CLIFFORD CHANCE

European Bank
for Reconstruction and Development

SMART CONTRACTS: LEGAL FRAMEWORK AND PROPOSED GUIDELINES FOR LAWMAKERS

OCTOBER 2018

CONTENTS

European Bank
for Reconstruction and Development

CLIFFORD
CHANCE

EXECUTIVE SUMMARY

Smart contracts and distributed ledger technology have the potential to transform the way in which businesses enter into, perform and enforce transactions. Increased automation of these processes via smart contracts could increase efficiency and lower costs, particularly for sectors that use high volumes of standardised contracts. Use of smart contracts may also have other benefits, such as reducing the risk of human error, increasing transparency and potentially reducing the number of disputes that end up in court.

To date, much of the discussion has centred around the possible use cases and practical benefits of this new technology. At the same time, lawyers and lawmakers have started to identify particular legal questions or concerns that arise in the context of smart contracts and distributed ledger technology. These include issues relating to consumer protection, liability allocation, fraud risks, enforceability and data protection.

In many cases, the existing legal and regulatory framework may already allow for the use of smart contracts. Courts are generally used to applying legal principles, tools and techniques to new situations. However, this may not always result in a desirable outcome from a policy perspective, or some laws may not be flexible enough to allow requirements to be fulfilled in an automated manner. Therefore, in some cases it may be helpful or even necessary for lawmakers to clarify or amend the way in which existing laws can be applied to smart contracts, or to introduce new legislation; for example to recognise the use of distributed ledgers as records of ownership if existing laws would not allow such recognition.

This paper is intended to provide a guide for lawmakers in considering these issues and seeking to create a legislative and regulatory environment that facilitates the appropriate use of smart contracts. It examines what smart contracts are and what functions they may perform. It then proceeds to consider some key areas of law that are relevant to the use of smart contracts, and seeks to provide practical guidance and recommendations for lawmakers when seeking to promote or facilitate the use of smart contracts.

INTRODUCTION

Efficiency and cost considerations are at the heart of the discussion as to why parties may wish to use smart contracts. Potential efficiencies arise from the fact that smart contracts use software to perform certain tasks relating to the contract through automated processes. Indeed, automation may serve to reduce expenditure of time and costs for the contracting parties. For instance, automation may be used to streamline front or back office procedures for entering into, performing or enforcing transactions. Of course, automation itself is not a new concept. Automated processes and services pervade much of modern life, from automation in the manufacturing industry to online banking, or even mundane examples such as the use of vending machines. However, at present, legal contracts are generally not automated, and require human input to draft, conclude, perform and enforce their performance. These can be time and resource-intensive processes. Therefore, the automation of some or all of these processes may well bring benefits in terms of increased efficiency.

Smart contracts may be particularly effective for sectors that use highly standardised contractual terms without material deviation. By way of example, in the mass transit and business sectors, the benefits of scale may exceed the cost of establishing and maintaining the software infrastructure. Smart contracts may also benefit society as a whole by reducing public or private sector costs, or the number of court proceedings. In addition, smart contracts may give rise to other benefits such as making it easier to enter into contracts without requiring an intermediary, or streamlining verification of identity, or execution and recording of transactions (as, for example, in the case of automated vehicle road tax payments). Smart contracts may also promote the use of electronic signatures, increase transparency and reduce conflicts by the automated enforcement of rights (and so avoid the need to remind the other contracting party to perform its obligation or to bring and enforce claims in court).

Reducing or shifting risk is a further important advantage of smart contracts, from both an individual and a systemic risk perspective. Efficiently automated cash flows reduce operational and counterparty risks. Automation shifts the risk of human failure in individual cases (e.g., if a human makes an error when carrying out actions under a traditional contract) to risks resulting from the use of software, which would run according to the way in which it has been coded. Accordingly, failures or errors in the performance of smart contracts may potentially be better limited and monitored (e.g., by checking the software has been coded correctly).

In the context of current discussions on the use of smart contracts, distributed ledger technology ("**DLT**") is generally understood as the primary means of automatically concluding, performing and/or enforcing the smart contract.[1] This brings a number of potential advantages, including increased transparency and efficiency, as all participants in a DLT network will have an identical copy of the shared ledger, which is updated according to the distributed ledger's consensus method (instead of relying on a trusted central authority or intermediary to hold and update records). Use of DLT also automates reconciliations across the network, and gives the parties to a smart contract a high level of assurance that automated actions under the smart contract cannot be interfered with.

Whilst a world in which contracts are concluded, executed and/or enforced solely by machines may be some way off, the dawn of an era of greater automation of contracts and the first steps towards smart legal contracts is upon us. This move towards automation is generally accepted as positive on the basis that it can lead to a reduction in friction and cost in the contracting process and, in general terms, achieving these reductions is considered to be a good result. Smart contracts have a number of potential applications – the Chamber of Digital Commerce's Report illustrates 12 possible use cases of smart contracts.[2]

1 The term "smart contract" was coined by Nick Szabo several years before the emergence of DLT; for example, see his 1994 article on Smart Contracts, available at http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html. However, in recent years, much discussion has focused on smart contracts that use DLT, and so the two concepts are often used synonymously.

2 See Chamber of Digital Commerce: "*Smart Contracts: 12 Use Cases for Business & Beyond*" (December 2016), available at: https://digitalchamber.org/wp-content/uploads/2018/02/Smart-Contracts-12-Use-Cases-for-Business-and-Beyond_Chamber-of-Digital-Commerce.pdf

Many of the countries in which the EBRD operates have been exploring possible uses of DLT and smart contracts. For example, both the Georgian Public Registry and the Ukrainian e-Governance Agency have been working on facilitating the use of smart contracts in real estate transactions, with proposals to create new DLT-based registers (amongst other things), while the National Bank of Belarus is considering facilitating the use of smart contracts in banking transactions.[3]

Existing legal regimes may have many principles, tools and techniques that can be applied to smart contracts in the same (or a similar) way as to other contracts. Indeed, courts are generally used to applying existing legal principles to novel situations. Nevertheless, there is a growing realisation that current laws (most of which significantly predate the development of smart contracts) do not have smart contracts specifically in mind. Consequently, they may not be flexible enough to encourage the use of smart contracts, or the outcomes of applying existing principles, tools and techniques to smart contracts may not always be desirable. A key initial task, therefore, is to assess the existing legislative and regulatory framework to determine whether adjustments to existing laws may be necessary or desirable to better facilitate the use of smart contracts. New legislation might also be needed; for example, to specifically recognise the use of distributed ledgers as a record of ownership of an asset, which existing law might not allow.

There is an implicit assumption that the development and use of smart contracts should be encouraged. In general, this seems reasonable, not least due to the potential benefits of automation. However, there may be circumstances in which the adoption of smart contracts might not be appropriate (for example, in the case of contracts with consumers, unless appropriate consumer protection measures are taken, such as ensuring that the consumer understands the contract). Consequently, another important task for lawmakers will be to identify these types of circumstances and clarify their expectations in those cases.

This paper has been written specifically with a view to positively contributing to the existing literature and thinking on smart contracts. With this in mind, the purpose of this paper is to provide guidance to lawmakers on some of the key areas of law that should be considered in the context of smart contracts. In order to do so, it is important that there is a common understanding of what is meant by smart contracts and what functions smart contracts might perform. Section 1 of this paper therefore provides a framework setting out what we mean by smart contracts and the functions they might perform. Section 2 of this paper then examines, within that framework, the key legal considerations and a roadmap for lawmakers as to the issues they should consider in seeking to create a legislative environment that facilitates the appropriate use of smart contracts.

3 See, for example, "*Georgia to use smart contracts in real estate registrations*" (February 2018), available at: http://agenda.ge/news/96094/eng; "*A house has been bought on the blockchain for the first time*" (October 2017), available at: https://www.newscientist.com/article/mg23631474-500-a-house-has-been-bought-on-the-blockchain-for-the-first-time/; and "*Belarusian banks may be allowed to sign smart contracts*" (April 2018), available at: http://eng.belta.by/economics/view/belarusian-banks-may-be-allowed-to-sign-smart-contracts-111225-2018/

1. What are smart contracts?

Let's start with a basic, but fundamental, question: what are smart contracts? Unfortunately, there is no single answer to this; even within legal and IT communities, there is no universally recognised notion of what a smart contract is. Rather, the term is used for a broad range of concepts based on two constituent elements: a "contract" and the property of being "smart". We consider what these elements mean and what the different conceptions of a "smart contract" are, in order to test how certain aspects of legal "contracting" might work in the context of smart contracts.

