The Cambridge Handbook of Smart Contracts, Blockchain Technology and Digital Platforms

Edited by
Larry A. DiMatteo
University of Florida

Michel Cannarsa
Lyon Catholic University

Cristina Poncibò
University of Turin

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE
UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre, New Delhi – 110025, India

79 Anson Road, #06–04/06, Singapore 079906

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of education, learning, and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9781108492560
DOI: 10.1017/9781108592239

First published 2020

Printed in the United Kingdom by TJ International Ltd. Padstow Cornwall

A catalogue record for this publication is available from the British Library.

ISBN 978-1-108-49256-0 Hardback

Summary of Contents

7 Smart Contracts

Contractual and Noncontractual Remedies

Cristina Poncibò is the author of paragraphs 7.1, 7.3, 7.4, 7.5, 7.6
Larry A. DiMatteo is the author of paragraphs 7.1 and 7.2

7.1 Introduction

This chapter will examine the impact of smart contracts' ability to self-perform, self-enforce, and self-remedy and the remaining applicability of contract law and contract remedies.[1] Smart contracts (coupled with blockchain technology)[2] have created visions of self-executing, self-enforcing, and self-remedying contracts that eliminate the need for courts or arbitral tribunals to apply contract law to disputes. The theory goes that since the possibility of breach is eliminated in such contracts, contract remedies become unnecessary.

It is best to start the current discussion with defining three important concepts – self-enforcement, self-help remedies, and "other remedies." Self-enforcement can be analogized to the remedy of specific performance (albeit with the extra step of obtaining a judicial order to perform). Self-help remedies may be analogized, in some cases, as a form of the remedy of injunction, such as the disablement of the time of use of the subject of the contract. Other remedies refer to the menu of remedies provided under contract law and their continued position as default law. The remaining importance of contract law remedies is premised on the view that no matter the degree of self-enforcement and the creation of self-help remedies in smart contracts, the parties will continue to have the ability to seek redress before courts and arbitral tribunals.

Just as the false argument that word contracts can be made to be clear and complete, the completeness of smart contracts is an illusion.[3] One commentator noted: "Terms of contracts, which are more complex than the immediate transfer of value and property are

[1] Portions of this chapter have become a shorter version of the analysis given in L. DiMatteo and C. Poncibò, "Quandary of Smart Contracts and Remedies: Role of Contract Law and Self-Help Remedies" (2019) *ERPL* (*European Review Private Law*) 805.

[2] A simple definition of smart contracts is stated as follows: "Smart contracts are self-executing electronic instructions drafted in computer code. This allows a computer to 'read' the contract and, in many cases, effectuate the instruction – hence the 'smartness' of the contract." R. O'Shields, "Smart Contracts: Legal Agreements for the Blockchain" (2017) 21 *N.C.B.I. (North Carolina Banking Institute)* 177 at 179. Smart contracts are different than ordinary electronic contracts in that the actual agreement is embodied in computer code, rather than English or another traditional language. See N. Szabo, "Smart Contracts: Building Blocks for Digital Markets" (1996), http://szabo.best.vwh.net/smart_contracts_2.html (coined the term "smart contracts").

[3] G. Cordero-Moss, "Interpretation of Contracts in International Commercial Arbitration: Diversity on More than One Level" (2014) 22 *ERPL* at 13 (sufficiently detailed and clear contracts can be interpreted internally "without them being influenced by any governing law. This impression has proven to be illusionary").

likely to not be efficiently encoded."[4] Thus, there remains a significant amount of "risk of *divergence* expressed in natural language between the meaning of the original contractual provision and its expression in code."[5] If word contracts are hopelessly incomplete, how can word contracts translated into code be any less incomplete? In the end, code is no different than the use of words since both are creations of human beings, meaning that perfect completeness will never by achievable. Also, contracts are always incomplete because parties cannot foresee all future events that may impact the parties' expectations related to their contract. Again, contract remedies will continue to play a role due to the unpredictability of future events.[6] The issue analyzed in this chapter is whether a complete set of remedies can be incorporated into smart contracts to preclude the need for contract law remedies in a default role. The chapter concludes that contract remedies will remain relevant, but recognizes that smart contracts may fill a void where enforcement of traditional remedies is not practical or too costly.

It is in the area of self-enforcement and remedies where the vision of smart contracts confronts the reality of contract law and business lawyering. Smart contracts need to be drafted by lawyers, focused on client interests and not technological prowess. In order for lawyers to best serve their clients, they would have to learn to write computable code, while judges would have to learn codes to interpret the contract or rely on an expert translation. Assuming that there is only a single interpretation of a computer code, does the issue over the correct or reasonable legal interpretation of contracts miraculously resolve itself? Put simply, is this a step too far in the advancement of the self-executing, self-enforcing smart contracts? Code will need to be converted into words, which reengages the same quandaries that have persisted in contract interpretation over the centuries.

Again, this chapter will provide some propositions relating to the role of remedies relating to the assertion that smart contracts are a self-enforcing legal regime. If smart contracts are completely self-enforcing, then there is no need for a remedial scheme. If smart contracts are not completely enforcing, can they provide a self-contained remedial system? Even if that is possible, the argument here is that self-enforcement or self-remedialization would still need to conform to the immutable rules of contract law. An unenforceable clause under contract law cannot be made enforceable simply by embedding it into a code; a contract term that is considered illegal or against public policy cannot be made legal in a smart contract. Finally, if a smart contract is too simple, then it may fail as to indefiniteness of terms.[7] In the end, the judicial remedies of adjustment (reformation) or voidance (rescission) will play the same roles in coded contracts as they

[4] S. Farrell, H. Machin, and R. Hinchliffe, "Lost and found in smart contract translation-considerations in transitioning to automation in legal architecture" (King & Wood Mallesons, Australia), 3, www.uncitral.org/pdf/english/congress/Papers_for_Programme/14-FARRELL_and_MACHIN_and_HINCHLIFFE-Smart_Contracts.pdf (last viewed April 22, 2018).

[5] Ibid.

[6] From a wider perspective of the public dimensions of private law, one scholar states that: "providing a private system of enforcement through code does not eliminate the state's role in contract." Mark Verstraete, "The Stakes of Smart Contracts" (2019) 50 (*Loyola University Chicago Law Journal*) 47 at 50.

[7] The indefiniteness of a contract may lead to its unenforceability (as a noncontract): "Certainty as to what constitutes the contractual terms (and whether they are comprehensive enough) is often a critical factor necessary to establish the formation of a legally binding contract in many jurisdictions. Smart contracts . . . may not satisfy such requirements." Norton Rose Fulbright White Paper, "Can Smart Contracts be Legally Binding Contracts: Key Findings" 4, www.nortonrosefulbright.com/files/norton-rose-fulbright–r3-smart-contracts-white-paper-key-findings-nov-2016–144554.pdf.

do in language contracts. Ultimately, courts will remain relevant in the enforcement and interpretation of smart contracts.

7.1.1 *Illusion of Self-Sufficiency*

The idea of a self-sufficient contract (fully comprehensive, perfect clear) is a cognitive impossibility. The claim that smart contracts can be fully self-executing belies the continuing role of contract law's interpretive and remedial functions in all types of contracts – whether smart or dumb. The fact that smart contracts may be self-executing does not mean that they overcome the problems of interpretation or prevent a party from seeking a remedy in court. Self-enforcement does not mean that human agents cannot move to block or prevent enforcement. While self-help remedies can be included in smart contracts, they do not block the use of the range of remedies that parties may seek under the general law of contracts.

If smart contracts are to claim the mantel of a fully self-contained, privatized legal system, then it will need to be fully self-enforcing and self-remedying. Can all issues of performance be converted into computer code? Is enforcement of a smart contract immune from the interpretive problems associated with ordinary contracts? These questions directly relate to the issue of contract remedies and their applicability to smart contracts. At the present, the conclusion is that contract law and judicially created remedies will continue to play a prominent role.

7.1.2 *Inflexibility of Smart Contracts*

Smart contracts are ultimately limited by their inflexibility. One of the great features of contract law is its inherent flexibility and malleability. It allows parties to improvise new types of contract terms that are customized to different types of contracts. In long-term, relational, and complex contracting, the contract is a combination of fixed terms, open-textured rules, and standards.[8] Smart contracts currently are only capable of replicating hard or bright line fixed terms or highly formalized rules. It is no surprise that smart contracts have first made their mark in financial transactions since banking and finance laws are based on highly formalized rules with very little room for adjustment or application of standards-like reasonableness.[9] It is these types of formal rules and terms that are more easily translatable to code.

