Christian Sillaber, Bernhard Waltl

Life Cycle of Smart Contracts in Blockchain Ecosystems

This paper discusses the life cycle of decentralized smart contracts, i.e. digital and executable representations of rights and obligations in a multi-party environment. The life cycle relies on blockchain technology, i.e. a distributed digital ledger, to ensure proper implementation and integrity of the smart contracts. The life cycle consists of four subsequent phases: Creation, freezing, execution, and finalization. For each phase actors and technological services are identified and explained in detail. With the life cycle at hand, risks and limitations of smart contracts and the underlying blockchain technology are briefly discussed.

1 Introduction

Technological improvements and innovations foster the digital transformation of modern societies. Many different notions of this transformation, e.g., digitization, on every level of society can be observed. This is also true in the legal domain, which is still at a very early stage of digitization. Advances in computer science and information systems research have now created technologies that are supposed to have the potential of reshaping wide areas of the legal system as it is established nowadays. Especially contractual law seems to be highly suitable for digitization. The idea of having contracts or contractual rights and obligations in an executable representation, so-called *smart contracts*, is not new but in the light of today's innovations more attractive than ever.

In this paper we aim at sketching a holistic life cycle of decentralized smart contracts made possible by recent innovations based on blockchain technology. We briefly introduce the technological ecosystem in section 2 which serves as a baseline for the definition and the illustration of the life cycle of smart contracts in section 3. We provide a critical reflection on the main drawbacks and limitations of smart contracts in section 4 and conclude with a discussion and outlook in section 5.

Dr. Christian Sillaber

ist wissenschaftlicher Mitarbeiter am Institut für Informatik an der Leopold Franzens Universität Innsbruck.

E-Mail: Christian.sillaber@uibk.ac.at

Bernhard Waltl

ist wissenschaftlicher Mitarbeiter an der Fakultät für Informatik der Technischen Universität München.

E-Mail: b.waltl@tum.de

2 Technological ecosystems for decentralized smart contracts

This chapter briefly introduces two technologies serving as a baseline for decentralized smart contracts: distributed ledgers (see section 2.1), and crypto-currencies (see section 2.2).

2.1 Distributed digital ledger and blockchain technology

In order to create a digital ledger, the contained data, i.e. transactions, need to be stored safely. This means that the total order and content of the occurring transactions need to be maintained. Within blockchains, individual transactions are grouped into blocks, which are consecutively persisted [7]. Each block, except the first (genesis block), contains a pointer to the previously validated block. Using a mathematical procedure, i.e. hashing, the chain of blocks are entangled such that non-tail blocks cannot be changed unnoticed. This, the integrity of the chain is guaranteed. In order to ensure the integrity it is necessary to recalculate the hashes of the individual blocks, which can be done at will by everyone participating at the peer-to-peer network.

In order to add a new block, i.e., a set of transactions, every miner participates in a race in which a certain task (see [9], chapter 6) needs to be done. The performing of such a task is called mining. The first miner having a correct solution to the task can add a new block to the chain and the remaining miners agree on the transactions. The difficulty of the problem adapts to the *mining power* that is available throughout the distributed ledger to compensate for improvements in computing performance. The miner which solved the task first is usually rewarded, e.g. by a mining coin, to ensure the attractiveness of mining, which is required to ensure the integrity of the persisted transactions. The evolution of the chain is up to a decentralized peer-to-peer network that adds new content and agrees on correct transactions. Still some issues regarding the propagation time of a transaction throughout the network, e.g., double-spending of coins, and the approval of a block by the network, e.g., accepting the transaction, exist. More details on the design and implementation are provided in [9].

2.2 Unregulated decentralized digital crypto-currencies

This section briefly introduces crypto-currencies that heavily rely on the distributed ledger technology, e.g., Bitcoin. According to the European Central Bank, virtual money generated and traded in digital platforms, such as Bitcoin, can be defined as an "unregulated, decentralized, digital crypto-currency" [1]. There have already been attempts to create a digital currency in the 1990s. However, those attempts still required a bank (custodian of ledger) holding the accounts for the ownership of the money. Today, blockchains offer a technical solution to distribute this ledger, i.e. the transaction log, across a peer-to-peer network while preserving integrity of the transaction log [9].

This innovation enables unregulated crypto-currency markets. As of January 2017 over 600 different cryptocurrencies are known which are traded on over 2400 markets.[1] Although there are plenty of different crypto currencies available, an analysis of the market capitalization shows that Bitcoin accounts for more than 87% of the overall crypto-currency market, followed by Ethereum (4.65%) and Ripple (1.7%). On the one hand this shows that the market is dominated by one crypto-currency. On the other hand, it can be seen that new crypto-currencies are created almost daily, which are used in highly specialized markets or for niche assets.