1.1 Exploring the terminology of smart contracts

This part of the paper examines the elements of a smart contract and demystifies some of the terminology used in the context of smart contracts.

1.1.1 Elements of a smart contract

Generally, a traditional contract expressed in natural language without any automatable aspects or elements (i.e., a contract which in no way consists of, and is not concluded, performed or enforced by, software) would not qualify as "smart". On the other hand, from a legal perspective, mere software or computer code without an agreement between the parties making use of that software or code cannot constitute a "contract" in today's legal sense, no matter how effective or smart the software or code is in executing pre-programmed steps.

The term "smart" in this context, therefore, really refers to the characteristic of being automatable – i.e., the ability of software to perform certain tasks through automated processes (this is discussed further at section 1.2). In this paper, the term "contract" denotes an agreement constituted by a set of legally enforceable rights and obligations between the parties.[4] The enforceable nature of smart contracts is considered further at section 1.3. The ways in which smart contracts may be formed (as opposed to what they are) are also the subject of further discussion in this paper.

1.1.2 Smart contract code vs. smart legal contracts

Taking this a step further and bringing the elements together, some authors point to two basic ways in which the term "smart contract" is often used. In particular, they distinguish the concept of "smart contract code" from the concept of a "smart legal contract"[5]. In this paper, when we refer to "smart contracts" we are generally talking about smart legal contracts, rather than smart contract code.

"Smart contract code" refers to computer code which, when executed[6], uses conditional logic to assess whether one or more pre-defined conditions are met and, if so, automatically executes specific tasks. This process is sometimes referred to as "if/then" logic or "if this then that" or "IFTTT" programming (the "this" being the conditions or triggers and the "that" being the tasks which are then performed).

These tasks could have contractual or legal relevance. They could involve the software taking steps to discharge an obligation (e.g., to issue instructions to make a loan repayment) or to exercise rights (e.g., to send a termination notice on a lease or a derivative contract). The tasks could also result in the transfer of ownership of assets (e.g., if the task is to update a property ownership register). However, it is not necessary that these tasks should have any contractual or legal relevance, and smart contract code need not relate to a legal contract at all.

"Smart legal contract", by contrast, refers to legal contracts which are partly or wholly represented and/or performed by software (in other words, the contractual obligations of a party to the contract are discharged through the automated performance of the software).

Taking account of both basic concepts and emphasising the aspects of automation and enforceability, some authors have tried to define smart contracts in a more encompassing manner, as follows:

> *"A smart contract is an automatable and enforceable agreement. Automatable by computer, although some parts may require human input and control. Enforceable either by legal enforcement of rights and obligations or via tamper-proof execution of computer code."*[7]

4 Within the IT community, the term "contract" is sometimes used differently and may be understood in that context as referring (merely) to self-executing code rather than to legally enforceable rights, obligations and other arrangements agreed between contracting parties. This highlights an important issue about the semantic difficulties or misunderstandings that can often arise in the context of discussions on smart contracts due to different professions using terms of art or terms which are vernacular to a particular group.

5 See, for example: "*Smart Contract Templates: foundations, design landscape and research directions*", Dr. C. Clack, V. Bakshi, Dr. Lee Braine (4 Aug 2016), available at https://arxiv.org/pdf/1608.00771.pdf v3 [cs.CY] 15 Mar 2017.

6 The term "executed" here is used in the computing sense of "run" or "performed", rather than the legal sense of "signed". Again, this is an example of where semantic differences and difficulties may arise, where the same terms may have different meanings when used by different professions or groups.

7 See *supra*, note 5.

We consider the two key elements of this definition (automation and enforceability) further at sections 1.2 and 1.3 below.

1.2. Automation as a key element

As stated above, a smart contract not only requires an agreement between the contracting parties, but is also characterised by being automatable in whole or in part:

> *"Automatable by computer, although some parts may require human input and control."*

Hence, there must be some degree of self-conclusion, self-performance or self-enforcement by software action (see section 1.3 below regarding self-enforcement by tamper-proof execution of computer code), where the contracting or third parties are not required to take action or intervene. Some human input, however, would not necessarily undermine the "smart" nature of the contract. Non-automatable contracts are not smart irrespective of whether they are concluded or represented electronically (for example, a contract which is simply concluded electronically by exchange of emails or which is set out in an electronic file) or relate to software (for example, an agreement pursuant to which software is purchased or leased) are referred to as "**non-smart contracts**".

"Smart" contracts are not necessarily intelligent – in fact, in many cases it is probably quite a misnomer. A contract should only be referred to as "intelligent" where the software is able to do and does more than automatically perform simple pre-set conditional action – i.e., the automation is built upon some kind of artificial intelligence ("**AI**").[8]

Automation in a DLT environment

Automation of actions generally requires the use of computers or other devices on which software that executes those actions can run. Automated conclusion, performance or enforcement of contracts or, more generally, execution of relevant tasks may occur on a shared or distributed ledger. While smart contracts and DLT or blockchain[9] (as the most well-known application of DLT) are often considered in the same breath, they are, in principle, distinct from one another.

DLT, however, has certain characteristics which make it attractive for smart contracts. DLT consists of a set of recorded data distributed within a computer network in a manner that each participant (or "node") holds the (identical) set of records. Recording a set of data or changing a record in such a shared database requires consent of the nodes according to the distributed ledger's consensus method. Once validated and approved by the nodes, the data is recorded or updated on each node concurrently so that the data is identical on each node at all times. The computer network can be public or private and both the access to the network and/or the ability of its participants to view specific data held in the network can be restricted.

In some cases, a distributed ledger may have one or more "operator(s)" with overriding administration rights, for example to make changes to the distributed ledger system code. Having such an operator may seem counterintuitive, given the distributive character of a distributed ledger and the underlying original concept of replacing trusted central authorities or intermediaries with trust in the system itself. However, there are several DLT platforms that have been developed, and continue to be administered in some way, by an operator.[10]

Under this model, users would typically be licensed to use the distributed ledger system by the operator and may then create their own DLT networks by running their own software applications that work with the underlying distributed ledger system software. This type of model may, for instance, be used for enterprise-grade financial services-based blockchain applications. Nevertheless, it is important to note that this is still fundamentally different from a traditional centralised model, where a central authority or intermediary maintains the definitive set of records relating to the transactions or accounts.[11]

8 Based on the software performing simple pre-set conditional action only and the above understanding of smart contracts as mere "smart contract code", it has, indeed, become a saying that smart contracts are neither "smart" nor "contracts" within the traditional meanings of these words.

9 Blockchain is a distributed ledger of "transaction" data which is managed by a peer-to-peer network. It consists of a continuously growing list of records (blocks) which are linked and secured using cryptography. Each block typically contains a cryptographic hash of the previous block, a timestamp and transaction data which makes it inherently resistant to modification of the data. Any change in the database requires creating a "transaction" which is cryptographically signed by its creator and which is recorded as a new block upon acceptance by the blockchain participants.

10 For example, R3 operates the Corda platform (see https://www.r3.com/corda-platform/) and IBM operates the IBM Blockchain (see https://www.ibm.com/blockchain/solutions). In general, it is possible to set up DLT platforms in a way that grants one more participants overriding administration rights. For example, in the context of a DLT-based property ownership register, it may be seen as helpful or desirable for the public authority operating the register to have certain administration rights, such as the ability to "correct" the register to reflect a court judgment that a property transfer was not valid.