The inflexibility of smart contracts versus word contracts provides a major obstacle to smart contracts expansion into the realm of more complex contracting where self-enforcement and self-remedying features may prove unattractive and inefficient.[10] Some contractual terms simply cannot be expressed through formal logic, because they

[8] Open-textured rules or terms are those that encourage the interpreter to look to the real world to find the correct interpretive meaning. The robust use of the reasonableness standard in Article 2 of American Uniform Commercial Code (UCC) is an example of an open-textured rule. Open-textured rules allow for the use of contextual input in the defining and application of the rule. See W. Twining *Llewellyn Papers* (University Chicago Law School Press 1968), 86.

[9] See UCC Article 3 (Negotiable Instruments) and Article 9 (Secured Transactions).

[10] Although one commentator notes that a degree of flexibility may be obtained by the use of a subsequent smart contract to amend or adjust an existing smart contract: "One possibility for achieving the effect of a reversal or change in terms is to create that result through the creation of a new smart contract which, when added to the existing contracts, has the effect of the desired reversal or change." H. Farrell,

imply human judgment. For example, a machine has no precise way to assess whether a party used "best efforts."[11] Word contracts help overcome the unpredictability of future events (change of circumstances) through provisions, which require a performance adjustment and the appropriate remedial response that smart contracts are unable to perform.

7.1.3 *Smart Contracts and Remedies*

Smart contracts may be outside the law, but they are not above the law.[12] The use of smart contracts to escape the legal system will not render traditional contractual defenses useless. Smart contracts' self-enforcement ability does not prevent self-enforcement from being subject to post hoc judicial review. In such cases, the remedies of restitution and disgorgement of profits may become more common since performance is not subject to breach.

The continuing importance of contract law remedies is supported by the role of such remedies in *ex post* adjudication. The *ex ante* view of contract law with its focus on the time of contract formation (consent of the parties), which is the domain of smart contracts, ignores contract law's ex post regulatory and remedial function.[13] In this regard, smart contracts cannot transplant contract law.[14]

An example of the irreplaceability of contractual remedies can be shown through a simple example. A transfer of an asset is executed by a smart contract, but is later invalidated in court due to fraud, duress, incapacity, illegality, and so forth. In the blockchain the asset remains the property of the transferee, but in the real world the law recognizes the title being held by the transferor. The only way to square this bifurcated ownership is for the court to issue an order of specific performance requiring the transferee to re-convey ownership through the blockchain.[15]

Finally, the problem of code writing and the possibility of contamination through a virus may lead to a breach of a smart contract: "The code embedding the contract terms can contain bugs or produce results that are not in accordance with the expectations of the parties."[16] Therefore, the self-enforcement can be viewed as a breach in cases where "its performance would not be as expected or intended by the parties."[17] Because of such possibilities, smart contracts may lead to the recognition of *ex contractu* legal remedies. The one that comes to mind is the tort of negligence. The divergence between a smart contract as intended and expectations of outcome could result in the emergence of torts

H. Machin, and R. Hinchliffe at 6. However, the speed of smart contract execution and the need of the parties to negotiate to any such changes may prevent a timely adjustment.

[11] D. Werbach and N. Cornell "Contracts Ex Machina" (2017) 67 *DLJ (Duke Law Journal)* 313–69, at 365.

[12] Farrell, Machin, and Hinchliffe at 2.

[13] Werbach and Cornell, note 11, at 361.

[14] "Smart contracting functions to ensure action. Contract law functions to recognize and remedy grievances. Smart contracts could not – even in theory – replace contract law." Ibid. at 363.

[15] A. Savelyev, "Contract Law 2.0: Smart Contracts As the Beginning of the End of Classic Contract Law" (2017) 26 *ICTL (Information & Communications Technology Law)* 1–19, available at https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2885241.

[16] S. Hourani, "Cross-Border Smart Contracts: Boosting International Digital Trade Trough Trust and Remedies," www.uncitral.org/pdf/english/congress/Papers_for_Programme/11-HOURANI-Cross-Border_Smart_Contracts.pdf.

[17] Ibid.

"for negligent coding or negligent updating."[18] One could also imagine cases of intentional miscoding as the basis for an action of misrepresentation or fraud.

7.2 Trust as Remedy

Contracts allow parties to bind themselves to undertake some action in the future. Enforceable contracts are the legal vehicle that allows market economies to form and are the engine of capitalism. A market economy is impossible without the recognition of private property rights and the ability to transfer those rights through contracts. Contracts are a substitute for purely trusting the other party to perform. The ability to bring a claim for breach of contract through litigation or arbitration provides a party with a level of certainty that the other party will perform in the future. The strongest contracts, however, are those that are never enforced because they reflect a certain degree of trust between the parties.

According to Francis Fukuyama, trust is the single most important factor in creating a high-performing national economy. In his book *Trust: The Social Virtues and the Creation of Prosperity*,[19] Fukuyama argues that cooperative economic behavior rests upon a "culture of trust." Trust has traditionally been anchored in private contract or government intervention (regulation). Blockchain offers a third possibility often referred to as "trustless trust." The self-verifying, self-enforcing nature of smart contracts provides a level of certainty (some argue absolute certainty) beyond traditional trust structures. The next section examines the variety of trust structures or institutions and the idea of trustless trust offered by smart contracts.

7.2.1 *Architecture of Trust for Smart Contracts*

Until blockchain technology, there were two primary trust architectures: Leviathanian (deference to a central enforcement authority, generally a government regulatory agency) and peer-to-peer (reliance on social norms and other governance mechanisms in self-contained communities, such as the diamond[20] and cotton[21] industries). One commentator has also advanced the idea of creating a Leviathan for the blockchain by establishing a public "Superuser" (government authority) with the power to modify the content of Blockchain databases.[22]

Werbach argues that blockchain introduces a third kind of trust architecture, "trustless trust," which is a system that makes it "possible to trust the outputs of a system without

[18] M. Raskin, "The Law and Legality of Smart Contracts" (2017) 1 *GLTR* (*Georgetown Law Technology Review*) 305, at 328.

[19] F. Fukuyama, *Trust: The Social Virtues and the Creation of Prosperity* (New York: The Free Press 1995).

[20] Lisa Bernstein, "Opting Out of the Legal System: Extralegal Contractual Relations in the Diamond Industry" (1992) 21 *JLS (Journal of Legal Studies)* 115–57. See also B. D. Richman, "Community Enforcement of Informal Contracts: Jewish Diamond Merchants in New York," The Harvard John M. Olin Discussion Paper Series, Discussion paper No. 384 (9/2002), 1–55.

[21] L. Bernstein "Private Commercial Law in the Cotton Industry: Creating Cooperation through Rules, Norms, and Institutions" (2001) 99 *UMLR (University of Michigan Law Review)* 1724.

[22] Savelyev, above note 15. For a map of experiences, see M. Alharby, A. van Moorsel, Blockchain-based Smart Contracts: A Systematic Mapping Study, study presented at the Fourth International Conference on Computer Science and Information Technology (CSIT-2017), 125–40.

trusting any actor within it."[23] Yet, Werbach warns that blockchain's technical reliability should not be confused with trust. He says: "Trust is a manifestation of goodwill of the one being trusted – it involves a positive expectation about the other party and a willingness to be vulnerable."[24] Blockchain technology has neither of these attributes. In place of trust as a substantive virtue, the blockchain provides technical or procedural type of trust.

Secured through cryptography, implemented through distributed consensus, networks replace the personal trust found in most commercial contracts.[25] In Nick Szabo's words, "Trust-minimized code means you can trust the code without trusting the owners of any particular remote computer."[26] The nature of distributed ledgers makes certain activities trustworthy without the need to trust anyone in particular. Blockchain proponents argue that as a result, the costly mechanisms of intermediation and legal enforcement can be avoided. Instead of trusting banks, courts, and governments, blockchain proponents suggest, we can trust math and computation, in the form of open-source cryptographic protocols. If they are even slightly correct, the potential is extraordinary, due to the lower costs and greater efficiency of such a technological trust substitute.

For libertarians, these technologies represent economic activity outside the bounds of sovereign state control. Currently, the regulatory oversight over blockchain-based transactions is limited. Courts arguably will find it extremely difficult to determine jurisdiction over blockchain-based smart contracts due to party anonymity at the core blockchain transactions. Furthermore, the smart contract's performance is a fait accompli since the coded transaction is auto-executed on the blockchain, so, technically, there can be no claim for breach of contract.