3 Smart Contracts

3.1 Definition

With the necessary components provided by the blockchain (e.g. distributed secure ledger, consensus protocol), complex digital asset transactions as well as decentralized autonomous organizations can be created [2, 4]. The term *smart contracts* was introduced by Nick Szabo in [8], where he first outlined how the computer-based execution of contracts between two parties could be secured without requiring any third party:

> *"A set of promises, including protocols within which the parties perform on the other promises. The protocols are usually implemented with programs on a computer network, or in other forms of digital electronics, thus these contracts are 'smarter' than their paper-based ancestors.*"

In [3] a first rough definition of decentralized smart contracts is provided as programs that are executed by all miners allowing parties that not necessarily know and trust each other to securely transact with each other. The correct execution of these programs is enforced by a consensus protocol [5].

A smart contract, therefore, consists of three distinct components: the contractual arrangements between the parties, the governance of preconditions necessary for the contractual obligations to take place, and the actual execution of the contract. [4]

- **Contractual arrangements between the parties**: The contractual obligations of the involved parties are negotiated and transformed into executable program code. The parties are identified through their blockchain accounts (wallets), and transactions denote fulfilled obligations between them. The executable program code ties them together through logical connections and the evaluation of conditions. The code is then implemented and stored in the distributed blockchain.
- **Governance of preconditions**: All participating nodes including miners are capable of executing smart contracts. They evaluate whether the preconditions defined in the smart contract are met or not.
- **Execution of the contract**: If the preconditions have been met, the contract is executed and the transactions are performed by the participating nodes. The correct execution is ensured through the consensus protocol. Therefore, smart contracts are self-enforcing, meaning that digital assets are allocated autonomously according to the predefined contractual terms.

3.2 Life cycle of decentralized smart contracts

The life cycle of a smart contract typically consists of four broad phases: *creation of the smart contract*, *freezing of the smart contract*, *execution of the smart contract* and *finalization of the smart contract*.

- **Create**: The creation phase can be divided into an iterative contract negotiation and an implementation phase. First, the parties have to agree on the broad content and objectives of the contract. This can be done online or offline and is similar to classic contract negotiations. All participating parties must have a wallet on the underlying ledger platform. Its identifier is in most cases pseudonymous [9], and it is used for the identification of the parties as well as of the transfer of funds.

After agreeing on the objectives and content of the contract, the agreement has to be turned into code. The codification of the contract is limited by the expressiveness of the underlying smart contract coding language [6]. To validate a smart contract's execution behavior and content, most smart contract environments provide the infrastructure to create, maintain and test the contract. As can be seen in classic programming languages, the transformation of requirements into code requires several iterations between the stakeholders and programmer(s). Smart contracts will be no different and will probably require many iterations between the negotiation and implementation phase.

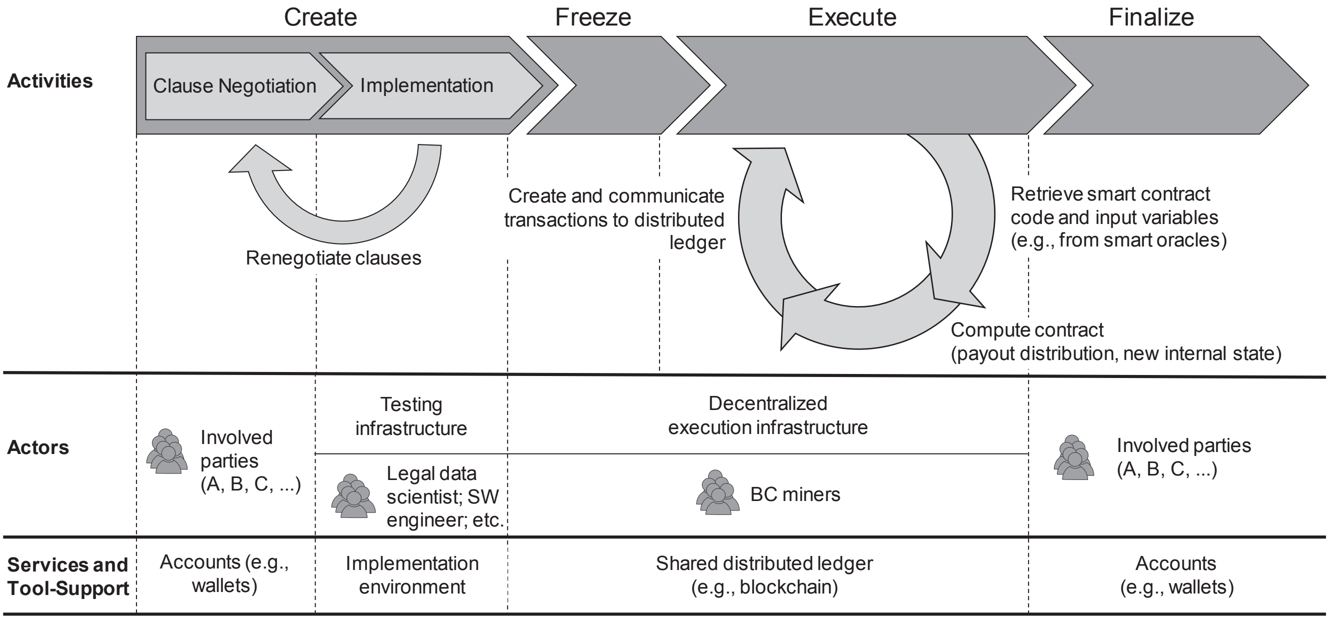
After the parties agree to the codified version of the contract, it is submitted to the distributed ledger during the publication phase. During this phase, nodes participating in the distributed ledger receive the contract as part of a transaction block and once the block has been confirmed by a majority of nodes, the contract is ready for execution.

