Applying Smart Contracts in Online Dispute Resolutions on a Large Scale and its Regulatory Implications

Janet Hui Xue* Ralph Holz**

1.Introduction

This chapter analyzes the feasibility of using smart contract technology to handle online dispute resolution (ODR) on a large scale. Online Dispute Resolution (ODR) is 'referred to as the use of technology to carry out the dispute resolution process.'[1] ODR combines Alternative Dispute Resolution (ADR) and Information and Communication Technology;[2] it can be used for disputes arising from both online e-commerce transactions and offline transactions such as purchases. ODR becomes particularly relevant in the context of cross-border e-commerce. It can include the traditional legal process, although it does not usually rely on it: the traditional legal process often involves a court, judge, and possibly a jury to decide a dispute.[3] However, ODR has a scalability problem: it cannot easily satisfy the sheer

[1] See Cortés Pablo,The Law of Consumer Redress in an Evolving Digital Market: Upgrading from Alternative to Online Dispute Resolution, p.44.

[2] Consumer ADR systems differ significantly from traditional out-of court processes employed between commercial parties by relying on commercial arbitration and mediation processes. See Cortés Pablo,The Law of Consumer Redress in an Evolving Digital Market: Upgrading from Alternative to Online Dispute Resolution, p.3.

[3] ODR has often been considered an online form of ADR. It aims to provide tailored a solution to a conflict between several parties. According to the FTC, there are two types of dispute resolution: mediation and arbitration. The former one refers to a mediator acting as a neutral third party, facilitating dialogue to help the parties solve the problem. The final solution depends on the parties reaching an agreement. Arbitration is

number of possible requests from millions of consumers, given how strongly e-commerce is still growing and the high number of intermediaries that are needed to support the financial transactions that ultimately enable e-commerce. Such intermediaries include a range of financial institutions: banks, brokers/dealers, and also other institutions that interact with the end-users of a financial transaction; the term also includes infrastructure such as payment, clearing, and settlement systems.[4] Several ODR systems exist at international and regional levels, including eBay's ODR system[5] or the domain name dispute resolution used by ICANN[6]. However, no *globally* viable ODR model has emerged yet.[7] Recently, so-called smart contracts on blockchains have been suggested as a possible solution for 'self-enforcing Online Dispute Resolution'[8]. The claimed advantage is that smart contracts may be able to deal with disputes much faster and on very large scale.

However, smart contract technology is in its infancy, and such advanced use has not been sufficiently explored yet. It is not entirely clear yet what smart contract platforms will eventually exist and which features they will support. Expectations of the viability of smart contract technology may be overly optimistic; there is also a certain degree of confusion regarding possible features and limitations. Importantly, while many use cases have been proposed, very few have been analyzed from the perspective of legal compliance. As a result, Small and Medium Enterprises (SMEs) that wish to enter the financial industry, e.g. third-party payment suppliers, face much uncertainty when they consider blockchains and smart

---

associated with a stronger force than mediation, but is less formal than court. The parties may appear at hearings, present evidence, or call and question each other's witnesses. An arbitrator or panel can make a decision after the cases have been presented; a decision may be legally binding. See original explanation from FTC: Alternative Dispute Resolution, https://www.consumer.ftc.gov/articles/0162-alternative-dispute-resolution.

[4] David Mills, et.al. , Distributed ledger technology in payments, clearing, and settlement, Finance and Economics Discussion Series Divisions of Research & Statistics and Monetary Affairs Federal Reserve Board, Washington, D.C., https://www.federalreserve.gov/econresdata/feds/2016/files/2016095pap.pdf, p.4.

[5] eBay, Dispute Resolution Overview, https://pages.ebay.com/services/buyandsell/disputeres.html, (accessed 7 November 2017).

[6] ICANN, https://www.icann.org/resources/pages/dndr-2012-02-25-en, (accessed 7 November 2017).

[7] See Chapter I 'Online Dispute Resolution and Prevention: A Historical Overview' in Ethan Katsh and Orna Rabinovich-Einy, Digital Justice: Technology and the Internet of Disputes (Oxford University Press, 2017); Louis Del Duca, Colin Rule, and Zbynek Loebl Facilitating Expansion of Cross-Border ECommerce - Developing a Global Online Dispute Resolution System (Lessons Derived from Existing ODR Systems – Work of the United Nations Commission on International Trade Law), Penn State Journal of Law & International Affairs, https://elibrary.law.psu.edu/cgi/viewcontent.cgi?article=1004&context=jlia.

