INTERNET POLICY REVIEW
Journal on internet regulation

Smart contracts
**Volume** 10 **Issue** 2

GLOSSARY ENTRY
OPEN ACCESS
PEER REVIEWED

# Smart contracts

**Primavera De Filippi** *Harvard University*
**Chris Wray** *Legal Graph Company Limited* chris.wray@legra.com
**Giovanni Sileno** *University of Amsterdam* g.sileno@uva.nl

**DOI:** https://doi.org/10.14763/2021.2.1549

**Published:** 20 April 2021
**Received:** 18 April 2021 **Accepted:** 4 December 2020

**Competing Interests:** The author has declared that no competing interests exist that have influenced the text.

**Citation:** De Filippi, P. & Wray, C. & Sileno, G. (2021). Smart contracts. *Internet Policy Review*, *10*(2). https://doi.org/10.14763/2021.2.1549

**Keywords:** Blockchain

**Abstract:** A smart contract is code deployed in a blockchain environment, or the source code from which such code was compiled.

This article belongs to the **Glossary of decentralised technosocial systems**, a special section of *Internet Policy Review*.

## Definition

A smart contract is code deployed in a blockchain environment, or the source code from which such code was compiled.

## Origin and evolution of the term

Nick Szabo first described *smart contracts* in the late 1990s. He envisioned placing contracts into code that could be both “trustless” and “self-enforcing”, enhancing efficiency and removing ambiguity from contractual relationships (Szabo, 1996). The idea was to eliminate the need for trust amongst the parties, by increasing the confidence that the contract will be performed exactly as designed (typically making breaches prohibitively expensive). To illustrate his concept, Szabo compared a smart contract to a vending machine. Individuals insert coins into the machine and—assuming the inserted amount is correct—the machine delivers the goods they requested. This predictable interaction requires little to no trust amongst the contracting parties: the vending machine has no choice but to deliver the goods upon receiving the money. The technological infrastructure of the machine is a guarantee that the contract will be fulfilled as intended.

Later, Szabo envisioned that smart contracts could be embedded into all sorts of property that is valuable and controlled by digital technologies to ensure that the associated contractual provisions are automatically executed by technological means (Szabo, 1997). From a historical perspective, the concept of using machines for the application of normative directives can be dated back to Leibniz, with his famous *Calculemus!* (De Arte Combinatoria, 1666), and returned to more concretely with the advent of legal expert systems in AI and attempts at formalisation of law (e.g. Sergot et al., 1984). Szabo’s proposal can thus be seen as a simplification of the higher-level goal set (with mixed results) by research on normative systems.

Today, the term *smart contract* has been adopted by the blockchain community to refer to code deployed and run in a blockchain environment (Buterin, 2013). In this sense, smart contracts are software programmes executed in a distributed manner by the miners of a blockchain-based network. Smart contracts take parameters (as an input) via incoming blockchain transactions, process these parameters according to some deterministic algorithm, and generate (as an output) either a state change in the smart contract memory or a new blockchain transaction.

Although they can be programmed in any language that can be compiled into a particular blockchain environment or virtual machine, the most prominent platform today for the deployment of smart contract code is *Ethereum*. Indeed, the Ethereum blockchain implements a Turing-complete [1] programming language, called *Solidity*, combined with a shared virtual machine (the Ethereum Virtual Machine or EVM), which has become the *de facto* standard for developing and deploying smart contracts. [2] As a programming language, Solidity is object-oriented, with a strong procedural flavour; its core components are imperative instructions defining "positive" actions, like for instance storing the result of a numeric expression in a variable, or logging certain events on the EVM.