11 For further background on DLT, see Distributed Ledger Technology (DLT) and Blockchain, FinTech Note No. 1, World Bank, 2017, available at: http://documents.worldbank.org/curated/en/177911513714062215/pdf/122140-WP-PUBLIC-Distributed-Ledger-Technology-and-Blockchain-Fintech-Notes.pdf and *Distributed ledger technology in payment, clearing and settlement: An analytical framework*, Committee on Payments and Market Infrastructures, February 2017, available at https:// bis.org/cpmi/publ/d157.pdf

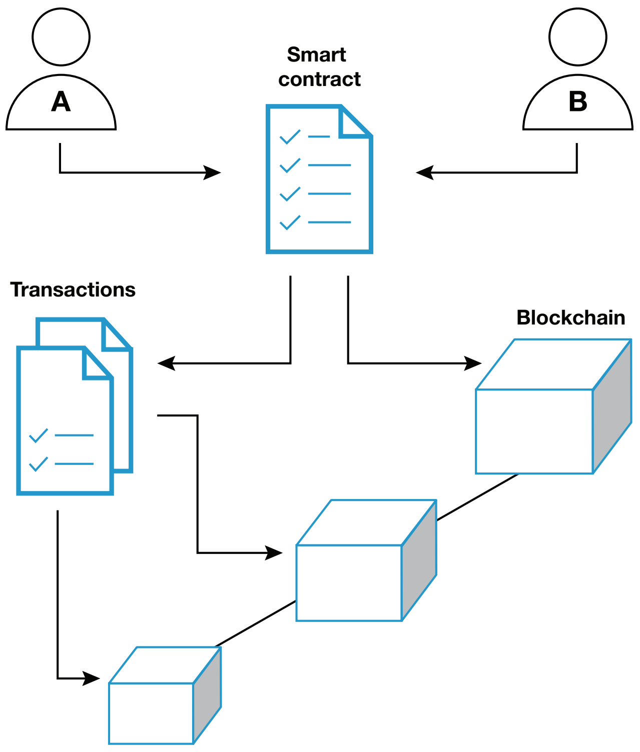
Even in a more "traditional" DLT model where there is no formal platform operator there will often be a core group of developers who have a similar role in practice by virtue of their technical expertise and ability to influence opinion and build consensus amongst other users.[12] Therefore, where we refer in this paper to a platform "operator", similar considerations will also often apply in respect of such a core group.

The data stored on the distributed ledger might be static (such as a date of birth or a name of an individual), dynamic (such as a current record of ownership of assets) or executable (such as a smart contract). The distributed ledger can be connected to external systems and data sources by means of an interface. An interface connecting a distributed ledger to a trusted data source or other input is generally referred to as an "oracle".

Using DLT as the basis upon which smart contracts may be concluded, performed and/or enforced has the advantage that there is only one binding software version of the smart contract embedded in the distributed ledger. Hence, frictions resulting from the parties using different software or different versions of the same software can be avoided and the parties have a high degree of assurance that the automated software-executed actions cannot be interfered with.

Figure 1 below illustrates how blockchain (or another form of DLT) could be used for smart contracts. In the diagram, parties A and B agree a smart contract, the terms of which are recorded in a "block" on the distributed ledger, or blockchain. The parties then enter into individual transactions (or individual contracts) under the smart contract, each of which is also recorded in a block on the blockchain. Once each block is confirmed as valid according to the relevant consensus method, it is linked or "chained" to the previous blocks in the blockchain.

Figure 1: Using blockchain for smart contracts

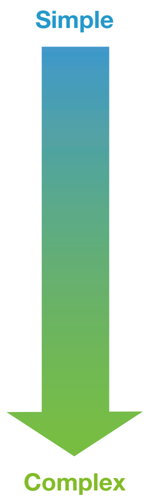


12 For further discussion of the different roles and hierarchy that may exist in a distributed ledger, see: "*The Distributed Liability of Distributed Ledgers: Legal Risks of Blockchain*", Dirk A. Zetzsche, Ross P. Buckley and Douglas W. Arner ([2017] UNSWLRS 52), available at http://www5.austlii.edu.au/au/journals/UNSWLRS/2017/52.pdf

Suitability for encoding and automation

In theory, some contractual clauses may be readily susceptible to encoding and automation. However, as further discussed below, not all terms are so susceptible. We refer to this propensity to be encoded or automated as "automatability". The spectrum of automatability of contractual clauses can be represented in the following diagram (Figure 2).

Figure 2: Spectrum of automatability of contract clauses



Monetary transaction: "Transfer £1 from A to B 12:00 GMT on 1 Jan 2018."

Asset ledger transaction: "Transfer ownership of asset X from A to B."

External input: "Add interest at LIBOR + 2%."

Participant input: "Unless A and B otherwise agree, perform X on 1 Jan 2018."

Dispute resolution: "If A and B do not agree, C shall decide."

Meta-clauses: "If any term of this contract is held to be unenforceable, the remaining terms of the contract will remain in effect to the extent that they are not invalid or unenforceable."

Use of complex legal concepts: "A will use its reasonable endeavours to do X."

Clearly, smart contracts lend themselves well to agreements with clear conditions and repetitive transactions. Of course, automatability will always depend on the type of software used. If the software is merely able to "tick off" simple conditions and execute specific actions, fewer clauses may be automatable than if the software can assess or interpret external data, facts and/or legal requirements, and/or make use of AI, and, ultimately, take decisions on its own. In any case, the software may have to be linked via an oracle to external data sources to be able to verify whether the conditions that trigger the relevant actions are met. These data sources can be diverse in nature (e.g., calendars, email correspondence, registers, databases, accounts or the text of laws and regulations).

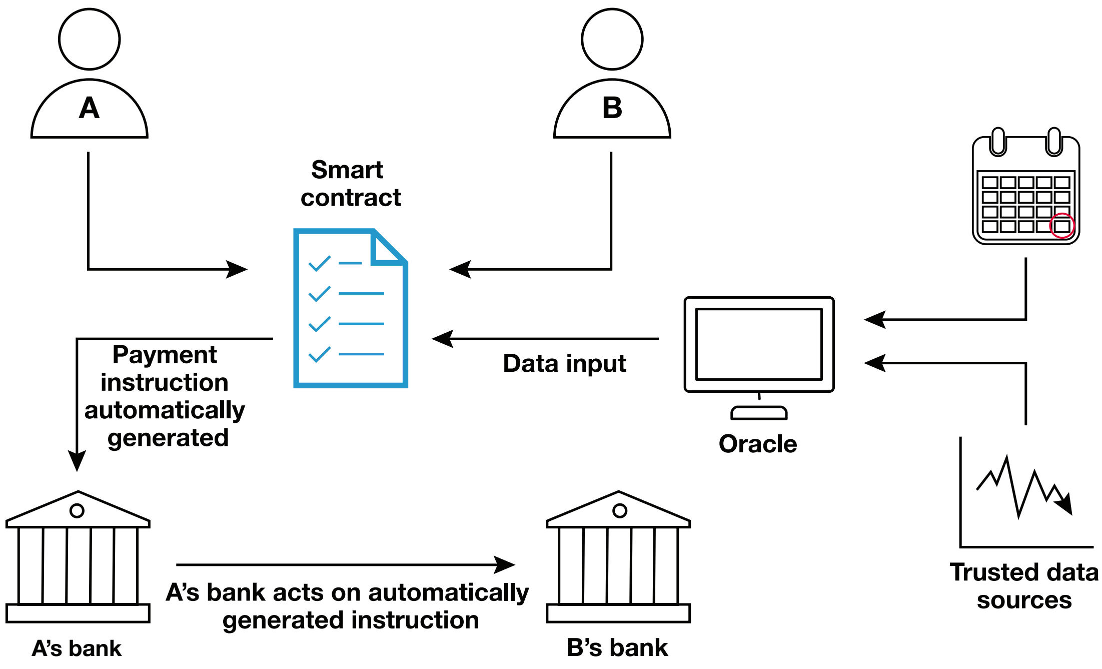
Set out below are a number of examples which further illustrate the spectrum of automatability:

- Clauses with simple conditional (if/then) logic and simple conditions triggering an action may be more easily encoded and automated than clauses which provide for a factual or legal assessment or for exercise of discretion.

  For example, a simple conditional clause stipulating the obligation to pay a specific amount or to deliver a specific asset on a specific payment or delivery date might be encoded and automated relatively easily. In financial services, this might include encoding a loan agreement so that the software automatically triggers a monthly loan repayment when the software receives an input confirming that it is the last day of the calendar month (i.e., without the need for human intervention or instruction) or so that the software automatically changes the monthly repayment amount where it receives an input confirming that there has been a change in the relevant reference interest rate (e.g., a central bank interest rate) – in each case, the conditions can be objectively determined.

  This is illustrated at Figure 3 below. As shown in the diagram, Parties A and B enter into a smart loan contract. The smart contract software is coded to receive inputs from trusted data sources via an oracle and to automatically generate payment instructions based on those inputs, in accordance with the terms of the smart contract. When the smart contract receives an input that it

Figure 3: DLT-based smart contract


A
B
Smart contract
Payment instruction automatically generated
Data input
Oracle
Trusted data sources
A's bank acts on automatically generated instruction
A's bank
B's bank

is the last day of the calendar month, it uses the interest rate input to calculate the correct monthly repayment amount under the smart loan contract. Then, the software automatically sends an electronic instruction to Party A's bank to transfer this amount from Party A's bank account to Party B's bank account. Party A's bank acts on the automatically generated instruction and transfers the payment to Party B's account.