[8] Pietro Ortolani, Self-Enforcing Online Dispute Resolution: Lessons from Bitcoin, 36 Oxford Journal of Legal Studies, 595, 595–96, 598–602 (2016).

contracts for e-commerce. Even traditional regulatory authorities, such as central banks and governments, face some challenges.

This chapter identifies and illuminates the feasible regulatory space to help understand how smart contracts for ODR platforms can possibly be regulated and embedded within current law systems. The chapter first defines smart contracts and explains their relationship to blockchain. It presents their capabilities and limitations and how they are different from conventional, more centralized approaches, thereby highlighting a unique contribution that is of great help for future ODR platforms: the enforced transparency of the dispute process. Based on this technical analysis, the chapter proposes an approach how smart contracts can be useful in ODR platforms as they are used today, such as the EU ODR platform[9], by outlining the principles applying to arbitrators and definitions of access control, e.g. the need for access for state actors. The chapter presents possible approaches to apply smart contracts to standardize the procedures of lodging cases on large scale and handling disputes in an efficient manner. The proposals are based on lessons learnt in the last few years in the development of smart contracts on the Ethereum blockchain, currently the only major blockchain that supports smart contracts.[10] The chapter concludes with an analysis of implications for future regulation and legal compliance.

For the purpose of this chapter, we will not need to distinguish between public blockchains, which allow anyone to participate, and private blockchains, which require some form of permission by an authoritative entity (or group of entities) to participate. The possibilities and limitations of smart contracts are largely the same for both; the difference is mostly that the permission-based system used in private blockchain allows to assign specific roles to certain participants more easily. Hence, we limit ourselves to the more general case of public blockchain in the following discussion.

[9] European Commission, Online Dispute Resolution: Resolve your online consumer problem fairly and efficiently without going to court, https://ec.europa.eu/consumers/odr/main/?event=main.home2.show, (accessed 27 May 2018).
[10] Solidity: High-Level Language for Implementing Smart Contracts, http://solidity.readthedocs.io/, (accessed 27 May 2018).

2. Blockchain: distributed ledgers, consensus, and trust

Blockchain is a form of distributed ledger technology: blockchain participants maintain a shared, joint view of all transactions that have ever occurred. Classically, solving the question how two parties can trust in the validity of a transaction would require intermediaries such as banks. Public blockchain replaces trusted intermediaries with a collective of a large network of participants that run a so-called consensus protocol; anyone can participate. The consensus protocol gives strong assurance in the validity of a transaction that the network as a whole has accepted. A number of consensus protocols exist. In the case of the famous Nakamoto consensus, employed in Bitcoin and in a variant also in today's Ethereum[11], one can show that as long as a majority of the participants execute the consensus protocol faithfully, the entire network will eventually achieve a consistent view of all balances between any two parties. Building consensus requires each participant to group transactions into so-called blocks. Blocks have predecessors and successors, and hence they form a chain. Creating a block requires investing computational effort to solve a cryptographic problem. The first participant who successfully creates a new block and links it into the blockchain is rewarded with a certain amount of currency (Bitcoin or Ether, respectively, for Bitcoin and Ethereum). This is generally called "mining".

The cryptographic problem is defined such that participants of equal computational power have an equal probability of creating a new block. This construction ensures that the probability of fraud becomes extremely small and negligible - attempts to interfere with the consensus require the fraudulent participant to have more computational resources than most other participants combined.[12] As long as the part of the network that has a "computational majority" behaves faithfully to the protocol, the network will preserve a trustworthy record of all transactions ever made. The consensus protocol is implemented in software that every

[11] Ethereum intends to eventually switch over to a different form of consensus; at the time of writing, this has not been sufficiently specified to include it in discussions.

[12] The initial belief was that an absolute majority is required. Due to participants being able to collude, it is currently believed that the required computational power is on the order of 25% of the network's total computational power.

participant can run.[13] In this chapter, we always use the term participant as an entity participating in the network and consensus protocol.

3. Smart contracts: Nature, execution, and distinctive features

The notion of smart contracts is an addition to the blockchain paradigm. A smart contract is a computer program that is executed by participants in a blockchain.[14] This is the same principle that web services employ: one can send and receive data from a web service via an interface, and use it for further processing and display. With smart contracts, the difference is that every participant can create a smart contract and define a set of invocable routines (called "methods") for other participants to use and interact with the smart contract. A toy example would be a smart contract that implements and exposes the functionality of a calculator in its methods. Other participants in the network, and in principle also other smart contracts, can invoke these methods to obtain results, e.g. they might send numbers to a method called "sum", and receive the total sum as a result.