Once deployed, the code of a smart contract is stored—in a compiled form—on the Ethereum blockchain and is assigned an address. In order to interact with the smart contract, parties send a transaction to the relevant address, thereby triggering the execution of the underlying code. As such, Ethereum can be regarded as a global and distributed computing layer, which constitutes the backbone for decentralised systems and applications (Buterin, 2013). While Ethereum was the first of its kind, similar functionalities have since been implemented in other blockchain-based platforms, the most popular of which are *Cardano, EOS, NEO, Tezos*, and *TRON*.[3]

Regardless of the blockchain on which they run, smart contracts fundamentally differ from standard software programmes because they can be executed independently from any centralised operator or trusted third party (De Filippi & Mauro, 2014). Indeed, to the extent that they rely on a decentralised network that is not controlled by any single operator (Chen & al., 2017), smart contracts are guaranteed to run in a predefined and deterministic manner, free from intervention by any particular third party (Voshmgir, 2017). Hence, just like a vending machine, smart contracts can be said to be *self-executing*, with a *guarantee of execution* (Buterin, 2013).

1. A programming language is Turing-complete if it is computationally equivalent to a Turing machine. That is, any problem that can be solved on a Turing machine using a finite amount of resources can be solved with that programming language using a finite amount of resources.
2. By contrast, Bitcoin Script is not Turing-complete.
3. Note that, although limited in its capabilities, Bitcoin's simple script language also allows for the creation of custom smart contracts like multisignature accounts, payment channels, escrows, time locks, atomic cross-chain trading, oracles, or multi-party lottery with no operator.

Smart contracts generally only implement basic functionalities, such as:

- token issuance for the purpose of fund-raising (as in the case of a token sale or Initial Coin Offering (ICO));
- issuance and management of tokens as digital collectibles (e.g. *cryptokitties*);
- decentralised marketplaces for the trading of digital tokens (e.g. OpenSea);
- conditional or recurrent payments based on a set of predefined conditions;
- joint savings accounts, allowing parties to withdraw only a particular amount every day;
- escrow systems programmed to execute a transaction whenever specific conditions are met;
- simple lottery systems [4] collecting funds and redistributing them to the selected winner(s);
- gambling systems (such as *prediction markets*) the operations of which are inherently transparent, permitting users to verify how much money the house has on hand for payouts (e.g. Augur).

Yet, by aggregating multiple smart contracts together, it is possible to create applications with more advanced functionalities. These include decentralised finance applications, such as lending platforms (e.g. MakerDAO) and liquidity pools (e.g. Uniswap, Aave); social media platforms (e.g. Akasha, Karma, Peepeth); or even distributed governance systems for blockchain-based assets, often referred to as *Decentralized Autonomous Organizations* (e.g. TheDAO, MolochDAO, DxDAO, etc.).

Perhaps one of the greatest potentials of smart contracts lies in the extent to which they can be used to complement or supplement existing legal contracts. They could be used, for instance, to increase the security of identification phases, to facilitate the subscription for shares in a company, the management of an insurance policy, or even the execution of an employment contract (Alhabry & Van Moorsel, 2017). However, most implementations of smart contracts in the legal field are still far from being widely adopted, or even useful. Indeed, for the majority of legal applications (beyond pure financial applications), much of the computation cannot be done by the smart contracts alone, because the smart contract does not have access to information that is not recorded on a blockchain. This is why many smart contracts rely on so-called "oracles": blockchain addresses controlled by some trusted third parties through which the relevant inputs to the contract are provided. Oracles make it possible for smart contracts to react to external data for the implementation of more sophisticated applications—such as a parametric crop

4. Note that because smart contract code is inherently and necessarily deterministic, randomised action—such as selecting a lottery winner—rely on novel sources of pseudo-randomness which are based on the content of previous blocks.

insurance service, which receives information from a national weather service and automatically disburses funds based on predefined conditions (Cohn & al., 2017). Relevant extensions enabled by oracles concern *ex-post* enforcement mechanisms and dispute resolution by means of witnesses, juries and other roles (e.g. Kleros), or more advanced *ex-ante* enforcement controls by means of external reasoners (see e.g., Idelberg et al., 2016; Liu et al., 2020).