Because decentralized smart contracts cannot be modified after being accepted by the blockchain, changes in the smart contract are not possible and require the creation of a new contract.

Although a smart contract has been stored on the blockchain, this fact alone should not be considered as a party's agreement to enter the contract as anybody can submit any smart contract to

[1] https://coinmarketcap.com/, last access on 01/05/2017

Figure 1 | The life cycle of a smart contract: phases, actors, and services.


Create
Freeze
Execute
Finalize
Activities
Clause Negotiation
Implementation
Renegotiate clauses
Create and communicate transactions to distributed ledger
Retrieve smart contract code and input variables (e.g., from smart oracles)
Compute contract (payout distribution, new internal state)
Actors
Involved parties (A, B, C, ...)
Testing infrastructure
Legal data scientist; SW engineer; etc.
Decentralized execution infrastructure
BC miners
Involved parties (A, B, C, ...)
Services and Tool-Support
Accounts (e.g., wallets)
Implementation environment
Shared distributed ledger (e.g., blockchain)
Accounts (e.g., wallets)

the blockchain indicating an obligation for any random wallet owner. Similarly, decentralized smart contracts can benefit any participant on the blockchain whether they agreed to receiving the benefits beforehand or not.

- **Freeze**: After the smart contract has been submitted to the blockchain, it is persisted by a majority confirmation of the participating nodes. In exchange for this service, and to prevent a flooding of the ecosystem with smart contracts, a fee has to be payed to the miners. From this point onward, the contract and all parties are public and accessible through the public ledger. During the freeze phase, any transfers made to the wallet address of the smart contract are being frozen and the nodes take on the role of a governance board, ensuring the preconditions for executing the contract are met.
- **Execute**: Contracts that are stored on the distributed ledger are read by participating nodes. The contract's integrity is validated and the inference engine of the smart contract environment (compiler, interpreter) executes the code. The inputs for the execution are collected from the smart oracles and involved parties (commitment to goods through coins) and the smart contract's functions are executed. The execution of the smart contract results in a set of new transactions as well as a new state of the smart contract. The set of results as well as the new state information are submitted to the distributed ledger and are validated through the consensus protocol.
- **Finalize**: After the smart contract has been executed, the resulting transactions and the new state information are stored in the distributed ledger and confirmed according to the consensus protocol. The prior committed digital assets are transferred (unfreezing of assets) and with the confirmation of all transactions, the contract has been fulfilled.

4 Drawbacks and limitations

In the following, drawbacks and limitations of decentralized smart contract crypto-currency ecosystems are briefly but not exhaustively sketched.

Blockchain: Besides the potential of blockchain as trustworthy decentralized digital ledgers, they suffer from severe technical and procedural limitations. A popular misunderstanding regards the anonymity of participating parties. Tschorsch et al. state "Blockchain's [are] anything but private" [9, p. 31]. The data stored in the blockchain is publicly accessible, i.e. by everyone involved in the peer-to-peer network. Every party can be identified with a unique wallet id, unless additional cryptographic measures are taken.

The desirable characteristics such as integrity and immutability of data, e.g., transactions and smart contracts, require an active mining pool. Miners need to invest energy and resources, i.e. computational power, to participate in the consensus process and to prevent the corruption of blocks. This investment has to be incentivized to prevent miners from not participating.

Blockchain technology exists in various distributions distinguished by different parameters, e.g., hashing-functions, data scheme of blocks, proof-of-X, etc. Consistency between different blockchains or changing the parameters at run time cannot be done easily and the chosen parameters predetermine the behavior and technological constraints. This can lead to a limited transaction speed of blockchain transactions or to a considerably high block approval time.[2]

Smart Contracts: The combination of a digital contract and its execution in a comprehensive ecosystem is highly attractive, as it can lower the transaction and execution costs significantly. However, a plethora of unsolved issues remain. As in the analog world, the negotiation of the content and the clauses remains.