This type of new trust architecture has powerful implications. The theory of governance of common-pool resources laid the groundwork for management of important resources as commons.[27] It has proven particularly significant in the digital economy, as an approach to both network infrastructure such as wireless spectrum and creative works such as software. "Trustless trust" has similar potential. But its implications are still unknown since the issues and consequences of the expanded use of the blockchain are undertheorized. Part of this lack of knowledge is due to the separation of the technological and legal elements of the technology. Thus, true acumen is only achievable by lawyers understanding the technological aspects of smart contracts and technicians understanding of the legal impact of the technology.

As noted above, the architecture of trustless trust provided by the blockchain is the basis for the argument that the technology can act as an alternative to the law. However, despite the expectation of the flawlessness of mathematically based systems, blockchains are systems designed, implemented, and used by humans. Subjective intent remains relevant

[23] K. D. Werbach, "Trust, But Verify: Why the Blockchain Needs the Law" (2016), Conference Paper, 1–74. Accessed October 23, 2018 at https://ssrn.com/abstract=2844409

[24] Ibid. 15.

[25] A. Walch, "In Code(rs) We Trust: Software Developers as Fiduciaries in Public Blockchains," G. Dimitropoulos, S. Eich, P. Hacker, and I. Lianos (eds.), *The Blockchain Revolution: Legal and Policy Challenges* (Oxford University Press 2018).

[26] N. Szabo, "The Dawn of Trustworthy Computing," Unenumerated Blog (December 11, 2014), http://unenumerated.blogspot.com/2014/12/the-dawn-of-trustworthycomputing.html (last accessed October 23, 2018).

[27] E. Ostrom, "Beyond Markets and States: Polycentric Governance of Complex Economic Systems," Nobel Prize Lecture (December 8, 2009), www.nobelprize.org/nobel_prizes/economicsciences/laureates/2009/ostrom_lecture.pdf Precisely, Ostrom uses the expression "governance of common-pool resources."

even when expressed through objective code. Blockchains are subject to selfish behavior, attacks, and manipulation. The scope of legitimate practices for blockchain-based systems is fundamentally a governance question, not a computer science one. Therefore, the efficiency of the blockchain in executing simply transactions (fund and information transfers) should not be seen as rendering law useless.

Scholars disagree on the relationship between code and law. Wright and de Filippi have predicted that blockchain will generate a new body of rules, "Lex Cryptographia," that is independent from state-created legal rules.[28] To the contrary, Werbach argues that, in most cases, economic agents will prefer to have the ability to enforce legal rights in court or by arbitration, as opposed to relying solely on the output of the blockchain. This recourse to legal institutions may, however, be limited if dispute resolution mechanisms are embedded in the underlying code.[29] However, this is unlikely to prevent a party from challenging the enforceability of that mechanism in an effort to seek legal remedies.

Due to its peculiar architecture smart contracts have been characterized as an independent "state of nature" outside the control of the state.[30] Such an outside law or private law system has been the subject of long-term inquiry by scholars, such as Anthony Kronman, Richard Posner, and Stuart Macaulay.[31] With respect to methods and contents, this is a diverse literature, but one with a common theme. Each of the writers is interested in the opportunities that individuals exploit, and the arrangements they invent, to enhance the security of their agreements where no legal remedies for breach exist or where those that do are plainly inadequate. Smart contracts seem to overcome these weaknesses in the traditional legal system.

7.3 Self-Help for Smart Contracts

This section analyzes smart contracts from the remedial perspective of contract law. Can smart contracts be a substitute for the remedies offered by contract law? This begs another question: Can there ever be a truly breach-less contract? The section concludes that the answers to these questions are in the negative. This conclusion rests on the commonsensical belief that at times smart contracts will diverge from party expectations and desired outcomes. Such divergence may occur due to mistakes or bugs (coding error), changes of circumstances, and cyber attacks (viruses). The fit of traditional contract law to the acceleration of technology is captured in the following statement: "Contract today increasingly links entrepreneurial innovations to the efforts and finance necessary to transform ideas into value."[32] The issue for contract law is that it is adaptable to technological innovation, such as in the case of the blockchain. The answer is likely

[28] See A. Wright and P. De Filippi, "Decentralized Blockchain Technology and the Rise of Lex, Cryptographia" (2015), 1–58. http://papers.ssrn.com/sol3/papers.cfm?abstractid=2580664 (last accessed November 10, 2018); M. Abramowicz, Cryptocurrency-Based Law (2018) 58 *ALR (Arizona Law Review)* 359.

[29] K. D. Werbach, "Trust, But Verify: Why the Blockchain Needs the Law" (2017), https://ssrn.com/abstract=2844409 (last accessed July 20, 2018).

[30] Kronman used the expression "state of nature" with respect to Contract Law. See the note below.

[31] S. Macaulay, "Non-Contractual Relations in Business" (1963) 28 *American Sociological Review* 55. A. Kronman, Contract Law and the State of Nature (1985) 1 *Journal of Law, Economics and Organization*, 5–32. R. Posner, *The Economics of Justice* (Cambridge: Harvard University Press 1981).

[32] R. J. Gilson, C. F. Sabel, and R. E. Scott, "Contract, Uncertainty, and Innovation" in S. Grundmann, F. Möslein, and K. Riesenhuber, *Contract Governance: Dimensions in Law and Interdisciplinary Research* (Oxford University Press, 2015) 156.

to be that the inherent flexibility of contract law, premised on freedom of contract and justice norms,[33] is adaptable to such change. A follow-up question is whether technological change results in a new transaction type? If so, does general contract law suffice or will a specialized rules regime be required?[34]

It is reasonable to expect that platform providers and tech developers will primarily rely on the blockchain ecosystem to solve any problems, such as mistake and change of circumstances arising out from a smart contract. The law will remain in the "shadows." But most contracts work within the shadow of the law. This is because most contracts are fully performed, performed enough to satisfy the other party, are readjusted through party negotiation, or simply "run out" because dispute resolution costs are prohibitive. In a small percentage of contracts, these mediating mechanisms fail and this is when the law comes out of the shadows. In the case of smart contracts, it must be recognized that no matter how self-enforcing the contract may be, self-enforceability does not prevent a party from seeking redress in a court of law.

With the lens of a comparative lawyer, one may say that blockchain technology, with its peculiarities (anonymity, decentralization), is equally "disrupting" the traditional civil and common law approaches to contracting, especially with respect to remedies. On the one hand, common law contract theorists have devoted particular attention to examining remedies for breach of contract. Part of the explanation for this attention is that remedies have played a crucial role in the development of the common law.[35] In the medieval writ system from which the modern common law evolved, causes of action were framed primarily in terms of remedies sought. The common law lawyer has thus traditionally approached contract issues through a remedial lens – a tradition kept alive in the practice of many common law faculties by starting contract courses with the subject of remedies. In addition, remedial issues are frequently of decisive importance in litigation and it is from litigation that the common law is developed. Because of this remedial perspective, the common law tends to prefer ex post enforcement, which is the opposite approach to smart contracts where all issues are viewed ex ante.[36]

The architecture of "trust" upon which smart contracts are based has two fundamental consequences for contract law. First, as noted above, smart contracts' automation shifts the focus on ex ante compliance instead of ex post contractual remedies. Currently, the most prevalent forms of security software methods include virus-scanning software, filtering firewalls, and detecting the traceroutes of attackers.[37] By using smart contracts, the parties are radically changing the paradigm of contract practice from ex post authoritative judgment to ex ante-automated assessments. Proponents of smart contracts believe that ex-ante automation results will only infrequently diverge from expected outcomes

[33] L. DiMatteo, "The Norms of Contracts: The Fairness Inquiry and the 'Law of Satisfaction' – A Nonunified Theory" (1995) *Hofstra Law Review* 349.

[34] Specialized rules can be developed within contract law or through statutory intervention. Examples are plentiful, including specialized rules for sales, negotiable instruments, letters of credit, consumer transactions, and so forth.

[35] See, P. Atiyah, *The Rise and Fall of Freedom of Contract* (Oxford: Clarendon Press 1979). G. Gilmore *The Death of Contract* (Columbus, OH: Ohio State University Press 1974).

[36] S. Issacharoff, "Regulating after the Fact" (2007) 56 *DePaul L. Rev.* 375.

