, A. Tumasjan, and I. M. Welpe, "Disrupting industries with blockchain: The industry, venture capital funding, and regional distribution of blockchain ventures," in *Proceedings of the 51st Hawaii International Conference on System Sciences (HICSS)*, 2018.

[3] R. Beck and C. Müller-Bloch, "Blockchain as radical innovation: a framework for engaging with distributed ledgers as incumbent organization," in *Proceedings of the 50st Hawaii International Conference on System Sciences (HICSS)*, 2017.

[4] F. Holotiuk, F. Pisani, and J. Moormann, "Unveiling the key challenges to achieve the breakthrough of blockchain: Insights from the payments industry," in *Proceedings of the 51st Hawaii International Conference on System Sciences (HICSS)*, 2018.

[5] G. Salviotti, L. M. De Rossi, and N. Abbatemarco, "A structured framework to assess the business application landscape of blockchain technologies," in *Proceedings of the 51st Hawaii International Conference on System Sciences (HICSS)*, 2018.

[6] B. Notheisen, F. Hawlitschek, and C. Weinhardt, "Breaking down the blockchain hype – towards a blockchain market engineering approach," in *Proceedings of the 25th European Conference on Information Systems (ECIS)*, 2017.

[7] CB Insights, "Blockchain investment trends in review," 2017.

[8] A. Lannquist, "Blockchain in enterprise: How companies are using blockchain today," *Blockchain at Berkeley*, 2018.

[9] A. Pinna and W. Ruttenberg, "Distributed ledger technologies in securities post-trading," Occasional Paper Series 172, European Central Bank, 2016.

[10] F. Glaser and L. Bezzenberger, "Beyond cryptocurrencies-a taxonomy of decentralized consensus systems," in *Proceedings of the 23rd European Conference on Information Systems (ECIS)*, 2015.

[11] R. Nickerson, U. Varshney, and J. Muntermann, "A method for taxonomy development and its application in information systems," *European Journal of Information Systems*, vol. 22, no. 3, pp. 336–359, 2013.

[12] N. Guarino, D. Oberle, and S. Staab, "What Is an Ontology?," in *Handbook on Ontologies*, pp. 1–17, 2009.

[13] R. Nickerson, J. Muntermann, U. Varshney, and H. Isaac, "Taxonomy development in information systems: Developing a taxonomy of mobile applications," in *Proceedings of the 17th European Conference on Information Systems (ECIS)*, 2009.

[14] K. D. Bailey, *Typologies and taxonomies: an introduction to classification techniques*, vol. 102. Sage, 1994.

[15] S. T. March and G. F. Smith, "Design and natural science research on information technology," *Decision Support Systems*, vol. 15, no. 4, pp. 251 – 266, 1995.

[16] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Q.*, vol. 28, pp. 75–105, Mar. 2004.

[17] K. Pohl, *Requirements engineering: fundamentals, principles, and techniques*. Springer Publishing Company, Incorporated, 2010.

[18] V. Buterin et al., "A next-generation smart contract and decentralized application platform," *Whitepaper*, 2018.

[19] M. Hearn, "Corda: A distributed ledger," *Whitepaper*, 2016.

[20] Hyperledger, "Hyperledger architecture, volume 1: Introduction to hyperledger business blockchain design philosophy and consensus," *Whitepaper*, 2017.

[21] Hyperledger, "Hyperledger architecture, volume 2: Smart contracts," *Whitepaper*, 2018.

[22] D. Loader, "Chapter 1 - the structure of clearing and settlement," in *Clearing, Settlement and Custody* (D. Loader, ed.), pp. 1 – 16, San Diego: Butterworth-Heinemann, second ed., 2014.

[23] S. Mai, "The european post-trade market-an introduction," *Deutsche Börse Whitepaper*, 2005.

[24] A. Milne, "The industrial organization of post-trade clearing and settlement," *Journal of Banking & Finance*, vol. 31, no. 10, pp. 2945–2961, 2007.

[25] F. Braeckevelt, "Clearing, settlement and depository issues," BIS Papers 30, 2006.

[26] M. Mainelli and A. Milne, "The impact and potential of blockchain on securities transaction lifecycle," Working Paper 2015-007, SWIFT Institute, 2016.