## Misconceptions

There are many misconceptions in the discussion around *smart contracts*. **First**, smart contracts are often believed to be script-like programmes executed on a blockchain, though from a technical perspective, the operations of smart contracts are ultimately defined by the set of instructions fed (in the form of "bytecode") into the virtual machine, which will be executed by the underlying blockchain network. This means that the actual performance of a smart contract does not depend on the subjective expectation of the parties, based on their interpretation of the source code, but merely on the operations dictated by the compiled bytecode deployed to the blockchain (De Filippi & Hassan, 2018).

This leads us to a **second** key misconception about *smart contracts*: they generally act as a technical representation of a legal contract, for at least two fundamental reasons. Firstly, in the *nature* of their expression: smart contracts are inherently more rigid (and therefore more limited) than legal contracts (De Filippi & Wright, 2018). While the clauses of a legal contract (written in natural language) may apply to an indefinite number of situations—because of the inherent flexibility and ambiguity of natural language—the provisions of a smart contract are expressed in a formalised language that does not have nearly the same degree of flexibility as natural language (Levy, 2017). As a result, many contractual clauses (e.g. *bona fide* obligations) cannot be codified in a blockchain-based infrastructure because they simply cannot be expressed in code (Sklaroff, 2017). Rigidity is also partially due to the closure determined by specific technological choices; for instance, although Solidity considers the use of libraries (i.e. reusable smart contract deployed code), those cannot be updated, and their semantic staticity is reflected in the contracts relying upon them. That being said, such limitations also represent one of the key benefits of a smart contract, as contracting parties may want their contractual performance to rely exclusively on precise and quantifiable outcomes.

Secondly, in the *scope* of their performance: only a very limited class of contractual obligations can be fully embedded into a smart contract (Mik, 2017). At a computational level, smart contracts enjoy the convergence of imperative instructions with positive duties, but this also means that they do not include explicit directives about e.g. prohibitions, nor about institutional power. This would not really be problematic if smart contracts were only concerned with operations under their control. However, most legal contracts refer to rights and obligations outside of the blockchain infrastructure, which cannot therefore be administered via a smart contract. If contractual obligations are triggered by external conditions, a smart contract will depend on a third party-operated programme (i.e. an "oracle") to record all the relevant information about such external conditions onto a blockchain (Egberts, 2017). If the contractual obligation itself requires an external intervention, no blockchain-based infrastructure will ever be able to guarantee the proper performance thereof. In particular, legal title to, or beneficial interest in, any property or asset that exists outside of the blockchain infrastructure (i.e. anything other than a blockchain-based asset) cannot be transferred merely by recording a state change into a blockchain, but only in accordance with applicable law. For instance, transferring land ownership cannot be performed automatically by a smart contract because it requires administrative formalities that cannot be completed on a blockchain. In this case, a smart contract could only record the payment, along with the current owner's intention to transfer ownership to a third party—e.g. via the transfer of an asset-backed token.

Sometimes, the mere act of transacting with a smart contract could give rise to a legal agreement, provided that the minimum legal requirements for contract formation are met in the relevant jurisdiction (Werbach & Cornel, 2017). Conversely, any additional provisions that cannot be fully codified in (and therefore automated by) a blockchain will merely qualify as a promise under an executory contract that may only be enforced through a court order (Herian, 2020). Thus, just as a vending machine can automate the performance of a contract to sell only the physical goods contained within it, so a blockchain-based smart contract can provide automatic performance of a contract relating only to transactions in blockchain-based assets (Hulicki, 2017).

A related problem is the impossibility of technically *nullifying* the execution of a smart contract in case some underlying conditions make its execution invalid from a legal point of view. Even if such a situation could be identified by means of an external oracle, the chain of transactions stemming from an invalid performance cannot be recovered, unless the possibility has been pre-codified within the smart contract itself.