- However, conditional clauses requiring an assessment to be conducted "to the satisfaction" of a contracting party or to take action "in a commercially reasonable manner" are much less susceptible to automation. These elements require a subjective assessment of the individual circumstances and it may be difficult to (sufficiently) express the relevant circumstances in software language or to automate them; in particular, the condition of "to the satisfaction" may require human intervention, whereby the relevant party whose subjective judgement is required would need to indicate whether or not such condition had been met.

- Automatability may also depend on the extent to which a clause could be interpreted in different ways. This will depend on factors such as whether the wording of the clause is ambiguous or requires the exercise of judgement when interpreting the clause. For example, a material adverse change ("**MAC**") clause may be difficult to code and automate if the meaning of "material adverse change" is not further specified in the contract. Conversely, it may be easier to code and automate the operation of a MAC clause if the criteria for a "material adverse change" are specified in great detail (such as by providing an exhaustive list of events constituting a MAC, which may leave little or no room for interpretation, particularly where a change can be quantified in a precise manner). The parties may therefore seek to make contracts easier to code and automate by drafting them in a way that minimises use of deliberately flexible terms or clauses. However, this may require the parties to anticipate and provide for all possible eventualities at the outset, which will not always be possible or desirable. The scope for different interpretations of a contract or clause will also depend on the rules of contractual interpretation applicable in the relevant legal system and whether these require a literal or purposive approach. The latter may involve an assessment of the parties' true intentions in light of the broader context of the transaction or contract.

- Clauses which do not contain conditions or executable instructions but merely determine arrangements (for example, clauses stipulating the governing law or jurisdiction) can be encoded but should also be distinguished from the types of automatable clauses we have been discussing so far; i.e., they have a particular legal or conceptual function which does not directly translate into an action.
- For clauses requiring an active decision, exercise of a right or other action (for example, a decision to choose among alternative rights such as demanding delivery of physical assets or a cash payment), the intention of the parties will determine whether and to what extent this should be conferred on the software or manually exercised or authorised by the parties.
- Likewise, parties may want to provide for the possibility to agree amendments to, or to waive, certain rights or obligations (for example, to grant a grace period or to defer delivery or payment obligations) or to temporarily suspend automatic performance in certain circumstances. This is likely to be more difficult to achieve in a DLT environment (see also section 2.4.1 below).
- Automatability may also be more difficult with regard to "unwritten" clauses. In civil law jurisdictions, contracts often do not set out all relevant clauses and principles since these are often stipulated in comprehensively codified laws which are "read into" the contract. In common law jurisdictions, similarly, contractual terms may be enriched or "read into" the contract by case law and equitable principles. There may be non-mandatory provisions which apply if and to the extent not agreed otherwise. There may also be mandatory provisions which apply irrespective of, and render void, deviating contractual clauses.

The relevant software may therefore need to "be aware of", understand and assess any relevant statutory provisions (and which version of such provision applies in case of a change in law) with regard to the specific contract and the underlying intention of the parties. It will need to have the capability to assess, for example, whether and to what extent the parties want to deviate from non-mandatory provisions or have these read into the contract. Depending on the individual circumstances and the legal system, this task may be complex because the software will need to interpret the contract and statutory provisions according to relevant rules of interpretation (taking into account, as required, relevant court decisions, circumstances surrounding the entry into the contract, etc.). The software will also need to understand that the parties will often not refer expressly to legal provisions they deviate from and that they may deviate by implicit contractual arrangement only.

1.3 Enforceable nature of a smart contract

As stated above, a smart contract must also be enforceable in some way. The two possibilities generally considered are either by way of enforcement using powers and tools provided for as a matter of the law of a particular jurisdiction ("legal enforcement") or by the so-called tamper-proof execution of computer code ("practical enforcement"):

> *"Enforceable either by legal enforcement of rights and obligations or via tamper-proof execution of computer code"*[13]

This distinction goes to the heart of the difference between a smart legal contract on the one hand and smart contract

13 See *supra*, note 5

code on the other, discussed at section 1.1.2 above.

A (smart) legal contract should be capable of being legally enforced in a court of law – in other words, a court would recognise that the agreement between the parties is legally binding and, as such, may enforce parties' legal rights through certain means. For example, a lender may ask the court to enforce its right in the (smart) legal contract to repayment by the borrower. In turn, the court may order the borrower to make the payment to the lender and the court or other real-world authorities may exercise powers to compel this to happen, for example, by seizing assets of the borrower.

However, a court would not have jurisdiction to exercise such powers where there is no legally binding agreement between the parties. In the case of smart contract code that does not constitute a legally binding agreement, parties may seek to use tamper-proof[14] technology to practically enforce the contract code instead. In particular, the parties may rely on the tamper-proof technology controlling the actions of the smart contract code so that the actions occur automatically, thus practically enforcing the performance of the contract code.

This form of practical enforcement may be achievable where all necessary actions can take place entirely within the software or technology environment. For example, smart contract code could be used to automatically transfer ownership of dematerialised securities by updating a distributed ledger that is recognised as the record of ownership of those securities.

However, practical enforcement via tamper-proof code becomes more difficult where the software or technology needs to interact with the real world, such as where delivery of a physical asset is required. Taking the sale of a new car as an illustration, if a smart contract were used to update the manufacturer's car ownership register to reflect the sale, use of tamper-proof software would not by itself be sufficient to practically enforce the transfer of ownership to the purchaser. In addition to an agreement between the parties on the transfer of ownership, practical real-world steps (such as physical delivery) may also be needed. If a dispute arose, an enforceable legal contract would generally be required in order for a court or other real-world authority to legally enforce the transfer and delivery of the car to the purchaser.

As noted at section 1.4 below, efforts are being made in a number of areas to "tokenise" real-world assets (such as real estate or company shares) whereby the transfer of a token recorded on a distributed ledger would be effective to transfer legal title to the underlying real-world asset. It therefore follows that a legally binding smart contract should be required to effect such a transfer.

1.4 Smart contract models

As noted above, the term "smart contracts" covers a range of different models, because the text of the contract may or may not be drafted in computer code (i.e., a programming language) and the software may have various roles in concluding, performing and/or enforcing (parts of) a contract by itself.

Integrated and Non-Integrated Models

In our analysis we refer to the following potential models by which smart contracts may be **written**:

- The contract itself could (in whole or in part) be written in a programming language (the "**Integrated Model**"). The software used to automate the contractual clauses may take a role in concluding, performing and/or enforcing (the whole or parts of) the contract. In this model, computer code would form an integral and binding part of the contract – i.e., some or all of the parties' rights and obligations would be expressed in a programming language rather than a natural language.

  For example, an agreement between Party A and Party B that Party A will transfer an asset to Party B at 10.00 am, might be agreed in computer code as follows:[15]

  If T = 10.00 CET, Execute Function:AssetTransfer X (Y to Z).

  T = Time

  X = Asset

  Y = Party A

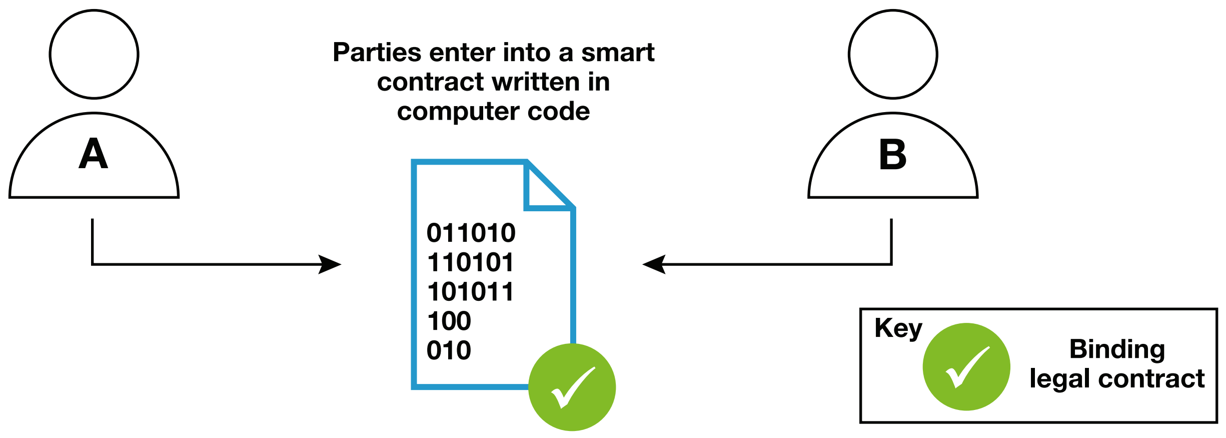
  Z = Party B

  This is illustrated in Figure 4a below, where the smart contract itself is written in computer code (and converted into binary code, in order to be read and executed by the computer).