A smart contract is stored inside a transaction sent to the blockchain; hence once the network has achieved consensus, the smart contract becomes an immutable entry in the blockchain. All participants are required to execute a smart contract that is "invoked", i.e. interacted with. Interactions with a smart contract are again transactions.[15] This means that any interaction with a smart contract is also stored in the blockchain, and hence transparent to the entire network.[16] This is a necessary prerequisite for participants to maintain a consistent view of the ledger. Smart contracts are identified by (cryptographic) addresses, in the same way that senders and receivers in the network are identified by such addresses (and in general

[13] It is conceivable that some entities choose not to be part of the execution of the consensus protocol; in this case they become relying parties; they are not participants by our definition.
[14] The concept of smart contracts was first introduced in the mid-90s by Nick Szabo. Nevertheless, their implementation remained theoretical until blockchain development. For an overview see Nick Szabo, 'Formalizing and Securing Relationships on Public Networks' (1997) 2 (9) First Monday, http://journals.uic.edu/ojs/index.php/fm/article/view/548/469, accessed 31 May 2016.
[15] Smart contracts can also have so-called "internal transactions": these are executions of code that are completely determined by the invocation of a smart contract. As all participants must execute the invocation, these internal transactions are not stored in the blockchain; but they can always be inferred later by executing the smart contract invocation.
[16] The execution of smart contracts normally happens in a so-called Virtual Machine (VM). VMs emulate a computer system. They are very commonly in use, even in browsers, to provide a standardised execution environment for programs. Users are rarely even aware of their presence.

not by their real-world identity). Just like users, smart contracts may even have control over digital assets.

The hope that has often been expressed for smart contracts is that they can be used to define the terms, rules, and penalties of true, legal contracts, *and* also automatically execute and enforce the associated obligations, with specific events triggering transactions. This is often referred to as "self-enforcement". They can also help to simplify transactions where more than one party has to sign off in order for it to become valid. However, it is important to understand that smart contracts are still just computer programs, i.e. a collection of algorithms. This allows to utilize them as means for payment, clearing, and settlement processes, which define ways to transfer funds, clear and settle securities, commodities, as well as their derivatives.[17] All of these are easily described algorithmically. In principle, this means that the procedures of handling dispute cases can also be encoded in a smart contract. However, smart contracts cannot be used to, e.g., reason about the validity of a claim if this requires interpretation of law. Furthermore, smart contracts always need to be invoked by transactions, which in turn requires the existence of a blockchain participant that can create such a transaction. At the time of writing, no blockchain system allows to define external events that would automatically trigger the creation of invoking transactions - some participant must always be involved.

A distinctive feature of smart contracts is that invoking a method requires a form of payment for the execution of the method. In Ethereum, for example, the operations defined in a smart contract add up to a total amount of "gas" that is required - "gas" here being an internal unit that measures the cost of execution. The participant (or contract) invoking the method must offer to pay for the required gas. The invoking party offers a "gas price", i.e. a conversion rate from Ethereum's internal currency, Ether, to gas, and a total amount of Ether they are willing to pay for the invocation. A participant who chooses the transaction for inclusion in a block must execute the invoked method. The gas spent is awarded as Ether to the first

[17] David Mills, et.al. , Distributed ledger technology in payments, clearing, and settlement, Finance and Economics Discussion Series Divisions of Research & Statistics and Monetary Affairs Federal Reserve Board, Washington, D.C., https://www.federalreserve.gov/econresdata/feds/2016/files/2016095pap.pdf, (accessed 7 July, 2018).

participant who creates a block that includes the invoking transaction. Smart contracts can themselves send and receive currency as specified by their authors; the authors can also endow a smart contract with currency to use before deploying it onto the blockchain.