[37] E. Mik, "Smart Contracts: Terminology, Technical Limitations and Real World Complexity" (2017) 9 (2) *Law, Innovation and Technology*, 269–300.

with the benefit of avoiding the costs of post hoc dispute resolution. Under this view the benefits of ex ante efficiency far outweigh bearing the costs of ex-post corrections.[38]

Second, the problem of avoiding injustice caused by error (improper coding) or changes in circumstances means that any such failures related to smart contracts, should be anticipated with coding that allows the triggering of self-help remedies or contractual adjustments. If such self-remedying provisions can be coded, then recourse to the courts to cure unjust outcomes will be lessened. This would require the loosening of the self-contained nature of blockchain technology to external triggers.

The central tenet of contract law is the freedom of parties to form and govern their own relationships. The creation and use of self-help remedies is within the scope of freedom of contract. Self-help enables the parties to keep their affairs private, out of the reach of interfering official bodies, thus creating a regulatory sandbox without supervision.[39] Also, platforms and developers have shown the desire to maintain individual control over the terms of the relationship, as well as a dislike or distrust for courts, which will incentivize them to create self-help remedies.

In contract law scholarship, the term "self-help" refers to private actions used to prevent or resolve disputes without assistance of a governmental official or disinterested third party.[40] The situations in which self-help may be invoked and the actions, which may be taken, are as varied as human imagination and creativity, such as in the case of the blockchain.

Indeed, self-help when used without challenge does not depend on formal invocation of the legal system. Our traditional approach toward contract doctrine is to consider actions occurring within a legal framework, an approach which precludes some forms of self-help. Because the law indirectly influences the decision to rely on self-help in the first instance, self-help remedies are likely to be underutilized in a system that focuses in large part on judicial actions.[41]

The current mainstream view treats self-help as an "extralegal" action rather than as an element of our legal framework. Although self-help is nonjudicial, it is not extralegal and does not lie outside the "shadow of the law." Self-help is a party's immediate reaction to a perceived problem. That reaction, however, does not occur in a vacuum. In most cases, parties, including platforms and developers, have "some understanding" of legal rights. That understanding guides the determination of how and when to use self-help. Thus, self-help is no different from many examples of social ordering that takes place under the "shadow of the law," influenced by but not explicitly invoking the legal regime. Thus, self-help remedies incorporated into smart contracts are within the scope of contract law analysis and not outside of it.[42]

[38] See J. H. Hsiao, "'Smart Contract' on the Blockchain – Paradigm Shift in Contract Law" (2017) 14 *US-China Law Review* 685–94.

[39] A regulatory sandbox is a framework set up by a financial sector regulator to allow small-scale, live testing of innovations by private firms in a controlled environment (operating under a special exemption, allowance, or other limited, time-bound exception) under the regulator's supervision. Therefore, in the case of smart contracts on blockchains, regulator's supervision is limited or absent. D. A. Zetzsche, R. P. Buckley, D. W. Arner, J-N. Barberis, "Regulating a Revolution: From Regulatory Sandboxes to Smart Regulation" (2017) 23 *Fordham Journal of Corporate and Financial Law* 31–103.

[40] M. P. Gergen, "Theory of Self-Help Remedies in Contract" (2009) 89 *B.U. L. Rev.* 1397–449.

[41] C.R. Taylor, "Self-Help in Contract Law: An Exploration and Proposal" (1988) 33 *Wake Forest Law Review* 839–907.

[42] Savelyev, above note 15.

A first type of self-help relevant for smart contract consists of the threat that one party will respond to its counterparty's failure to adjust by reducing or terminating future dealings. This strategy incentivizes parties to agree to exceptions to the immutability of smart contracts. Secondly, the morality of platforms and developers, their reliance on gossiping and reputation, rather than their calculations of individual gain, encourages proper behavior in approaching smart contracts. In other words, they act as members of the blockchain ecosystem and do not behave opportunistically even when it is in their economic interest to do so (even when they are not under threat of punishment or retaliation). Clearly, the Dao case that is described below in Section 7.5.5 has shown how platforms and their members are capable, in certain circumstances, of subordinating their own welfare to the community that serves them.

A third type of self-help is normative or dispositional informal sanctions, which can operate at the level of social groups rather than among individuals. In compact and homogenous communities, like blockchains, the community as a whole can sanction the breach of one member's obligation to another by ostracizing the malefactor, cutting off not just business ties but removing all the benefits of belonging to the group. The next section discusses some interesting examples of these types of mechanisms by distinguishing self-help remedies related to codifying, security, privacy, and performance, as well as blockchain-based self-help remedies.

7.4 Self-Help: Codifying, Security, Privacy, and Performance

Platforms and developers already rely on self-help based on blockchain technology.[43] Legal scholars have struggled in mapping and understanding approaches that are technical and not legal in a strict sense. Legal scholars emphasize the human side of transactions conducted via blockchain platforms, including the loss of a blockchain private key, receipt of a defective product, the ability to show ownership, to invoke state consumer protection rights, or the need to verify title to land before entering a transaction on the blockchain. Moreover, smart contracts cannot access outside information unless it is written into the blockchain. Werbach and Cornell state that:

> There is a Frankenstein dimension to a smart contract: An instrument that fuses something innately human (entering into and enforcing agreements) with something mechanical, derived from scientific experiments. Science fiction authors since Mary Shelley have warned of the consequences of such cyborgs. Perhaps the benefits of smart contracts will exceed the costs. Perhaps the benefits can be magnified or the costs minimized. We should, nonetheless, carefully assess the both sides of the ledger.[44]

For our purposes, we note that, contrary to the claims of promoters of blockchain technologies, the rule of the code cannot replace law and automation will not solve every risk with respect to smart contracts. Thus, contrary to the myth of code as law, the design of legally approved self-help remedies is inevitable.

[43] H. Eenmaa-Dimitrieva and J. Schmidt-Kessen, "Regulation through code as a safeguard for implementing smart contracts in no-trust environments" (2017) EUI Working Paper LAW 2017/13, 1–31.

[44] Werbach and Cornell, note 11 at 364.

7.4.1 *Codifying*

There are four issues that face developers in writing smart contracts, namely, the difficulty of writing correct contracts, legal training for coders, the inability to modify or terminate contracts, the lack of support to identify underoptimized contracts, and the complexity of programming languages.

7.4.1.1 Writing Correct Smart Contracts

The first issue is the difficulty of writing correct smart contracts. Correctness of smart contracts in this context means contracts that function as intended by their developers. The reason why it is important to have correct smart contracts is because they control valuable currency units. Thus, if smart contracts are not executed as intended, some of their currency units will disappear.

In an attempt to tackle this issue, a number of solutions are suggested. One solution is to semi-automate the creation of smart contracts to ease the process of writing them (translation of human-readable contract representations to smart contract rules).[45] By using smart contracts, the parties aim at changing the paradigm of contract practice to embrace automation where no breach of contract is admitted instead of the traditional ex post remedial perspective of breach. In this way, the parties deliberately try to preclude ex-post corrections by relying on blockchain technology. At the outset the use of smart contracts was characterized by high hopes in coding and technology, resulting in a failure to determine and analyze potential problems. Thus, incorporation of self-help into smart contracts was not properly vetted, which explains their underutilization in existing contracts.

Indeed, the most compelling reason for the use of smart contracts is their guarantee of performance (impossibility of breach). This is based upon the assumption that the electronic format and Boolean logic remove the need for nuance and interpretation by eradicating the ambiguity found in traditional contracts. This ignores the fact that in the real world it is both operationally and socially beneficial to leave some contract terms underspecified; vagueness preserves operational flexibility for parties to deal with unexpected circumstances and sets the stage for social stability in an ongoing relationship as advanced by relational contract theory.[46] Moreover, it is important to note that not all clauses in contracts are susceptible to automation and self-execution. Even where a clause might technically be capable of being automated, it might not always be desirable to automate it.

Another solution is to provide developers with guidelines to aid them in writing correct contracts. Smart contracts, for example, can be built from contract templates designed with input from legal experts. Some authors propose the use of smart contract templates, based on the idea of Ricardian contracts that is a digitally signed triple ⟨P, M, C⟩, where P is the legal prose (legal, business, and regulatory semantics) from which denotational

[45] C. K. Frantz and M. Nowostawski, "From Institutions to Code: Towards Automated Generation of Smart Contracts" *2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS*W)*, 210–15.