14 Here tamper-proof refers to software which runs in a way which is unstoppable – either due to deliberate or accidental acts.

15 The "computer code" is not based on any specific programming language and would ultimately need to be converted or "compiled" into machine code (binary) in order for it to be executable by computer.

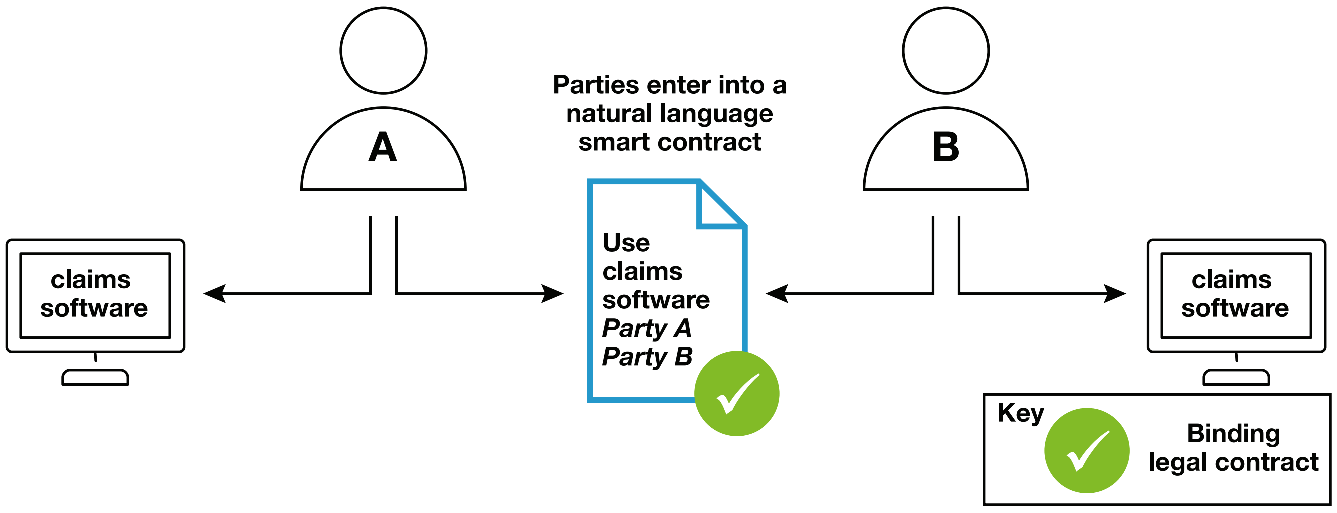
Figure 4a: Integrated Model


Parties enter into a smart contract written in computer code
A
B
011010
110101
101011
100
010
Key
Binding legal contract

- As illustrated in Figure 4b below, a smart contract could also be drafted exclusively in natural language but include an agreement between the parties to use specific software to perform and/or enforce (the whole or parts of) the contract or to conclude contracts (the "**Non-Integrated Model**"). For example, a company and an insurer may agree to use the insurer's automated claims-handling software. The software would review and determine claims made by the company and might even automatically instruct payments to be made in respect of determined claims. Under the Non-Integrated Model, this software or code would not itself form part of the contract.

In practice, it may not always be clear whether software or computer code is merely referred to in a contract (i.e., the Non-Integrated Model) or whether it is legally incorporated into and forms part of a contract (i.e., the Integrated Model). In such cases, this question may need to be determined based on the individual fact pattern and according to relevant legal principles (see section 2.1.1 for further discussion of this issue).

Figure 4b: Non-Integrated Model



Concluding and Performance Models

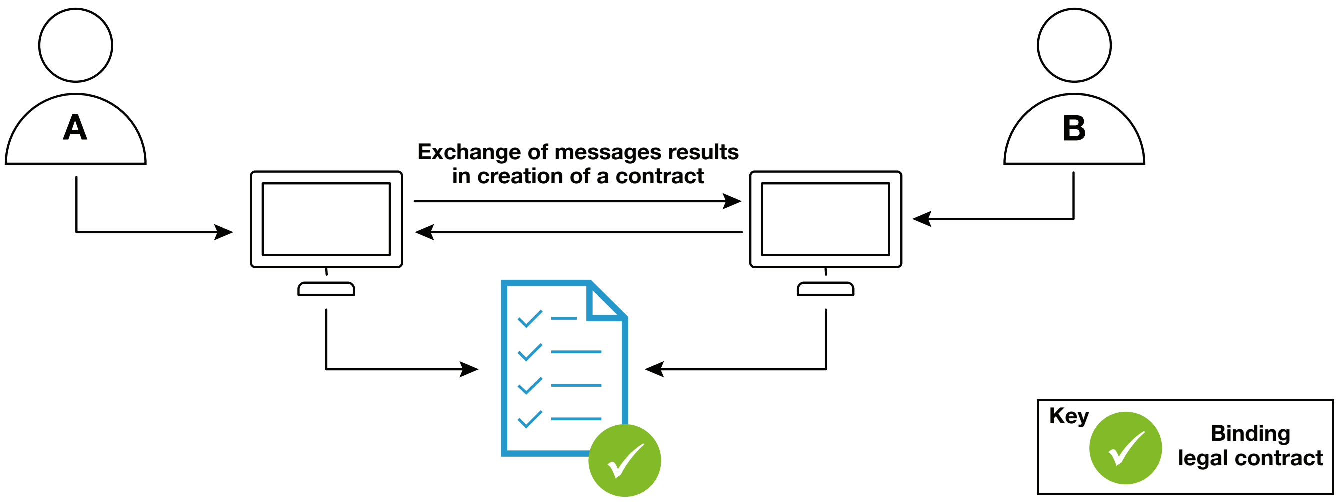
When we consider the ways in which a contract may be **concluded**, **performed** and/or enforced, we also differentiate between two (non-exclusive) models:

- The contract itself could be concluded directly and autonomously by software with or without prior separate agreement of the parties (the "**Concluding Model**"). In other words, the software may form or create legally binding obligations between the parties. For example, one computer running the software on behalf of a party might send an offer to sell securities to another computer running the software on behalf of another party which may autonomously accept the offer to purchase those securities. This is illustrated by Figure 4c below.
- In addition or instead, the contract could be performed and/or enforced by automated software (the "**Performance Model**"). Taking our securities trading example above, upon conclusion of the agreement between the parties, those computers might send instructions elsewhere to effect the agreed sale/purchase. If the two computers were nodes operating a distributed ledger which contains the definitive record of ownership of an asset, the software running on the two computers (or nodes) may be able to perform the transfer of the asset by making the relevant entry on the ledger.

  Whilst this may be easier to conceive in a context where the relevant "asset" is entirely native to a DLT environment (such as cryptocurrencies, tokens or coins)[16], there are advanced efforts in the marketplace to achieve similar results for other assets. This includes tokenised "real world" assets such as money, securities, real estate, etc.[17] (such that a transfer of a token represented by ledger records is effective to transfer legal title) and other methods not involving the tokenisation of any particular asset at all but using the ledger as a record of ownership, perhaps in place of custodial records or a property registry[18].

  This is illustrated in Figure 4d below, where the smart contract software automatically generates instructions based on pre-programmed logic. In this example, the software sends an electronic instruction to Party A's bank to make a real-world transfer from Party A's bank account to Party B's bank account (e.g. to pay for the securities referred to in the above example).

Figure 4c: Concluding Model



16 The term "native" here refers to assets, instruments or rights which exist only on the blockchain (i.e., in digital form).