**Several other** misconceptions about smart contracts are related to trust. First, it is often said that smart contracts are entirely self-executing (Zhou et al., 2019). Yet, as highlighted above, a smart contract will always rely on a certain amount of trust and/or verification, especially when its execution depends on external information recorded onto a blockchain by a third party (Guadamuz, 2019). If the smart contract depends on a given “oracle” for its basic functionality, the failure of such an oracle to provide the necessary information will prevent the execution of the smart contract (Muhlberger et al., 2020). More fundamentally, a smart contract’s proper functioning ultimately depends on the network of miners that operate the underlying blockchain network (De Filippi et al., 2020). Were these miners collectively to decide to prevent the execution of a smart contract, they could either censor all transactions addressed towards that particular smart contract’s address (a soft fork) or modify the blockchain protocol in order to change the code of the smart contract or its implementation (a hard fork). While such an intervention is unlikely to happen on a recurrent basis, it is not merely theoretical—as shown by the hard fork of the Ethereum blockchain in the aftermath of TheDAO attack [5] (Reijers et al., 2018).

## Conclusion

A *smart contract* is code deployed in a blockchain environment, or the source code from which such code was compiled. It is executed in a distributed manner by the miners of the underlying blockchain network if and when the underlying conditions are met. Execution of a smart contract is triggered via a blockchain transaction and will produce a change in the blockchain state.

---

## References

Alharby, M., & Van Moorsel, A. (2017). Blockchain-based smart contracts: A systematic mapping study. *Computer Science & Information Technology*, *7*(10). https://doi.org/10.5121/csit.2017.71011

Buterin, V. (2013). *Ethereum whitepaper: A next-generation smart contract and decentralized application platform* [White Paper]. https://ethereum.org/en/whitepaper/

Chen, L., Xu, L., Shah, N., Gao, Z., Lu, Y., & Shi, W. (2017). Decentralized execution of smart contracts: Agent model perspective and its implications. In M. Brenner, K. Rohloff, J. Bonneau, A. Miller, P. Y. A. Ryan, V. Teague, A. Bracciali, M. Sala, F. Pintore, & M. Jakobsson (Eds.), *International conference on financial cryptography and data security* (pp. 468–477). Springer. https://doi.org/10.1007/978-3-319-70278-0_29

5. TheDAO was a decentralised investment fund deployed as a smart contract on the Ethereum blockchain in 2016, which raised over USD$ 150 million in less than one month. However, a few days before the launch, a vulnerability was found in the code of the smart contract governing TheDAO, which was exploited in order to drain over USD$ 60 million from the fund.

Cohn, A., West, T., & Parker, C. (2017). Smart after all: Blockchain, smart contracts, parametric insurance, and smart energy grids. *Georgetown Law Technology Review*, *1*(2), 273–304. https://georgetownlawtechreview.org/smart-after-all-blockchain-smart-contracts-parametric-insurance-and-smart-energy-grids/GLTR-04-2017/

Corrales, M., Fenwick, M., & Haapio, H. (Eds.). (2019). *Legal Tech, Smart Contracts and Blockchain*. Springer. https://doi.org/10.1007/978-981-13-6086-2

De Filippi, P., & Hassan, S. (2018). Blockchain technology as a regulatory technology: From code is law to law is code. *arXiv*. https://arxiv.org/abs/1801.02507

De Filippi, P., Mannan, M., & Reijers, W. (2020). Blockchain as a confidence machine: The problem of trust & challenges of governance. *Technology in Society*, *62*. https://doi.org/10.1016/j.techsoc.2020.101284

De Filippi, P., & Mauro, R. (2014, August 25). Ethereum: The decentralised platform that might displace today's institutions. *Internet Policy Review*. https://policyreview.info/articles/news/ethereum-decentralised-platform-might-displace-todays-institutions/318

De Filippi, P., & Wright, A. (2018). *Blockchain and the law: The rule of code*. Harvard University Press.

Egberts, A. (2017). *The oracle problem-an analysis of how blockchain oracles undermine the advantages of decentralized ledger systems*. https://doi.org/10.2139/ssrn.3382343

Guadamuz, A. (2019). All watched over by machines of loving grace: A critical look at smart contracts. *Computer Law & Security Review*, *35*(6). https://doi.org/10.1016/j.clsr.2019.105338

Herian, R. (2020). Smart contracts: A remedial analysis. *Information & Communications Technology Law*, *30*(1), 17–34.