[2] A Bitcoin blockchain block requires 8 minutes in average to be approved (see https://blockchain.info/de/stats, last access 03/01/2017).

Still, the involved parties need to agree on the contracts and the modalities considering the potentials of a decentralized execution infrastructure.

Since the contract is represented as software code, the implementation effort (see figure 1) must not be neglected. From an organizational point of view, the question arises, which persons have the required skills to implement a smart contract, i.e., translation into machine readable code. In addition, the machine readable code, i.e., programming language, needs to be powerful enough to express the required norms with all necessary conditions, and in the same time it must be restricted to be robust against malicious behavior.

The amount of functionality provided by programming languages depends and can vary throughout different smart contract ecosystems. However, if a Turing-complete language is chosen, severe drawbacks regarding computability exist.

Decentralized crypto-currencies: Those crypto-currencies that rely on blockchain technologies to ensure integrity within a decentralized environment suffer from three main drawbacks: privacy and anonymity limitations, consensus finding, and concurrency. As already mentioned, blockchain technologies are not private (anonymous). Stored transactions (and contracts) can be read by everyone. Without any additional mechanisms, the contracting parties are identifiable via their wallet id occurring within the contract.

Finding consensus is delegated to the chosen consensus mechanism (see [9], pp. 24) and consequently to the participating mining community. However, as Tschorsch stated, the information propagation among honest miners is essential. Otherwise the system becomes fragile and insecure. Related to the propagation time, the problem of concurrency exists. Although, *"double spends are and will always be possible"* ([9], p. 31), the risk of it can be minimized by introducing additional computational complexity to the infrastructure.

5 Conclusion

As technological improvements and innovations made the dream of the 90s of truly decentralized smart contracts possible, law and technology moved closer together. Contracts – or contractual rights and obligations in general – can now be written in code, which is executed without central authority and which can be analyzed from anyone participating in the public ledger, decentralized smart contracts promise to bring fundamental changes to our legal ecosystem.

Each phase of a smart contract's lifecycle promises to decrease costs, increase transparency and trust but also introduces new challenges, costs and incompatibilities with existing laws. As the creation of decentralized smart contracts is split-up into a classic clause negotiation phase and a code implementation phase, fees supposedly saved by needing fewer lawyers are to be weighed against the costs of smart contract programmers.

While the interpretation of non-smart contracts largely depends on their interpretation by the concerned parties and sometimes judges, the results of executing a smart contract can be independently tested by both parties, and the test always yields the same results. The benefit of having a (sometimes unpredictable) judge presiding over the legitimacy and content of a contract is replaced by the merciless judgment of the blockchain's consensus protocol and the immutability of the contract ensured by cryptographic protocols. This solves several (perceived) weaknesses of contract execution, but also introduces new problems and challenges in the real world.

The cryptographically secured interconnection between the execution of the contract and the finalization of resulting payments puts the decentralized execution infrastructure and participating actors into the role of contract enforcers. This implicit dispute resolution mechanism inherently increases transparency and fairness for participating parties. However, the irreversible nature of finalized smart contracts introduces new questions about redress as the resulting set of transactions are by definition immutable and irreversible.

While several parts of a smart contract's life currently require specialized technical knowledge (e.g. creation of wallets, programming a smart contract), a foreseeable popularization of decentralized smart contracts driven by the promise of cost savings, will surely lower existing barriers of entry and lead to the development of user friendly smart contract creation, testing, and sharing environments. Our investigation of the life cycle has shown that there are several issues to be addressed by policy makers and software engineers in order to fully embrace decentralized smart contracts.

Literatur

[1] European Central Bank. *Virtual Currency Schemes*. Report. European Central Bank. (2012) Technical Report.

[2] Vitalik Buterin. *A next-generation smart contract and decentralized application platform*, white paper.

[3] A. Kosba, A. Miller, E. Shi, Z. Wen and C. Papamanthou, *Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts*, 2016 IEEE Symposium on Security and Privacy (SP), San Jose, CA, 2016, pp. 839-858.

[4] Koulu, Riikka. *Blockchains and online dispute resolution: Smart contracts as an alternative to enforcement*. SCRIPTed 13 (2016): 40.

[5] Loi Luu et al. *Making smart contracts smarter*, in Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, ACM, pp. 254-269.

[6] Surden, Harry. *Computable contracts*. UCDL Rev. 46 (2012): 629.

[7] Swan, Melanie. *Blockchain: Blueprint for a new economy*. O'Reilly Media, Inc., 2015.

[8] Nick Szabo. *The idea of smart contracts*, Nick Szabo's Papers and Concise Tutorials 6.

[9] F. Tschorsch and B. Scheuermann. *Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies*, IEEE Communications Surveys & Tutorials (COMST) (18:3), pp. 2084-2123. 2016.