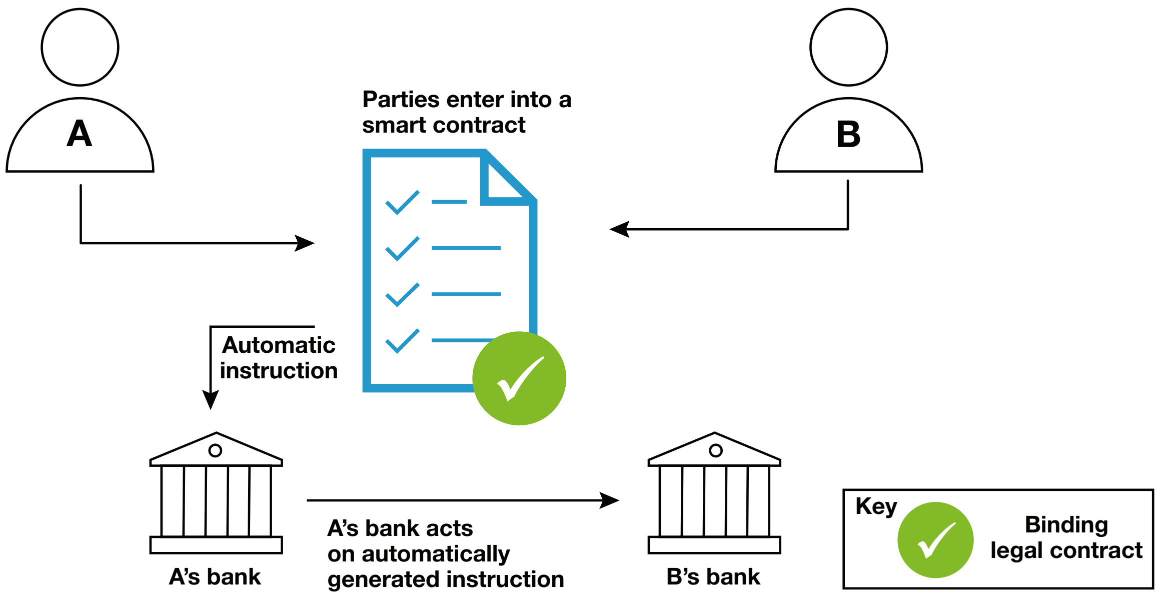
17 For example, see "*Commerzbank, KfW and MEAG simulate security transaction via Blockchain*", available at: https://www.commerzbank.de/en/hauptnavigation/presse/pressemitteilungen/archiv1/2017/quartal_17_03/presse_archiv_detail_17_03_68938.html

18 For example, see "*Credit Suisse and ING execute first live transaction using HQLAx securities lending app on R3's Corda blockchain platform*", available at: https://www.credit-suisse.com/corporate/en/articles/media-releases/cs-and-ing-execute-first-live-transaction-201803.html

The Performance Model and the Concluding Model may each be combined with the Integrated Model or the Non-Integrated Model and can be combined with one another.

Each model has its own particular legal and technological challenges. It will depend on the status and development of the technical and the legal environment as to whether smart contracts are, or may be, written in natural language with certain automated aspects (e.g., delivery or payment) only, or whether they could also be or become entirely self-concluded, self-performed and self-executed contracts.

Figure 4d: Performance Model


A
Parties enter into a smart contract
B
Automatic instruction
A's bank
A's bank acts on automatically generated instruction
B's bank
Key
Binding legal contract

2. Analysis

With that background, we move to the analysis of smart contracts. Since most of the countries in which the EBRD operates have a civil law system, the analysis is based on concepts of civil law and illustrated, where appropriate, with examples. The analysis covers the following areas:

- Existence and fundamental elements of valid and binding contracts;
- Challenges to the validity and binding nature of smart contracts including:
  - legal capacity;
  - authority requirements;
  - identification and verification of identity;
  - laws on "general terms and conditions" and other mandatory or non-mandatory laws; and
  - form and public registration requirements;
- Addressing deficiencies and mistakes in smart contracts, including allocation of liability;
- Amendments to smarts contracts;
- Governing law and jurisdiction;
- Dispute resolution;
- Confidentiality and data protection; and
- Anti-money laundering ("**AML**"), counter-terrorism financing ("**CTF**") and know your customer ("**KYC**") requirements and anti-bribery and corruption laws.

The analysis is accompanied by concrete recommendations and should serve as a useful roadmap for lawmakers in their attempt to navigate this nascent field and create an environment that facilitates the use of smart contracts.

Notably, the analysis cannot sensibly consider all issues relevant to smart contracts, so this paper does not cover considerations such as non-legal (i.e., practical) and regulatory implications. Nor does this paper consider issues relating to privacy and data protection (in detail), telemedia, IT, cyber-security, liability, insurance, tax, intellectual property, penal, competition, constitutional and human rights laws, or questions of legal ethics and legal philosophy (for example, arguments from an ethical or philosophical perspective as to whether and to what extent one should recognise the legal capacity of software). All of these are worthy of further exploration, of course.

As discussed above, smart contracts are likely to use DLT, particularly in the financial services space (although it is not absolutely necessary that they should) and, hence, we refer to DLT in this paper. DLT, of course, raises many interesting legal questions in its own right which, whilst we do not seek to address them directly, may add further considerations to the smart contract-related questions that this paper seeks to address.

2.1 Existence of valid and binding contracts

A fundamental question that lawmakers will need to consider is whether smart contracts are valid and binding contracts which give rise to legal obligations for the contracting parties.

2.1.1 Fundamental elements of a binding contract

Every legal system has its own applicable rules for what makes a contract valid and binding. In each jurisdiction, this question will need to be assessed in the context of the applicable legal rules and each legal system will establish its own minimum requirements. Notwithstanding unique questions in each legal system, many aspects and approaches will be quite similar or comparable.

Using German law as an example of a civil law jurisdiction, a contract is formed by an "agreement" of the contracting parties on the "essential terms".[19] The "essential terms" (the parties and their performance obligations) must be determined or be sufficiently determinable (from the individual circumstances or from common practice or market standards). For example, in a car purchase, the "essential terms" might be the identity of the seller and the purchaser, the obligation of the seller to transfer to the purchaser the ownership rights in a specific car (along with the car itself) and the obligations of the purchaser to pay a specific purchase price and to accept the transfer by the seller.

Continuing with German law, the agreement is formed by "congruent unilateral declarations of intent".
A declaration of intent is the expression of a will to achieve a specific legal result (i.e., to establish, amend or terminate a legal relationship). The declarations establishing a contract are generally an offer by one party and an acceptance by the other. Both parties may make their declarations and, hence, form the agreement by signing or otherwise agreeing a contract which sets out the relevant obligations. Under certain conditions, parties may also "declare" their intention implicitly by taking certain actions or by refraining from acting or being "silent". The existence and content of each declaration (offer or acceptance) must be assessed from the perspective of its respective recipient. Parties can, of course, conclude contracts in person (or at least directly between themselves even if not physically present), be represented by third parties (a representative) or appoint third parties to deliver or receive their declarations (a messenger).

If a smart contract is based on a traditional agreement entered into in any conventional way (such as in person, on paper or by electronic means, i.e., we are not in the realms of the Concluding Model where software concludes the contract autonomously), the requirements for the conclusion of valid contracts would be assessed just as for any non-smart (i.e., traditional legal) contract. As discussed above, under many legal systems these requirements would be met if the essential terms of the particular type of contract had been sufficiently agreed.

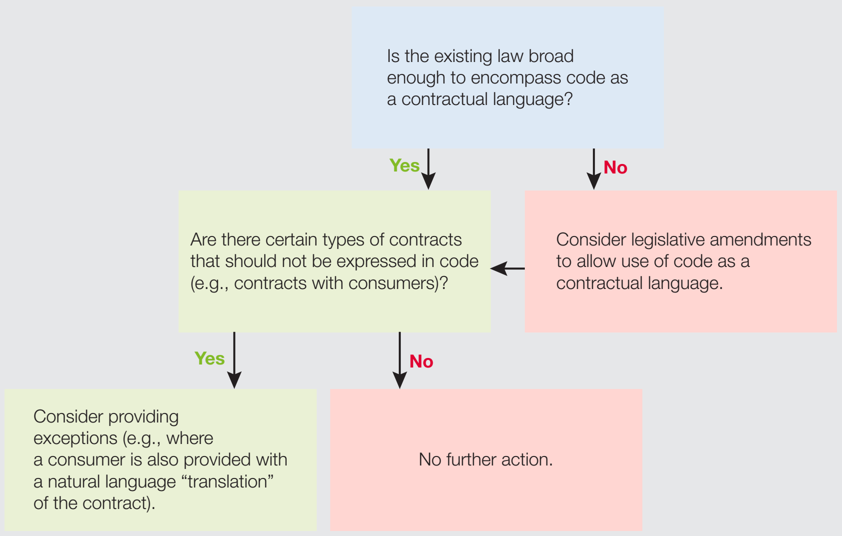
In the Non-Integrated Model, specifically, the contract is drafted in natural language just as any non-smart contract. The parties may agree, amongst other things, on the use of particular software for one or more specific purposes. This could be to perform an aspect of the contract or to enforce it in some way. The contract (or the offer and acceptance establishing it) will usually be evidenced by positive natural language statements to that effect given by each party. The rules applicable in the relevant legal system will determine

19 As in other civil law jurisdictions, the parties are free to determine the content and form of the contract within the boundaries of mandatory law. The agreement on all obligations and requirements other than the "essential terms" may be determined in line with applicable legal provisions. If the parties do not agree otherwise, non-mandatory statutory law applies.