Importantly, smart contracts can “protect” their methods and endowments by specifying conditions for access. Smart contracts have owners who have special access privileges, e.g. to empty the current holding, or to shut down the smart contract (which makes further invocations impossible). Ownership is implemented using the typical asymmetric cryptography of blockchain; the owner’s private key is needed to interact with the protected methods of a smart contract. In this sense, smart contracts implement the same form of access control that is common to most of computer security. It naturally means that all the known problems of access control - participants losing keys, keys being stolen, etc. - also apply to smart contracts. In classic computer security, access control is generally accompanied by operational procedures, like requiring only thoroughly vetted personnel within an organisation being allowed to make changes, or the four-eye principle to initiate transactions. Smart contracts can reproduce all these features; but by default, there are no operational procedures predefined. Together, this opens a classic regulatory space: in theory, smart contracts could be mandated to feature default access for certain (state) actors. Furthermore, regulation may impose on all owners of smart contracts an obligation to keep their keys for access control in a safe place.

The regulatory space extends beyond the authors of smart contracts. All smart contracts are run on the software provided by the developers of the blockchain. Hence, it is trivial to “blacklist” certain transactions in the software and “rewrite history”: if the majority of participants accept such a decision by the developers, then the effect is that of the transaction never having happened. While this seems to be a solution to deal with fraudulent smart contracts, it is important to note that it is a bottleneck: developers can impossibly keep track of all fraud.

In summary, the distinguishing feature of smart contracts is *not* their expressivity: smart contracts cannot express any computation or workflow that classic software could not also express. The appeal of smart contracts lies in the way *a larger network executes them and provides a transparent record of every invocation*. This is a fundamental difference to software as it is run today in financial institutions, insurances, government agencies etc. A record of all transactions is stored with *all* participants in the blockchain.

Another limitation is also crucial: smart contracts cannot reason about the "meaning" of terms and whether they are fulfilled or not. They cannot make decisions that we traditionally associate with judges interpreting written law or ombudspeople mediating. In the context of ODR, this means that humans will still usually be required to assess cases, just as is the case now. Smart contracts do not remove the *human* scalability problem of ODR. Their potential lies in a *unification of technical procedures*.

As with all programs, smart contracts can contain flaws and errors that even highly experienced programmers will not spot.[18] This has repeatedly been shown to be highly problematic, with the case of "The DAO" being the most spectacular one. "The DAO", short for Decentralized Autonomous Organization, was the first smart contract that attempted to implement the workflows of an entire organisation and make it execute "autonomously", i.e. without human managerial activity. In the case of "The DAO", the rules in the smart contract would have allowed investors to vote on projects to fund; profits would have flowed back to the investors. Anyone could become an investor by acquiring "The DAO tokens", which represented investor rights and were implemented with smart contracts[19].

Unfortunately, in the case of "The DAO", a relatively simple coding error resulted in a massive breach, with the equivalent of 3.6m Ether stolen by an attacker (then worth around $US 60m). The attack could not be contained due to the autonomy of the smart contract. In

[18] We also note a famous mathematical result: the logic that underlies all computers implies that there will always be programs that contain errors that cannot be detected by another program.

[19] Token is a term that is currently mostly used in the context of Ethereum. A token is implemented in (one or more) smart contracts: sending a certain amount of currency (Ether) to the respective smart contract results in the sender being awarded a certain number of tokens. These tokens can either represent units of a new currency, or they can entitle the owner to execute certain functionality in smart contracts that accept these tokens. As such, tokens are a layer on top of Ethereum: they allow the development of applications that are based on tokens specific to the application's needs. Indeed, tokens are such a popular use case that the so-called ERC20 standard provides rules how they should be implemented.

the end, the Ethereum developers intervened and blacklisted all transactions relating to "The DAO" in the next release of their software, eradicating the attack from history. While this is the best-known case, many others flaws have been reported, with surveys available[20]. In a recent study[21], Brent et al. built a tool to investigate all 141,000 smart contracts in the Ethereum blockchain. They could show that up to 60% of smart contracts contained errors of the kind that made "The DAO" exploitable[22]. The authors also found that a single-digit percentage of smart contracts fail to secure their Ether holdings against unauthorized access.

4. A model combining smart contract technology with humans-in-the-loop

In the following, we present a regulatory model that combines smart contracts and human intervention as an extension of the principles of current ODR systems.[23] The model helps improve the efficiency of handling cases that are of a common nature (i.e. they are very similar) while maintaining the flexibility needed for legal reasoning and interpretation. In this model, the transparency provided by the blockchain's logging of all transactions is an invaluable source of information for later auditing; it also allows to collect data about cases, complaint types, and outcomes for further investigation when necessary. This information is automatically timestamped and available to all participants. Although we do not define exactly how case information is stored (e.g. supporting documents), we note that the blockchain itself usually does not offer such a feature. Such data is usually stored externally, but it can be referenced in the blockchain in an integrity-preserving way: any later attempt to modify referred documents would be immediately detectable.