[46] I. MacNeil and D. Campbell, *The Relational Theory of Contract: Selected Works of Ian MacNeil* (Sweet & Maxwell 2001).

semantics are captured and represented; M is a map (key-value pairs) of parameters used in P and C, and C is the platform-specific code that expresses operational semantics.[47]

7.4.1.2 Training for Coders and Legal Engineering

After smart contracts are created, but before being placed on the blockchain, best practice would use lawyers in reviewing the contract terms to check their compliance with the law. Legal code audits could be implemented analogous to the security audits widely used by firms engaged in software development. This approach consists of a sort of legal engineering of smart contracts.[48]

Some experts have released online materials (such as tutorials) to help developers write correct smart contracts. The final solution offered here is the adoption of formal verification techniques to detect unintended behaviors or consequences of smart contracts.[49] This can help developers recognize those behaviors before posting their contracts to the blockchain. Some scholars have utilized formal methods to analyze and verify the correctness of smart contracts,[50] while others have combined formal methods with game theory techniques to validate smart contracts.[51]

7.4.1.3 Modifying Smart Contracts

The second issue to be addressed is the inability to modify or terminate smart contracts due to their immutability. In the real world, modification is a form of self-help – in that it permits parties to respond to problems arising in executory or partially executed contracts in order to preserve contracts and help avoid the need for official intervention that might otherwise be necessary. In particular, the law recognizes certain excuses that will absolve a party from performance or require some sort of modification.[52] In civil law, both impossibility and economic impracticability are bases for excuse. This poses a problem for the smart contract when the coders have not included in the coded text an ability to manage impossibility and impracticability.

In an attempt to tackle this issue, some scholars have presented a set of standards to allow smart contracts to be changed or terminated along the lines of traditional contract

[47] C. D. Clack, V. A. Bakshi, and L. Braine, "Smart Contract Templates: Essential Requirements and Design Options," arXiv:1612.04496 (December 16, 2016), https://arxiv.org/abs/1612.04496 (last accessed 20 November 2018).

[48] K. D. Werbach, "Trust, But Verify: Why the Blockchain Needs the Law" (2017), https://ssrn.com/abstract=2844409 (last accessed July 20, 2018).

[49] K. Delmolino, M. Arnett, A. Kosba, A. Miller, and E. Shi, "Step by Step towards Creating a Safe Smart Contract: Lessons and Insights from a Cryptocurrency Lab" in *International Conference on Financial Cryptography and Data Security* (Springer, 2016), 79–94.

[50] K. Bhargavan, A. Delignat-Lavaud, C. Fournet, A. Gollamudi, G. Gonthier, N. Kobeissi, N. Kulatova, A. Rastogi, T. Sibut-Pinote, N. Swamy, et al., "Formal Verification of Smart Contracts: Short Paper" in *Proceedings of the 2016 ACM Workshop on Programming Languages and Analysis for Security* (ACM, 2016) 91–96.

[51] G. Bigi, A. Bracciali, G. Meacci, and E. Tuosto, "Validation of Decentralised Smart Contracts through Game Theory and Formal Methods" in *Programming Languages with Applications to Biology and Security* (Springer, 2015) 142–61.

[52] In the common law, these circumstances are described primarily by the doctrines of duress, unconscionability, mistake, misrepresentation, frustration, and discharge for breach.

law.[53] Interestingly, such standards are taken from legal contracts and then defined to fit the context of smart contracts. For example, termination, rescission, modification, and reformation terms have been coded for smart contracts and tested on Ethereum. Another possibility, assuming the parties to a given smart contract are known (depending on the type of blockchain), the parties can create a new transaction to reverse undesirable outcomes of the previously coded and executed transaction in order to implement *post hoc* corrections of mistakes, bugs or any other problem that had occurred

7.4.1.4 Avoiding Underoptimized Smart Contracts

The third issue relates to the lack of support in identifying under-optimized smart contracts. To run a smart contract, each computational or storage operation in the contract costs money. An underoptimized smart contract is a contract that contains unnecessary or expensive operations. Such operations result in a high cost at the user's side. In an attempt to tackle this issue, scholars identified seven programming patterns (unnecessary and expensive operations in a loop) in smart contracts, which lead to additional costs. They have also proposed ways to enhance the optimization of such patterns to reduce the overall cost of executing smart contracts. For example, a tool was developed to detect contracts that suffer from those patterns. The tool was applied to current Ethereum smart contracts and found most of them suffer from such patterns.[54]

7.4.1.5 Complexity of Programming Languages

The last issue to be addressed is the complexity of smart contract programming languages.[55] Current smart contracts are based on procedural languages, such as Solidity. In a procedural language, the code is executed as a sequence of steps. Thus, programmers specify what should be done and how to do it. This makes the task of writing smart contracts in those languages cumbersome and error prone. For example, Pettersson and Edström attempted to prevent three kinds of common errors made by developers: failure to account for unexpected states, failure to use cryptography, and overflowing the Ethereum Virtual Machine's (EVM's) stack.[56] They proposed the use of a functional programming language called Idris to help mitigate these risks. Based on this scheme, they developed a code generator that transforms code produced by an Idris compiler to Serpent code (an Ethereum scripting language), which can be subsequently compiled into EVM bytecode. This process does not, however, solve the translation problem. In addition, in order to address this issue, others have proposed the utilization of logic-based languages instead of procedural languages. In logic-based languages, programmers do not necessarily have to specify the sequence of steps in creating a contract. This will ease the

[53] B. Marino and A. Juels, "Setting Standards for Altering and Undoing Smart Contracts" in *International Symposium on Rules and Rule Markup Languages for the Semantic Web* (Cham, Switzerland, Springer International Publishing, 2016), 151–66.

[54] Clack, et al., *supra* note 47.

[55] F. Idelberger, G. Governatori, R. Riveret, and G. Sartor, "Evaluation of Logic-Based Smart Contracts for Blockchain Systems" in *International Symposium on Rules and Rule Markup Languages for the Semantic Web* (Cham, Switzerland, Springer International Publishing, 2016), 167–83.

[56] J. Pettersson and R. Edström, "Safer Smart Contracts through Type-Driven Development" PhD thesis, Master's thesis, Dept. of CS&E, Chalmers University of Technology & University of Gothenburg, Sweden (2015).

complexity of writing smart contracts. However, algorithms for logic-based languages are expensive and may be inefficient.[57]

From a different perspective, Frantz and Nowostawski considered the issue of authoring smart contracts from the subject-matter expert's perspective by proposing a semi-automated method for the translation of human readable contracts to smart contracts on Ethereum. The authors developed a domain-specific language for contract modeling, where rules are expressed in plain English and then translated into the Solidity vocabulary.[58] However, this solution is anchored on Ethereum, and it is not clear how useable or adaptable it is on other platforms. In addition, it does not incorporate the semantics of the legal and business language a lawyer would use to draft denotational semantics.

7.4.2 *Security Issues*

Security and privacy measures protecting blockchain transaction are central in protecting the parties' rights and obligations. Marino and Juels note that "When promises are embedded in technology, one (perhaps the only) way to breach them is to disrupt that technology."[59] Most smart contracts include security measures aimed at deterring this type of breach. This presents a special challenge for programmers that in writing a smart contract, correctness matters a great deal since the consequences of bad code writing can be dire. "Bugs" are inherent in computer code so the effectiveness of smart contracts is dependent on the careful writing of code, since errors cannot be fixed after the fact.[60]

A review of the literature shows the existence of six security issues, namely, transaction-ordering dependency, timestamp dependency, mishandled exception, criminal activities, re-entrancy, and untrustworthy data feeds. In addition to these issues, several vulnerabilities have been found in Ethereum smart contracts.[61]

The first issue to be addressed is transaction-ordering dependency. This problem occurs when two dependent transactions that invoke the same contract are included in one block. The order of executing transactions relies on the miner. However, an adversary can successfully launch an attack if those transactions are not executed in the right order. For example, assume there is a puzzle contract that incentivizes the user who solves the puzzle. A malicious owner is monitoring the solutions provided by the users. Once a user submits a correct solution to the puzzle (Tu), the malicious owner immediately sends a transaction (To) to update the contract's reward (such as reducing the reward). Those two transactions (To and Tu) might be included in the same block by chance. If the miner executed To before Tu, the user would get a lower reward and the malicious owner would succeed in his attack. To prevent this type of manipulation, Natoli and Gramoli suggest the use of Ethereum-based functions to enforce the order of transactions.[62]

[57] Idelberger, et al., *supra* note 55, at 167.
[58] Frantz and Nowostawski, *supra* note 45, at 210–15.
[59] Marino and Juels, *supra* note 53, at 152.
[60] N. Szabo, "Formalizing and Securing Relationships on Public Networks" (1997) 2(9) *First Monday*.
[61] N. Atzei, M. Bartoletti, and T. Cimoli, "A Survey of Attacks on Ethereum Smart Contracts (sok)" in *International Conference on Principles of Security and Trust* (Springer, 2017), 164–86.
[62] C. Natoli and V. Gramoli, "The Blockchain Anomaly," in *15th International Symposium on Network Computing and Applications (NCA)*, 310–17, IEEE, 2016.