Considerations for lawmakers

As a first step in determining whether a smart contract meets the requirements of a legally valid, binding and enforceable contract under the applicable law, lawmakers should ascertain whether the law is broad enough to encompass programming languages as a type of language in which a contract can be written. If the law does not allow for free choice of contract language or if the definition of "contract language" is not broad enough to include computer code, then relevant laws may have to be amended in order to expressly allow parties to enter into a contract written in computer code. An argument that computer code may not be considered easily readable or understandable and, therefore, not suitable for all or specific types of contracts, could be countered by the fact that parties may enter into legally enforceable contracts that are not written in their native language.

Lawmakers should also consider whether there are some types of contracts where it should not generally be acceptable to express the contract in a programming language. An example may be contracts with individual consumers who, by their nature, will not generally be able to read or understand computer code. Having identified such types of contracts, lawmakers should consider whether there may be exceptions where these contracts could be expressed in a programming language; for example, where a consumer is also provided with a natural language translation of the contract (in the same way that a translation might need to be provided if the contract were expressed in another foreign language).


Is the existing law broad enough to encompass code as a contractual language?
Yes
No
Are there certain types of contracts that should not be expressed in code (e.g., contracts with consumers)?
Consider legislative amendments to allow use of code as a contractual language.
Yes
No
Consider providing exceptions (e.g., where a consumer is also provided with a natural language "translation" of the contract).
No further action.

"To the extent the legal system allows for free choice of language, there may be good arguments to support the view that this freedom should not necessarily be limited to natural languages"

whether and the extent to which the agreement and its content must be determined solely from the wording of the agreement or whether the broader context must be taken into account when interpreting the contract.

In the Integrated Model, by contrast, the contract is drafted (in whole or in part) in a programming language. Hence, it must be possible to determine or evidence the "agreement" on the "essential terms" from the programming language contract or a combination of the natural and programming language parts of the contract.

Whether and the extent to which applicable law allows for the "agreement" to be expressed (or contractual provisions to be written) in a programming language will be driven by the relevant legal system (and sometimes by specific contexts of a legal system). There are many considerations which will influence whether this is acceptable. For example, computer code may be considered unsuitable either for all contracts or for specific types of contracts (such as contracts with consumers) on the basis that it is not easily readable or understandable. Applicable laws may go as far as to stipulate that any such terms are not binding on one or both parties. However, the same principle would apply to any contract in a natural language with which the parties are not familiar (such as a contract written in a foreign language that the contracting parties do not speak well or at all). To the extent the legal system allows for free choice of language, there may be good arguments to support the view that this freedom should not necessarily be limited to natural languages. Under many civil law systems it should generally be possible to draft provisions in a programming language. However, there may be exceptions; see sub-section 2.1.5 below, which discusses the treatment of "general terms and conditions" as a particular class of contractual clauses.

As a practical matter, the contracting parties could translate some or all provisions from programming language into natural language, if they do not understand the programming language. However, if they agree on the translation as being binding, this may amount to a Non-Integrated Model (i.e., where the contract itself is expressed in natural language and merely refers to software or code to be used in the conclusion, performance or enforcement of the contract). Depending on the rules of the relevant legal system, even a non-binding translation for convenience or ease of understanding might be taken into consideration when interpreting the programming language or when dealing with deficiencies in the parties' understanding of the programming language.

2.1.2 Capacity

As a general matter, the parties to a contract must have legal capacity to enter into the contract; otherwise, the contract will not be valid, binding or legally enforceable. Every legal system has rules governing the capacity (or general powers) of individuals and legal persons, such as companies, to enter into contracts or to issue notices in connection with contracts.

In the Concluding Model, parties may use software to automatically conclude new contracts by issuing an offer or acceptance. The software might also generate other legal notices or perform certain other legal actions in accordance with the contract (in the case of the Concluding Model or the Performance Model). By way of practical example, software could automatically request a loan and thereby conclude a loan agreement if there is a consenting lender who accepts the request for the loan. A smart contract might also automatically terminate a commercial real estate lease if the rent is not paid on time, by issuing the relevant termination notice to the tenant.

These are actions that the parties to a traditional (non-smart) contract might typically take themselves. Therefore, this raises a question as to whether the software itself might be seen as a party to a smart contract under the Concluding Model or act as the representative or agent of an underlying principal and, if so, whether it would have the legal capacity to do so.

At present, the answer to both of these questions is likely to be "no", although see "*Future thinking – could software become a new type of legal person*?" at page 20 for a discussion of some of the legal issues that lawmakers would need to consider if granting software the legal capacity to become a party to a contract itself. Instead, it is more likely that the actions of software or a machine to conclude a smart contract (or to issue notices in connection with the contract) would be best characterised as communicating on behalf of a contracting party.

Automated software used in the Concluding Model would thus be understood as a form of messenger that the contracting parties use to communicate with each other or with third parties. Such a role may not generally require the software or machine to have its own legal capacity to enter into contracts (although there may be some circumstances in which legal rules require a messenger to have legal personality, which the software or machine may not have[20]).

In this way, the software could deliver an offer, acceptance or notice which has been issued by a contracting party. For example, a potential borrower could programme software to automatically issue a loan request when his/her bank balance reaches a certain threshold (a paradigm example of IFTTT). Similarly, the software could be coded so that the loan request is automatically accepted, provided that certain pre-programmed conditions are met (such as the loan not

20 For example, a court summons may need to be delivered in person.

exceeding a certain size). This exchange of communications could give rise to a binding agreement between the contracting parties (here the borrower and the lender), even though the messages are sent automatically and without further input from the parties.

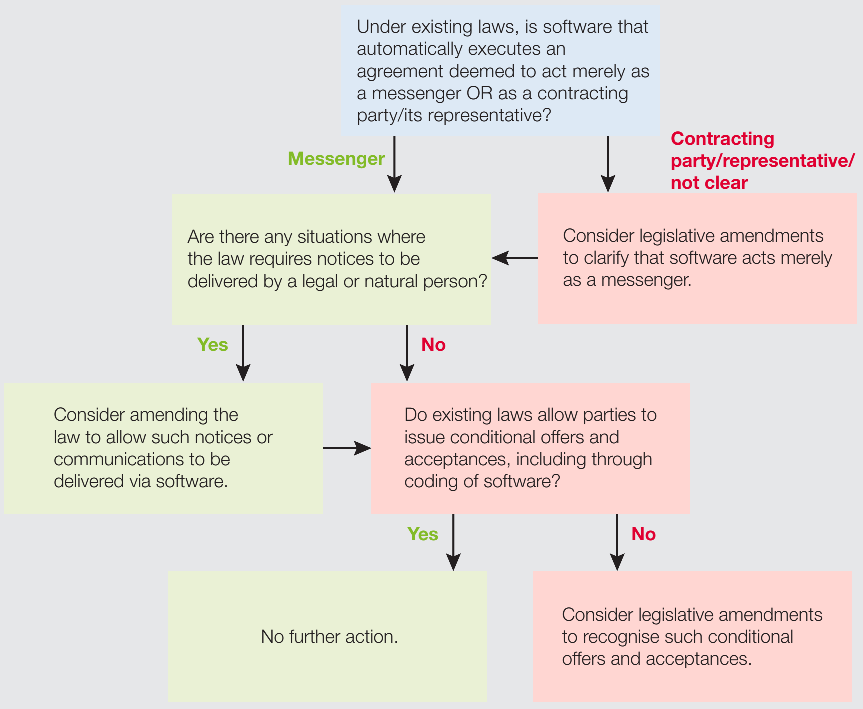
If the software is to be understood as a mere messenger in this situation (and not as a party to the contract or the agent of such party), the relevant legal system may therefore characterise the parties' actions when coding the software as the issue of a conditional offer and acceptance. In the above example, the borrower would be characterised as making a conditional offer for a loan by coding the software to automatically request the loan when the relevant condition is met (i.e., when his/her bank balance reaches the relevant threshold). Similarly, the lender would make its conditional acceptance of the offer by coding the software to automatically accept any loan request that meets the pre-programmed conditions. We anticipate that most legal systems will provide for parties to be able to issue a conditional offer, acceptance or notice in advance of the relevant condition being met, including via automated software, provided that the content of such offer, acceptance or notice is sufficiently determined or determinable when issued.