Our model is based on a proposed application for smart contracts: so-called DApps - distributed applications. DApps are meant to be applications that "live" on the blockchain,

[20] Atzei N, Bartoletti M, Cimoli T. A Survey of Attacks on Ethereum Smart Contracts. In: Springer-Verlag New York, Inc. Springer-Verlag New York, Inc.; 2017; New York, USA: 164–186.

[21] Lexi Brent, Anton Jurisevic, Michael Kong, Eric Liu, Francois Gauthier, Vincent Gramoli, Ralph Holz, Bernhard Scholz: Vandal: A Scalable Security Analysis Framework for Smart Contracts, https://arxiv.org/abs/1809.03981.

[22] Note that this does not necessarily mean that all these contracts are similarly exploitable in practice: the errors could also be in "harmless" methods.

[23] For existing different ODR platforms, see Louis Del Duca, Colin Rule, and Zbynek Loebl, Facilitating Expansion of Cross-Border E-Commerce - Developing a Global Online Dispute Resolution System (Lessons Derived from Existing ODR Systems – Work of the United Nations Commission on International Trade Law), 1 Penn. St. J.L. & Int'l Aff. 59 (2012), http://elibrary.law.psu.edu/jlia/vol1/iss1/4.

i.e. the deployed blockchain environment provides an execution environment, including messaging and integrity-preserving references to external storage, for the needs of the applications. The application logic, i.e. its functionality, is implemented as smart contracts. We note that, at present, there is a huge gap between vision and reality: no blockchain offers such a complete execution environment. Current proposals for DApps focus on implementing them as websites whose every activity is recorded in smart contract transactions on the blockchain. This adds transparency and allows to use its currency for payments as well.

In our model, the ODR is implemented as such a DApp: an online application allows access via the browser, and the blockchain is responsible for transparency and payment. This establishes a foundation for further-reaching ODR: other ODR systems can be linked to an existing one by invoking the methods that the smart contracts implement. The result is that different ODRs can implement the same processes, with the blockchain guaranteeing auditability for all.

Using transparent audit records, which are broadcast throughout the entire network, is also notable as it creates another form of checks-and-balances: it exercises pressure on the human arbitrator and thus disincentivizes unfair or biased judgement. The steps leading up to a case (which is presented to the arbitrator) can be defined in code that is transparent to everyone. The arbitrator cannot easily abuse their power. Once a human arbitrator has made their decision, the remaining steps of ODR, e.g. payout of settlements, or rejections, can then be facilitated in smart contracts as well. This establishes a completely transparent chain of events: from initial claim to outcome. The standardization helps to streamline the process and make it faster. Smart contracts also allow to establish feedback channels. These range from simple positive/negative "voting" (thumbs up/down) to more complex designs, e.g. providing feedback while at the same time expressing support for another proposal. Votes are again transparent throughout the network. This lends itself naturally to feedback regarding agreement or disagreement with the decision of the arbitrator, either by affected parties or by external parties who are merely observing the ODR.

In addition to transparency, standardization may be the strongest contribution smart contracts make in our model: larger regulatory bodies can provide templates that (partially or completely) define an ODR process. The points where human arbitrators must interact would be well defined in the template. In this way, efficiency would be combined with sufficient room for further reasoning and interpretation requiring legal judgement. Companies can choose (or be mandated) to use the template. For example, the EU could standardize certain features of ODR platforms EU-wide; these templates could also be used outside the EU in efforts to harmonize the ODR implementation. Typical templates can be provided to lodge complaints, categorize dispute cases, and prioritize the processing of cases based on certain characteristics (e.g. urgency, monetary harm). The human decision remains a part of the overall process, but the human interaction would occur at precisely defined moments, with clearly specified inputs and outputs. Only more exotic cases might have to be done outside of this system (but still with a record of this fact kept). Payments could be transparently linked to decisions; even more complex financial structures can be implemented in transparent ways. An example could be debt issuances, where par value, tenor, and payment structure are parameters of a smart contract template. We summarize these aspects of our model in Table 1. However, we emphasize that the use of smart contracts is not without risks, as discussed above in terms of access control and ownership of smart contracts.