Another suggested approach is using a guard condition such that a contract code either returns the expected output or fails.[63]

The second issue is timestamp dependency. This problem occurs when a contract uses the block timestamp as a condition to trigger and execute transactions (such as sending money). The block timestamp is usually set at the current local time by the miner who generated the block. The problem with the timestamp is that a dishonest miner could vary its value by about 15 minutes from the current time, while the block is still accepted by the blockchain system. As the timestamp of a block is not guaranteed to be accurate, contracts that rely on timestamp value are vulnerable to threats by dishonest miners. To avoid this problem, the block builder could use the block number as a random seed for contracts instead of using the block timestamp. This prevents manipulation because the value of the block number is fixed (miners cannot vary the block number value).[64]

The third issue is mishandled exception vulnerability. This problem occurs when a contract (caller) calls another contract (callee) without checking the value returned by the callee. In calling another contract, an exception (such as run out of gas) is sometimes requested by the callee. This exception, however, might or might not be reported to the caller depending on the construction of the call function. Not reporting an exception might lead to threats as in the KingOfTheEther (KoET) contract. In KoET, an adversary might send a transaction that results in an exception in order to buy the throne from the current king for free. With respect to this issue, some have highlighted the importance of checking the value returned by the callee. In the KoET example, the code can be improved to not release the throne till the payment from the adversary is completed successfully without any exception.[65]

The fourth issue is reentrancy vulnerability.[66] This problem occurs when an attacker utilizes a recursive call function to conduct multiple repetitive withdrawals, while their balances are only deduced once.

The fifth issue is the commission of criminal activities on the blockchain. An author highlighted the feasibility of constructing three different types of criminal activities in smart contract systems, namely sale of secret documents, theft of private keys and calling-card crimes, and a broad class of physical-world crimes.[67] These crimes can be implemented efficiently in the Ethereum blockchain by utilizing cryptographic techniques. There currently are no proposed solutions to the above two issues.

The last issue is the lack of trustworthy data feeds: oracles. An oracle is a party (or a technical source such as a database, or a person) who plays the role of "source of the truth" for a smart contract. The other parties of the smart contract trust that the oracle will provide the right information for the execution of the contract, because they cannot verify the information on chain. Putting it differently, using an oracle means receiving data from outside of a blockchain. An oracle provides a connection between real-world events

[63] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making Smart Contracts Smarter" in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16 (ACM, 2016), 254–69.

[64] Ibid.

[65] *Supra* note, at 12.

[66] Ibid.

[67] A. Juels, A. Kosba, and E. Shi, "The Ring of Gyges: Investigating the Future of Criminal Smart Contracts," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, (ACM, 2016), 283–95.

and a blockchain. In particular, some smart contracts require information (data feeds) from outside the blockchain.[68] The problem is that there is no guarantee that the information provided by an external source is trustworthy.

Zhang and other colleagues have proposed to develop a trusted third party between external sources and smart contracts to provide authenticated data feed for smart contracts.[69]

7.4.3 *Privacy Issues*

The two main privacy issues posed by smart contract are the lack of transactional privacy and the lack of data feeds privacy. An example of a lack of transactional privacy[70] in blockchain systems is when all transactions and users' balances are publicly available to be viewed. This lack of privacy could limit the adoption of smart contracts as many people consider financial transactions (such as stock trading) as confidential information.[71] Watanabe and colleagues have proposed to encrypt smart contracts before deploying them to the blockchain. Only participants who are involved in a contract can access the contract's content by using their decryption keys.[72]

A lack of data feed privacy occurs when a contract that requires a data feed sends a request to the party that provides those feeds. However, this request is exposed to the public as anyone in the blockchain can see it. Kosba and colleagues have proposed extending their Town Crier (TC) tool to support private requests.[73] Thus, a contract can encrypt the request using the TC's public key. Upon receiving the encrypted request, the TC can decrypt it using its private key. This guarantees that the content of the request is kept secret from other users and contracts in the blockchain.

7.4.4 *Performance Issues*

One possible performance problem relates to the sequential execution of smart contracts. In blockchain systems, smart contracts are executed sequentially (one contract at a time). However, this would affect the performance of the blockchain systems negatively as the number of contracts that can be executed per second is limited. With the growing number of smart contracts in the future, the blockchain systems will not be able to accommodate such a large scale of transactions. Zhang and colleagues suggest executing smart contracts in parallel as long as they are independent ("do not update the same

[68] J. I-H Hsiao, "'Smart' Contract on the Blockchain – Paradigm Shift for Contract Law?" (2017) *USCLR (US-China Law Review)* 685–94, 691.

[69] F. Zhang, E. Cecchetti, K. Croman, A. Juels, and E. Shi, "Town Crier: An Authenticated Data feed for Smart Contracts," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16 (ACM, 2016), 270–82.

[70] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts" in *2016 IEEE Symposium on Security and Privacy (SP)* (IEEE, 2016), 839–58.

[71] Ibid.

[72] H. Watanabe, S. Fujimura, A. Nakadaira, Y. Miyazaki, A. Akutsu, and J. J. Kishigami, "Blockchain Contract: A Complete Consensus Using Blockchain," in *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)* (IEEE, 2015), 577–78.

[73] Zhang, et al., *supra* note 69.

variables").[74] By doing so, the performance of blockchain systems would be improved as more contracts could be executed per second. However, parallel execution of contracts faces a challenge in how to execute contracts that depend on each other at the same time. It is, therefore, essential to conduct research on identifying and tackling performance issues to ensure the ability of blockchains to handle larger scales of transactions.

7.5 Blockchain-Based Self-Help

Platforms and developers also represent interesting cases of cooperative self-help based on their ecosystems. Economists frequently refer to historical institutions in discussions of the institutional determinants of economic development and the economic role of social capital. Particular attention in recent years has been lavished on the Maghribi traders of the eleventh-century Mediterranean, following the work of Avner Greif.[75] In the absence of formal legal contract enforcement, the Maghribis developed an informal contract-enforcement mechanism based on multilateral relationships within a closely-knit "coalition." This mechanism exemplifies both the feasibility of private alternatives to the public legal system as a basis for economic transactions and the key role of social capital and informal institutions in developing economies. Furthermore, the Maghribis held "collectivist" Judaeo-Muslim beliefs and norms, which led them to develop different institutions from their "individualistic" Christian counterparts, and this is held to exemplify the pivotal role of cultural differences in explaining institutional and economic development. Economists have drawn far-reaching lessons from Greif's work.

Can understanding the Maghribis-like system help us to understand smart contact governance and management? For example, the use of self-help remedies for smart contracts is based on multilateral relationships within blockchain coalitions. Platforms and developers share a "collectivist" view of their roles because of their commitment to the world of the Code. Cox, Arnold and Villamayor-Tomás argue that the blockchain is an example of Commons 3.0 in that it provides a technical solution (cryptographic consensus) to the problem of cooperation in joint or group production at scale while still maintaining the benefits of commons-type (polycentric) institutional governance. A blockchain is a trustless commons in which effective rules are embedded in constitutional smart contracts that are cryptographically secure and crypto-economically implemented.[76]

This view of the blockchain as a collective or closely knit coalition is curious with respect to trustless smart contracts where the parties do not know each other, nor identify themselves in their communities. To be precise, blockchain networks can be private with restricted membership (permissioned blockchains), or they can be accessible to any person in the world (unpermissioned blockchains). There are also "consortium blockchains," where the process of validating transactions is controlled by a fixed set of nodes.

[74] M. Vukolić, "Rethinking Permissioned Blockchains" in *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, BCC '17 (ACM, 2017), 3–7.