[27] D. C. Mills, K. Wang, B. Malone, A. Ravi, J. C. Marquardt, A. I. Badev, T. Brezinski, L. Fahy, K. Liao, V. Kargenian, *et al.*, "Distributed ledger technology in payments, clearing, and settlement," Working Paper 2016-095, FED, 2016.

[28] J. Van de Velde, A. Scott, K. Sartorius, I. Dalton, B. Shepherd, C. Allchin, M. Dougherty, P. Ryan, and E. Rennick, "Blockchain in capital markets," whitepaper, Euroclear Group, Oliver Wyman, 2016.

[29] Diverse, "Distributed ledger technology in payment, clearing and settlement," committee on payments and market infrastructures, BIS, 2017.

[30] E. Benos, R. Garratt, and P. Gurrola-Perez, "The economics of distributed ledger technology for securities settlement," Working Paper 670, Bank of England, 2017.

[31] F. "Le Borne", D. Treat, F. Dimidschstein, and C. Brodersen, "Swift on distributed ledger technologies," position paper, SWIFT, 2018.

[32] B. S. Bernanke, "Clearing and settlement during the crash," *The Review of Financial Studies*, vol. 3, no. 1, pp. 133–151, 1990.

[33] R. McGill and N. Patel, *Clearing & Settlement*, pp. 43–58. London: Palgrave Macmillan UK, 2008.

[34] P. V. Cayseele and C. Wuyts, "Cost efficiency in the european securities settlement and depository industry," *Journal of Banking Finance*, vol. 31, no. 10, pp. 3058 – 3079, 2007.

[35] M. Geranio, "Fintech in the exchange industry: Potential for disruption," *Masaryk UJL & Tech.*, vol. 11, 2017.

[36] F. Hawlitschek, B. Notheisen, and T. Teubner, "The limits of trust-free systems: A literature review on blockchain technology and trust in the sharing economy," *Electronic Commerce Research and Applications*, vol. 29, pp. 50 – 63, 2018.

[37] B. Notheisen, J. B. Cholewa, and A. P. Shanmugam, "Trading real-world assets on blockchain," *Business & Information Systems Engineering*, vol. 59, pp. 425–440, Dec 2017.

[38] K. Wüst and A. Gervais, "Do you need a blockchain?," *IACR Cryptology ePrint Archive*, p. 375, 2017.

[39] V. Gramoli, "From blockchain consensus back to byzantine consensus," *Future Generation Computer Systems*, 2017.

[40] X. Xu, C. Pautasso, L. Zhu, V. Gramoli, A. Ponomarev, A. B. Tran, and S. Chen, "The blockchain as a software connector," in *Software Architecture (WICSA), 2016 13th Working IEEE/IFIP Conference on*, pp. 182–191, IEEE, 2016.

[41] X. Xu, I. Weber, M. Staples, L. Zhu, J. Bosch, L. Bass, C. Pautasso, and P. Rimba, "A taxonomy of blockchain-based systems for architecture design," in *Software Architecture (ICSA), 2017 IEEE International Conference on*, pp. 243–252, IEEE, 2017.

[42] E. Hamza, A. Outchakoucht, and J. P. Leroy, "A blockchain-based access control for big data," *International Journal of Computer Networks and Communications Security*, vol. 5, no. 7, pp. 137–147, 2017.

[43] C. Cachin and M. Vukolić, "Blockchains consensus protocols in the wild," *arXiv:1707.01873*, 2017.

[44] A. Chepurnoy, T. Duong, L. Fan, and H.-S. Zhou, "Twinscoin: A cryptocurrency via proof-of-work and proof-of-stake.," *IACR Cryptology ePrint Archive*, vol. 2017, p. 232, 2017.

[45] S. D. Angelis, L. Aniello, R. Baldoni, F. Lombardi, A. Margheri, and V. Sassone, "Pbft vs proof-of-authority: applying the cap theorem to permissioned blockchain," in *Italian Conference on Cyber Security*, 2018.

[46] Y. Lewenberg, Y. Sompolinsky, and A. Zohar, "Inclusive block chain protocols," in *International Conference on Financial Cryptography and Data Security*, pp. 528–547, 2015.

[47] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *Security and Privacy (SP), 2014 IEEE Symposium on*, pp. 459–474, IEEE, 2014.