5. Regulatory implications and potential from the use of smart contracts in ODR

The claimed potential of blockchain-based smart contracts is to make complex exchanges more transparent and reduce the means of undesired interference with transactions.[24]

***Table 1*: Smart contracts in ODR**

Aspect	*Relevant features of smart contracts*	*Immediate effects of using smart contracts*	*Outcome*

[24] See Paul H. Farmer, Jr., Speculative Tech: The Bitcoin Legal Quagmire and the Need for Legal Innovation, 9 J. BUS. & TECH. L. 85, 89 (2014).

Transparency	Every participant has the same view of the definitions/methods of the smart contract. Every participant has the same view of the outcome.	Each participant can view dispute cases. Each participant can vote/provide feedback on decisions made by the arbitrator.	Better checks and balances exercise pressure on arbitrators.
Efficiency	Well defined processes with clear hand-over points to human arbitrators.	Certain amount of templating allows for categorization of cases. Linked into feedback and payout mechanisms.	Shorter time to dispute resolution, immediate payout
Standardization	Templated smart contracts to mandate procedures and principles of handling cases.	Reuse of well-defined procedures, reducing overhead. Possible to define variants based on templates.	Harmonization of implementation of ODR while leaving flexibility.

However, traditional regulators like central banks and governments need to adapt to the possibilities and limitations of the new regulatory space. Concerning the regulation of smart contracts in ODR systems, this means to strike a balance between the efficient processing of cases and creating an inclusive environment for increased participation, allowing those participants with fewer social resources to redress their interests in a more efficient way. Both increases, in efficiency and participation, may be gained by standardizing processes in code. Taking the EU ODR platform as described in our model as an example, EU-wide legal regulatory standards become easier to implement; at the same time all participants can be given both the right and means to access and view the transaction record and, on a voluntary basis, provide feedback for the decision-making process. Importantly, a record of all feedback is kept, and attempts to abuse the mechanism can be tracked. In the following, we discuss the changed nature of co-regulation, with a particular focus on errors in smart contracts and regulatory implications.

5.1 Changed nature of co-regulation

This new model does not preempt regulatory manipulation as it exists in traditional models of regulation. Governments can still mandate certain functionality in smart contracts, and they can be authors of ODR smart contract templates. Co-regulation takes place, but changes in nature: in this context, it means that many more actors have equal rights to view and provide feedback on the processes built into smart contracts. Co-regulation would imply a combined approach of regulating by code and regulating by law. However, in such a hybrid co-regulation approach, many principles need to be further clarified: how is access control defined, i.e. which parts of smart contracts can be activated, updated or deactivated, by whom, and under which conditions.

This form of co-regulation opens participation in the regulatory process to any interested actor. Primary actors include the traditional actors as we know them from regulation of financial systems, but also new intermediaries (brokers, dealers, and other entities interacting with end users), and finally individual consumers. Traditional regulators as we know them from the current financial system continue to exist but would adopt an additional role: the templated processing of ODR cases allows to analyze decisions on a much larger scale, while taking the amount and quality of feedback into account. Intermediaries are typically objects of ODR cases - payment suppliers would be a typical example. An ODR case should ideally contain a record of financial transactions relating to the disputed exchange. As is the case today, intermediaries in these transactions may often be located in different jurisdictions. This is a classic cross-border issue. The advantage of templated ODR smart contracts would be that intermediaries find it more acceptable to participate in the ODR process thanks to the code being unambiguous, hence reducing uncertainty and potential for disputes how to interpret the ODR process itself. Individual consumers are empowered in two ways. First, the existence of templated ODR contracts makes it less likely that they will be subjected to conditions pre-determined by corporations alone, without reasonable opportunity to negotiate terms. Second, consumers can provide much more immediate feedback and catch the regulators' attention, who can react faster to a growing number of similar disputes. This may help prevent unfortunate results as in previous settings, where single courts often did not enforce terms proffered by consumers in a contract.[25]

5.2 Errors in smart contracts

Even though code is unambiguous, smart contracts can still contain errors. Two kinds of errors can occur. The first kind of error we need to consider is when the developers of the smart contract misinterpret the (legal) meaning of ODR processes and implement them

[25] See, e.g., James Gibson,J.; Vertical Boilerplate, 70 WASH. & LEE L. REV. 161, 167–69 (2013) ; Cheryl B. Preston & Eli W. McCann, Unwrapping Shrinkwraps, Clickwraps, and Browsewraps: How the Law Went Wrong from Horse Traders to the Law of the Horse, 26 BYU J. PUB. L. 1, 23 (2011).

incorrectly. The second error occurs when the creators' interpretation is correct, but their use of the programming language results in an exploitable bug: seemingly correct code can be the target of an attack such that the attacker can 'take over' the smart contract, i.e. cause it to run incorrectly and execute steps that are controlled by the attacker, and not intended by the original developer. Errors of the second kind can often, but not always, be detected by automated tools - a variety exist, e.g. the one by Brent et al.[26], Kalra et al.[27], or Luu et al.[28], to name a few. Theoretical limitations make it impossible to universally identify all possible errors. As of today, errors of the first form cannot be detected automatically at all as no machine can reason about the correctness of a mapping of a legal interpretation to code. This is still a task for humans.