[75] A. Grief, "Reputation and Coalitions in Medieval Trade: Evidence on the Maghribi Traders" (1989) 49(4) *JEH (Journal of Economic History)* 857–82. The author referred to "informal contract-enforcement mechanism based on multilateral relationships within a closely-knit 'coalition'." See also A. Grief, *Institutions and the Path to the Modern Economy: Lessons from Medieval Trade* (Cambridge: Cambridge University Press 2006). For additional examples, see notes 20 and 21.

[76] M. Cox, G. Arnold, and S. Villamayor-Tomás "A Review of Design Principles for Community-Based Natural Resource Management" (2010) 15 *ES (Ecology and Society)* 38.

In addition, the parties are not identifiable in the case of crypt transactions. In these cases, they are anonymous. .

In our case, blockchains minimize the amount of trust required from any single actor in the system. They do this by distributing trust among different actors in the system via an economic game that incentivizes actors to cooperate with the rules defined by the protocol. In other words, the case considered here confirms that the social dimension of contracts matters in a technological age.[77]

This comes close to resembling the 1990s vision of cyberlaw experts that digital communities would define their own rules, free from state involvement. In fact, Wright and De Filippi draw a direct connection between the blockchain's "Lex Cryptographica" and the "Lex Informatica" of software code. They argue that the blockchain "could make it easier for citizens to create custom legal systems, where people are free to choose and to implement their own rules within their own techno-legal frameworks."[78]

The next sections review some examples of community self-help, such as mining, deposits, gossiping and, as last resort, social repudiation.

7.5.1 *Mining*

At the heart of bitcoin there is a set of protocols often called Nakamoto Consensus.[79]

Consensus means that participants in a network have confidence that what they see on their ledgers is accurate and consistent. Without a robust means of ensuring consensus, any bitcoin participant could, for example, spend the same bitcoins multiple times (known as the double-spend problem), or claim it had more currency than it really did. The trouble with most approaches to consensus on digital systems is that it is easy to create a nearly infinite number of fake network nodes (known as a Sybil attack). Even if most real users are honest, an attacker can create enough nodes to dominate the network and impose its own false consensus on the system (this challenge is well known in cryptography and is referred to as the "byzantine generals problem").

Until bitcoin, it had no scalable solutions. Nakamoto's answer cleverly combined cryptographic techniques with insight from game theory. The basic approach is called a byzantine fault tolerant protocol. Instead of trusting an individual actor, you trust a network of actors, who express themselves through voting. In Nakamoto's version, these actors engage in a process known as mining, through which they validate and establish consensus over chunks of bitcoin transactions. Every full node sees every transaction, and there is only one consensus ledger mirrored across every machine on

[77] K. Levy, Book-Smart, "Not Street-Smart: Blockchain-Based Smart Contracts and the Social Workings of Law" (2017) 3 *ESTS (Engaging Science, Technology, and Society)* 1–15.

[78] Wright and De Filippi, *supra* note 28, at 44–47. See generally J. Reidenberg, "Lex Informatica: The Formulation of Information Policy Rules through Technology" (1997) 76 *TLR (Texas Law Review)* 553 (developing the concept of Lex Informatica by analogy to the historical Lex Mercatoria).

[79] S. Nakamoto, "Bitcoin: A Peer-To-Peer Electronic Cash System" (2008) at https://bitcoin.org/bitcoin.pdf (Digital currencies such as Bitcoin that are based on encryption techniques are often referred to as cryptocurrencies.); Nick Szabo, "The Dawn of Trustworthy Computing," *Unenumerated Blog* (December 11, 2014), at http://unenumerated.blogspot.com/2014/12/the-dawn-of-trustworthycomputing.html.

the network. Even if some of the miners are untrustworthy, the system holds so long as the majority is honest.

The major limitation of such a protocol is the possibility of Sybil attacks[80]: if it is easy and rewarding to be untrustworthy, some participants will be. Hence, the second cryptographic technique in bitcoin is proof of work. Proof of work makes voting costly. Bitcoin's proof of work system requires miners to solve cryptographic puzzles involving one-way functions known as hashes. Votes require massive and growing computing power, which is sufficiently expensive to deter Sybil attacks. The benefits of cheating are less than the costs. Nakamoto Consensus affirms the integrity both sides of each individual transaction and of the ledger as a whole. It does so by aggregating together transactions into blocks. The proof of work system is tuned dynamically to generate a valid solution to the hashing puzzle for a block roughly once every ten minutes. Each block thus validated is cryptographically signed with the hash of the prior block, creating an immutable chain of sequential blocks. The longest chain of blocks represents the consensus state of the system.

Bitcoin is also designed to be censorship and tamper resistant. There is no central control point or network where a government could manipulate or block. Once a transaction is recorded, it cannot be changed. The final key piece of Nakamoto Consensus is the game-theoretic or psychological dimension: why will miners bother to mine? Proof of work is expensive, literally; it requires expensive computing hardware and even more use of electricity. Miners would not be incentivized sufficiently out of altruism. Nakamoto's solution was elegant. The miner who successfully validates a block receives a reward in a valuable currency.[81]

7.5.2 *Deposits or Escrow Services*

Deposits can be used to establish a level of trust between new and existing agents. This approach is akin to using agents to participate in an incentive-based protocol to provide correct (verifiable) computations.[82] This approach is adapted for a privacy-preserving computation protocol and for verifiable computations on Ethereum.[83] Each agent that wants to participate in these protocols is not trusted by default. This distrust results from the openness of the blockchain platform (no central authority gives access) and the direct accessibility of cryptocurrency. Assuming a rational agent, there is a motivation to break privacy or allow incorrect solutions if this optimizes the agent's own utility function. To deter this type of behavior, new agents have to deposit a certain cryptocurrency value for participation. In the aforementioned protocols, this deposit is returned when an agent decides to stop participating. However, dishonest or corrupt agents can be penalized by destroying their deposit or distributing it to honest agents.

[80] John R. Douceur, "The Sybil Attack," *Revised Papers from the First International Workshop on Peer-to-Peer Systems* 251 (2002), http://nakamotoinstitute.org/static/docs/the-sybil-attack.pdf

[81] "Bitcoin: The Magic of Mining" *Economist* (January 10, 2015), at 58, www.economist.com/node/2163812.

[82] R. K. and I. Bentov, "How to Use Bitcoin to Incentivize Correct Computations," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* (New York, USA: ACM Press, 2014), 30–41.

[83] G. Zyskind, O. Nathan, and A. S. Pentland, "Decentralizing Privacy: Using Blockchain to Protect Personal Data" *IEEE Security and Privacy Workshops* (May 2015) 180–84.

7.5.3 *Gossiping*

Gossiping can be used to communicate experiences with other agents in a P2P fashion and thereby establish trust or reputation scores. Indeed, when most people conduct business over the Internet, they are less interested in the legal consequences of their transactions than the interconnectedness that results from the exchange. This can be seen in the way that people rate their experiences on eBay, Uber, and TripAdvisor. Users of these service providers rate their experience with the vendor based on the quality and timeliness of the delivery of the service or product. These ratings create a reputation for the service provider and build relationships of trust in the network or community. Even though the nature of the marketplace means that participants will very likely never meet, their interactions give rise to exchanges where the parties to the transaction are relying on each other's status established through these conversations, rather than the strict legal rights expressed in terms and conditions

Clearly, it is an essential ingredient of any e-commerce reputation system to manage the integrity of feedback and to ensure that only genuine users provide it (and not, for example, by fake identities created by the person or persons who want to synthesize an improvement in their reputation). Anyone can browse eBay, but in order to join in the business of this community, buyers and sellers must first be registered with the platform. Exchanges are only possible when users are signed into the system with their unique identities. Even though users regularly use pseudonyms, those names are linked back to genuine pre-validated email addresses and credit cards. This system ensures that real people are the puppet masters of their avatars and that they must behave according to the rules of the marketplace. Under the rating system, the more stars received by a member, the more reliable and trustworthy they are, increasing their popularity with other members, and thereby resulting in significant economic advantages for those users.