Hulicki, M. (2017). The legal framework and challenges of smart contract applications. *Conference on System Sciences*, 3–4. http://www.cs.bath.ac.uk/smartlaw2017/papers/SmartLaw2017_paper_3.pdf

Idelberger, F., Governatori, G., Riveret, R., & Sartor, G. (2016). Evaluation of logic-based smart contracts for blockchain systems. Rule technologies. In J. J. Alferes, L. Bertossi, G. Governatori, P. Fodor, & D. Roman (Eds.), *Rule Technologies. Research, Tools, and Applications* (pp. 167–183). Springer International Publishing. https://doi.org/10.1007/978-3-319-42019-6_11

Lauslahti, K., Mattila, J., & Seppala, T. (2017). *Smart contracts–How will blockchain technology affect contractual practices?* (Report No. 68). ETLA. http://hdl.handle.net/10419/201350

Levy, K. E. (2017). Book-smart, not street-smart: Blockchain-based smart contracts and the social workings of law. *Engaging Science, Technology, and Society*, *3*, 1–15. https://doi.org/10.17351/ests2017.107

Liu, L., Sileno, G., & Engers, T. V. (2020). Digital enforceable contracts (DEC): Making smart contracts smarter. In S. Villata, J. Harašta, & P. Kremen (Eds.), *JURIX 2020: The 33rd annual conference on legal knowledge and information systems* (pp. 235–238). https://doi.org/10.3233/FAIA200872

Mik, E. (2017). Smart contracts: Terminology, technical limitations and real world complexity. *Law, Innovation and Technology*, *9*(2), 269–30. https://doi.org/10.1080/17579961.2017.1378468

Mühlberger, R., Bachhofner, S., Ferrer, E. C., Di Ciccio, C., Weber, I., Wöhrer, M., & Zdun, U. (2020). Foundational oracle patterns: Connecting blockchain to the off-chain world. *International Conference on Business Process Management*, 35–51. https://doi.org/10.1007/978-3-030-58779-6_3

Reijers, W., Wuisman, I., Mannan, M., De Filippi, P., Wray, C., Rae-Looi, V., Vélez, A. C., & Orgad, L. (2018). *Now the code runs itself: On-chain and off-chain governance of blockchain technologies*. 1–11. https://doi.org/10.1007/s11245-018-9626-5

Savelyev, A. (2017). Contract law 2.0: 'Smart'contracts as the beginning of the end of classic contract law. *Information & Communications Technology Law*, *26*(2), 116–134. https://doi.org/10.1080/13600834.2017.1301036

Sergot, M. J., Sadri, F., & Kowalski, R. A. (1986). The British Nationality Act as a logic program. *Communications of the ACM*, *29*(5). https://doi.org/10.1145/5689.5920

Sklaroff, J. M. (2017). Smart contracts and the cost of inflexibility. *University of Pennsylvania Law Review*, *166*, 263–303. https://scholarship.law.upenn.edu/prize_papers/9/

Szabo, N. (1996). Smart contracts: Building blocks for digital markets. *EXTROPY: The Journal of Transhumanist Thought*, *16*.

Szabo, N. (1997). *The idea of smart contracts*. Nick Szabo's Papers and Concise Tutorials. https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_idea.html

Voshmgir, S. (2017). Disrupting governance with blockchains and smart contracts. *Strategic Change*, *26*(5), 499–509. https://doi.org/10.1002/jsc.2150

Werbach, K., & Cornell, N. (2017). Contracts ex machina. *Duke Law Journal*, *67*(2), 313–382. https://scholarship.law.duke.edu/dlj/vol67/iss2/2/

Zou, M., Cheng, G., & Soria Heredia, M. (2019, April). In code we trust? Trustlessness and smart contracts. *Computers and Law*. https://www.scl.org/articles/10493-in-code-we-trust-trustlessness-and-smart-contracts

Published by

Alexander von Humboldt
INSTITUTE FOR INTERNET
AND SOCIETY

in cooperation with

CREATe

centre internet et société

R&I IN3 Internet interdisciplinary Institute
Universitat Oberta de Catalunya