5.2.1 Co-regulation: intervention in the case of errors

Incorrectly executing code should be halted and any wrongly executed steps undone. However, the blockchain execution model makes intervention of any kind relatively difficult. It would require (human) intervention by a recognized actor with appropriate mandate. To fix an error of the first kind, the erroneous smart contract would have to be terminated and replaced or updated. This is a lengthier process and likely more costly than an advisory and clarification issued by a traditional regulator. However, it is relatively clear wherein the problem lies and the record of previous transactions can be rolled back in an orderly fashion, and an update mechanism in the smart contract may be able to solve the problem.

In the case of the second error, intervention may be significantly harder, and financial loss harder to prevent. The problem is the presence of an active attacker who is taking active steps to corrupt the processes. A faulty smart contract can be terminated or changed to prevent further losses just like in the first kind of errors, but time is of the essence: as we have seen in the case of The DAO, heavy losses can be incurred within very little time.

[26] Ibid., 21.
[27] Kalra S, Goel S, Dhawan M, Sharma S. ZEUS: Analyzing Safety of Smart Contracts. In: Networks and Distributed Systems Security Symposium, (2018).
[28] Luu L, Chu DH, Olickel H, Saxena P, Hobor A. Making Smart Contracts Smarter. In: ACM Computer and Communications Security Conference, (2016).

Undoing the damage also required intervention and crucial changes to the actual blockchain software. To do this in our ODR model, governments or multi-government bodies are likely the only actors in a position to mandate such changes and see them applied within a short time. Any delay would harm the trust of actors in the ODR process. An advantage of using blockchain-based smart contracts, however, would be that (criminal) investigation would be simplified as experts can view the audit trail and transaction history, which also provides the necessary information in short order to determine each party's liability and possible compensations to be made. To address this second risk of smart contracts, the code for ODR processes may well need to be written and maintained by engineers employed by a given organisation (e.g. the EU, to stay in our example) - for many, this might incur substantial changes in their operational models.

5.2.2 Co-regulation: risk allocation and responsibilities in case of erroneous smart contracts

Intervention by governments is not the only (co-)regulatory implication that one needs to take into account. We argue that smart contracts do not obviate the need for a number of classic, supporting mechanisms that are implemented outside the actual blockchain/smart-contract system.

Risk allocation: Since errors in smart contracts are unavoidable, relying organisations need a clear understanding of their legal obligations, and the obligations of their partners, with respect to risk allocation and distribution. In particular, financial organisations must consider how risks should be allocated if their trade partners lose funds - an important consideration in case of intermediaries.[29] Such arrangements need to be in place outside of the definitions of the smart contract, and before its deployment begins or the risk materializes[30]. Inconsistent understanding between different traders would otherwise collide with legal compliance in terms of liability and agreement which party needs to become involved in which way.

[29] Stuart D. Levi, et.al., An Introduction to Smart Contracts and Their Potential and Inherent Limitations, https://corpgov.law.harvard.edu/2018/05/26/an-introduction-to-smart-contracts-and-their-potential-and-inherent-limitations/#16, (accessed 26 May 2018).

[30] Risk allocation cannot be implemented in another smart contract: errors could also exist in that contract's definition.