Some cryptocurrency exchanges have designed trading platforms that provide information about the number of trades undertaken by each trader and the ratings provided by other users. The feedback is represented by color-coded dots and percentage rankings to reflect each trader's level of recent trading activity and the satisfaction of their customers. However, these apps are not built into the blockchain network and so suffer from a lack of decentralization since they depend upon the trustworthiness of those providing the feedback.[84]

Thus, the solutions available to prevent feedback abuse are generally reliable but centralized under the control of a trusted agent. However, by building a decentralized and distributed feedback management system on top of the blockchain, it is possible to provide reliable reputation ratings. A key feature of this system would be to attach more weight to the feedback of an established and trusted user on the network than new identities.[85]

[84] A. Schaub, R. Bazin, O. Hasan, and L. Brunie, "A Trustless Privacy-Preserving Reputation System," in J. H. Hoepman and S. Katzenbeisser (eds.), *ICT Systems Security and Privacy Protection* (SEC 2016); *IFIP Advances in Information and Communication Technology*, Vol. 471 (Cham, Switzerland, Springer International Publishing, 2016). See also D. Carboni, "Feedback based Reputation on top of the Bitcoin Blockchain," arXiv:1502.01504 (February 5, 2015) (last accessed November 20, 2018), https://arxiv.org/abs/1502.01504.

[85] Examples of proposed solutions are given in: A. Burak Can, B. Bhargava. "SORT: A Self-Organizing Trust Model for Peer-to-Peer Systems," *IEEE Transactions on Dependable and Secure Computing* 10.1 (January

7.5.4 *Reputation*

Reputation is invaluable in small communities with repeated transactions (like private blockchain environments). Parties not conforming to communal norms or that breach contracts suffer negative reputation consequences resulting in exclusion from future interactions. Thus, it is in parties' self-interest to build a reputation of reliability, honesty, and fair dealing.

However, integrity and reputation are of little importance in blockchain transactions. Misbehaving parties can simply erase their history by creating a new pseudonym.[86] In contrast, most people that conduct business over the Internet are more interested in interconnectedness. As discussed in the previous section, the rating of services create a reputation for the service provider and build relationships of trust in the network or community.[87]

In fact, participants use a variety of tools to promote their "reputations" in a decentralized blockchain. "Uprightly" is a platform that allows participants to enhance their reputations. It is independent of any other platform.[88] This way, users become market participants in order to safeguard their interests in the process. Further, Uprightly requires no third-party intervention; it is censorship-resistant; and decentralized.

7.5.5 *Social Repudiation*

The infamous Dao attack is the reason why Ethereum created a hard fork that split it into two platforms.[89] The Dao is the abbreviation for "The Decentralized Autonomous Organization" and is represented as a complex smart contract, which was considered to have revolutionized Ethereum forever. The Dao was not owned by anyone, and it previously worked as follows: a group of people developed the smart contract(s) that run the organization; followed by an initial funding period, where Dao Tokens were sold (namely: ICO11), which represent ownership (the right to vote). Once the funding was over, developers made smart contract proposals, and the users that owned tokens voted on approving and funding the proposals. Token holders had the option to opt-out from the organization and get back the money they invested, if for example, they did not endorse a smart contract that was getting funded. This was done through a split function in which the user would automatically opt out from the organization and get their money back. The user could also choose to create a child Dao and anyone else that did not agree on

2013), 14–27. H. Zhao and X. Li. "VectorTrust: Trust Vector Aggregation Scheme for Trust Management in Peer-to-Peer Networks" (2013) 64 *JS (The Journal of Supercomputing)* 805–29.

[86] N. Nisan, T. Roughgarden, É. Tardos, V.V. Vazirani and C. H. Papadimitriou, *Algorithmic Game Theory* (Cambridge: Cambridge University Press, 2008) at 682. Being able to cheaply create a new pseudonym in order to dodge a bad reputation is called whitewashing. One response is to distrust newcomers since their identities may mask a bad reputation or to require that pseudonyms be linked to a real person or business.

[87] W.L. Felstiner, R. L. Abel, and A. Sarat, "The Emergence and Transformation of Disputes: Naming, Blaming, Claiming ..." (1980) 15 *LSR (Law & Society Review)* 631.

[88] "Uprightly" is discussed at https://bitcoinexchangeguide.com/uprightly-ico-upt-token/.

[89] The digital Decentralized Autonomous Organization ("DAO") had an objective to provide a new decentralized business model for organizing both commercial and nonprofit enterprises. It was instantiated on the Ethereum blockchain. The DAO raises funds and members write smart contracts to manage the investing of the funds. In the case mentioned here, a hacker managed to create a parallel DAO and stole 3.6 million units of Ether (i.e., a cryptocurrency).

funding (endorsing) a specific proposal, could join it. This design made it popular by offering flexibility, full control and complete transparency. However, this design allowed a hacker to exploit a re-entrancy vulnerability in the "split DAO" function. The attacker was able to steal one third of Dao's funds (3.6 million Ether), which was considered to be worth around $60 million at that time. The consequences of this attack had on Ethereum were unprecedented. The price of Ether dropped from $20 to $13 and the hack challenged the notion of immutability.

Interestingly, the incident ultimately led the Ethereum community to rescind (hard fork) the blockchain, by violating the underlying principle of distributed ledger technology.[90] The taking of Ether did not constitute a breach since the hacker followed Dao rules, but in a strategic and malicious manner. The ETH community's hard fork was a way to protect the agreement (crowd-sourced venture capital), but it did so via social repudiation (breach) of the "contract" itself. Thus, one may say that social repudiation provides an extreme, and inefficient, self-help mechanism to correct errors or manipulation of the blockchain. Clearly, the case also shows how platforms and their members establish guiding norms of behavior on the understanding that subordinating their own welfare to that of the community serves their own self-interests.

7.6 Conclusion

The chapter has explored the remedial perspective with respect to smart contracts. First of all, it noted the peculiar architecture of trust for smart contracts and its legal implications. In particular, it stresses the central role played by self-help remedies in blockchain environments, which is needed to keep the law in the background. Some different issues were identified with respect to self-help remedies, namely, coding, security, privacy and performance issues. Coding and security issues are the most commonly discussed issues. This is because smart contracts store valuable currency units and any security breach or coding error could result in the loss of money. The identified coding issues are the difficulty of writing correct codes, the inability to modify or terminate contracts, the lack of support to identify underoptimized contracts and the complexity of programming languages. The identified security issues are transaction-ordering dependency, time-stamp dependency, mishandled exception, reentrancy, untrustworthy data feeds, and criminal activities. The identified privacy issues are the lack of transactional privacy and the lack of data feeds privacy (oracles). The identified performance issue is the sequential execution of smart contracts. Although there are some proposed solutions to these problems, some of them are only abstract ideas lacking concrete evaluation. Presently, there is already a great deal of reliance on blockchain self-help, including the use of and forfeiture of deposits, gossiping, mining, and social repudiation. These remedies show that the social dimension of contracts remains relevant in the blockchain world.[91]

[90] A hard fork, or hardfork, as it relates to blockchain technology, is a radical change to the protocol that makes previously invalid blocks/transactions valid (or vice versa). This requires all nodes or users to upgrade to the latest version of the protocol software. Put differently, a hard fork is a permanent divergence from the previous version of the blockchain, and nodes running previous versions will no longer be accepted by the newest version, www.investopedia.com/terms/h/hard-fork.asp #ixzz5E3U9U2Pg (last accessed 10 November 2018).

[91] G. Sandstrom, "Who Would Live in a Blockchain Society? The Rise of Cryptographically Enabled Ledger Communities" (2017) 6 *SERRC (Social Epistemology Review and Reply Collective)* 27–41.

The key point made in the chapter is that the need for formal legal remedies is less frequent in smart contracts. It will be important to monitor the emergence of measures (proactive and reactive) aimed at preventing or (when necessary) respond to vitiating elements such as manipulation, errors in coding, breach, eavesdropping, and interference in smart contracts, as well as *ex contractu* or business responses, before resort is made to contract law and the legal system.

The continued implementation of self-help remedies may diminish the use of the formal legal system, but they cannot transplant the legal system or prevent the use of the legal system for parties to obtain contractual remedies. Smart contracts cannot avoid law as coded terms remain subject to judicial review, especially when highly scrutinized clauses (limitation of liability or limitation of remedy clauses) or unenforceable clauses (unconscionable terms, penalty clauses that are deemed to be excessive or avoidance of consumer protection law) are coded into the contracts. Smart contracts also remain subject to contract laws policing doctrines, such as mistake, duress or coercion, "surprising terms,"[92] misrepresentation or fraud, as well as implied terms, such as the duty of good faith and the duty of reasonable efforts in agency contracts.

[92] BGB §305c ("Provisions [that are] are so unusual that the contractual partner of the user could not be expected to have reckoned with them, do not form part of the contract").