Technical support for risk allocation: As risk allocation must be defined outside the smart contract, there remains a need for technical infrastructure and support to define, assess and quantify risks, and develop strategies to handle risks of different kinds. Smart contracts cannot eliminate this cost; the infrastructure support must provide the room for legal interpretation. The aftermath of the attack on The DAO may serve to illustrate the consequences of unclear risk allocation and procedures. Since the The DAO was advertised precisely with the understanding that all rules were laid out in non-ambiguous terms in code, was the attacker not within their rights to exploit the bug and extract the funds? This question was considered seriously by the developers of Ethereum and led to a split of the community. The main developers decided to change the blockchain software such that all transactions relating to the attack would be ignored by participants - in essence, "rewriting the immutable blockchain". However, a sizable faction disagreed and continued to work with the old software, hence creating a second, "forked" blockchain *with identical smart contracts running*. We also note that support infrastructure of the kind required has been envisioned, but not yet implemented in existing ODR platforms, either. For example, it is absent in the current EU-wide portal for ODR, which was established in 2016, under Regulation (EU) No 524/2013 on online dispute resolution for consumer disputes[31], which allows consumers and traders to submit complaints which can be directed and addressed through the platform.[32]

Some old regulatory challenges still remain: As we have shown, regulation implemented in smart contract code does not exclude traditional regulators and intermediaries from participation. However, once errors as described above occur in smart contracts, with potentially high financial losses, then judges and juries must still review and interpret the meaning of code and its legal basis.[33] The blockchain system may seem to make it harder

[31] European Commission, Report from the Commision to the European Parliament and the Council on the functioning of the European Online Dispute Resolution platform established under Regulation (EU) No 524/2013 on online dispute resolution for consumer disputes, 13 December 2017, https://ec.europa.eu/info/sites/info/files/first_report_on_the_functioning_of_the_odr_platform.pdf; See Regulation, https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2013:165:0001:0012:EN:PDF.
[32] The platform, which was launched in January 2016, is available at https://webgate.ec.europa.eu/odr/ (accessed 13 Apr 2016).
[33] If smart contracts proliferate,judges and juries will have to review them to determine their legal basis and evidentiary status.

for attackers to disguise their real identity because executing (attacking) a method in a smart contract may require a form of authentication. However, attackers may simply compromise the identity of another, legitimate participant by classic identity theft and use the assumed identity to stage their attack. The problems of attribution of attack and liability for protecting one’s identity are the same as of today. This may be exacerbated by a high number of intermediaries, including those in other jurisdictions, who participate in the ODR system.

6 Summary

Smart contracts, run on blockchains, are often cited as a game-changing innovation that can make ODR systems more scalable. In this chapter, we have analyzed this view critically and mapped the regulatory space and implications of using smart contracts for ODR.

We gave an in-depth description of the technology behind smart contracts and highlighted their technical potential, but also their limitations. In terms of expressivity, smart contracts do not replace the human arbitrator as they cannot reason about the validity of a claim in the context of law. All steps of a smart contract, including the creation of the initial claim and the final decision by the arbitrator, are triggered by participants in the ODR system.

However, smart contracts do offer some true advantages. We provide a model that makes good use of these to improve the ODR process. First, the blockchain system provides complete transparency about the executed steps, creating an audit trail that is hard to forge and where details can be easily retrieved by participants. This leads to much higher accountability for the arbitrators in the decision-making process. For ODR, governments can implement regulation in the form of providing smart contract templates that define procedures that are legally required. Templates contain the standardized methods that can be executed by a multitude of ODR platforms, eliminating the need to implement them independently and hence reducing the degree of legal uncertainty in terms of compliance. Such smart contracts also provide a unified way to gather data about ODR outcomes and claims, thus allowing to analyze the effectiveness of the ODR process with more ease and a higher degree of confidence. Importantly, an efficient feedback mechanism can be implemented to collect opinions from participants whether the outcome met their expectations or not. This allows for a new form of co-regulation.

Ultimately, smart contracts are not a panacea. First, they introduce new problems as they may contain errors of two kinds: wrong procedures due to legal misunderstandings, and errors that make the code vulnerable to exploitation. We showed that intervention is not easy in either case, and damage can occur fast and be hard to undo. A classic set of mechanisms and support infrastructure still needs to be in place to deal with risk assessment and risk allocation in case of errors.

Dr. Janet Hui Xue*

Janet Hui Xue is Research Associate at the University of Sydney where she is affiliated with the Department of Government and International Relations, School of Social and Political Science. She is also a Research Associate at the One Belt One Road Programme at the Faculty of Law, and Visiting Scholar at Wolfson College, both at University of Oxford. Her contact information is hui.xue@sydney.edu.au or hui.xue@wolfson.ox.ac.uk.

Dr. Ralph Holz**

Ralph Holz is Lecturer in Networks and Security in the School of Computer Science at the University of Sydney. He is co-appointed at the Sydney Nano Institute, where he is Theme Leader for Communications, Computing, and Security. Ralph can be reached by email at ralph.holz@sydney.edu.au.