Considerations for lawmakers

As a first step, lawmakers should determine whether, under existing laws, software that automatically executes an agreement (if certain conditions are met) is deemed to act merely as a messenger of the relevant contracting parties, or whether it might be characterised as a contracting party itself (or as its representative or agent). If existing laws do not provide a clear answer, lawmakers may wish to consider amending applicable laws to clarify that software acts merely as a messenger or medium of communication in this context.

Lawmakers should also assess whether existing laws allow for contracting parties to issue conditional offers and acceptances, including via automated software (as would often be the case where parties enter into smart contracts under the Concluding Model). If not, lawmakers may wish to consider amending existing laws to facilitate the use of smart contracts under the Concluding Model, for example to recognise the actions of the parties in coding software to automatically conclude a contract where relevant conditions are met, as the issue of a conditional offer or acceptance.

Finally, lawmakers should identify whether there are any situations in which the law requires notices (or other communications relating to contracts) to be issued by a legal or natural person. If this is the case, lawmakers should consider whether it would be appropriate to amend existing laws to allow such notices or communications to be delivered via software, in the context of a smart contract.


Under existing laws, is software that automatically executes an agreement deemed to act merely as a messenger OR as a contracting party/its representative?
Messenger
Contracting party/representative/not clear
Are there any situations where the law requires notices to be delivered by a legal or natural person?
Consider legislative amendments to clarify that software acts merely as a messenger.
Yes
No
Consider amending the law to allow such notices or communications to be delivered via software.
Do existing laws allow parties to issue conditional offers and acceptances, including through coding of software?
Yes
No
No further action.
Consider legislative amendments to recognise such conditional offers and acceptances.

Future thinking – could software become a new type of legal person?

Under existing laws, the role of automated software in concluding contracts might be most easily characterised that of *a messenger or medium of communication* by which an offer, acceptance or notice may be communicated between the parties. However, it is also conceivable that the software or machine might *contract on behalf of, or legally bind, a contracting party* (for example, as agent) or, perhaps more interestingly, even *enter into an agreement as contracting party itself* (as principal).

At present, there does not seem to be a great practical need for software to take on these other types of roles. However, in the longer term and as smart contracts become more sophisticated, it may be beneficial or even necessary to provide for software itself to be a "party" to a contract. For example, under the Concluding Model, it may become more difficult to characterise software as a mere messenger where the software uses AI or machine learning (rather than conditions that are expressly pre-programmed by the parties) to determine whether, when and on what terms to issue offers and acceptances. Treating software as a "party" to a smart contract could also help resolve questions relating to liability allocation where there is an error in contract code developed by a third party, as it may open up the possibility for the software itself to be held liable for the error.

There are various considerations that lawmakers would need to take into account. Firstly, lawmakers might seek to amend applicable laws to create a new type of legal person, for example by establishing the concept of an "electronic person", which could have capacity to enter into a contract and/or have capacity to act as the agent or legal representative of a contracting party (much in the same way as a company has legal personality and capacity to enter into contracts or to act as a representative or agent of a contracting party).

Lawmakers should consider whether the capacity of software to enter into a contract should be limited in some way, for example, by requiring the consent of the person appointing the electronic person, or of all persons using the software.

Lawmakers should also consider whether the creation of legal personality means that legal responsibility and liability would therefore rest with the software itself, as a non-natural legal person. If so, consideration should also be given to whether the software needs to be able to own assets (and possibly hold some minimum level of assets) in case of incurring a liability. Furthermore, rules would need to be developed on when the software, as a legal person, would have capacity and authority to enter into contracts and on how to evidence such authority. Existing rules for companies will generally deal with these types of issues and so they are likely to be instructive in this regard.

2.1.3 Authority requirements

Having established that the parties to a contract have legal capacity to enter into the contract, it is also necessary to consider whether the persons (or software, if not acting as a mere messenger – see section 2.1.2 above) entering into the transaction or performing the relevant action have the authority to do so.

In general, this will depend on the factual situation. A natural person entering into a contract as principal (i.e., not acting on behalf of anyone else) would not usually need any particular authority to do so. However, a person may not be able to enter into an agreement on behalf of a natural person or a company unless he/she has been authorised to do so (for example, by the board of directors, in the case of a company). Such authority may be general or specific to particular acts, and it may result from laws, corporate constitutional documents, contracts, etc. The effects of a lack of authority or (non-) approval may vary between legal systems or differ depending on specific circumstances within a legal system. For instance, a number of jurisdictions provide some protection for a party to a contract where the other party held themselves out as having authority to enter into a contract, even though they did not actually have authority to do so. This is sometimes referred to as "**ostensible authority**".

Requirements relating to authority are not specific to smart contracts, but smart contracts may have to ensure that these requirements can be met and verified as appropriate within automated processes. This may mean identifying those processes that require approval or authority and potentially seeking such approval or authority in advance or ensuring that the approval or authority can be confirmed or obtained through the use of interfaces or oracles that allow software to connect to external systems and data sources.

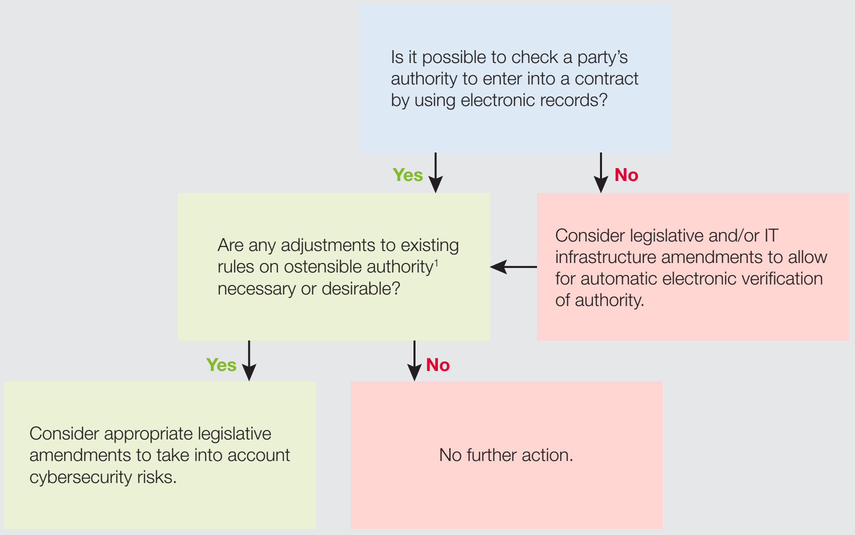
Considerations for lawmakers

In many instances parties will wish to determine whether their counterparty has the necessary authority and/or has obtained the correct approvals to enter into a contract and/or perform certain actions. This is not unique to smart contracts and it is common for parties to carry out this type of due diligence before entering into any type of contract. However, in the context of smart contracts, parties may seek to carry out such due diligence in an electronic or automated manner.

Therefore, as a first step, lawmakers should check whether, under existing laws, such determination is permitted to be made using electronic records (for example, by reference to a central company registry or to credit reference agencies). If not, lawmakers may need to consider whether changes should be made to existing laws to allow for automated electronic verification of relevant approvals and authority.

Lawmakers will also need to consider whether the necessary infrastructure is in place to allow for electronic verification of authority and approvals in practice – i.e., whether an electronic public register of directors or of board resolutions exists, which could be accessed and read by the DLT software. If not, lawmakers may need to consider whether to create or encourage creation of such electronic registers.

Lawmakers should also consider whether any adjustments to existing rules or legal principles relating to ostensible authority may be appropriate in the context of smart contracts. In some cases, the existence and use of electronic registers may make it harder for a person to falsely hold themselves out as having authority to enter into a contract on behalf of a company or other entity. On the other hand, it is more likely that smart contracts will be entered into remotely and so other risks, such as hacking or digital identity theft, might be higher.


Is it possible to check a party's authority to enter into a contract by using electronic records?
Yes
No
Are any adjustments to existing rules on ostensible authority[1] necessary or desirable?
Consider legislative and/or IT infrastructure amendments to allow for automatic electronic verification of authority.
Yes
No
Consider appropriate legislative amendments to take into account cybersecurity risks.
No further action.

[1] Ostentible authority refers to a situation where a person appears to have authority to act, even though they do not have such authority.

For example, if a company director uses their private key to "sign" a DLT-based smart contract,[21] the software might need to check not only that the individual signing the contract really is a director of the company but also that the director has authority to enter into the contract on behalf of the company (e.g., a board resolution or the company's constitution) before executing the transaction.

Turning to consider what types of information or data sources may be needed to validate a person's authority, public registers may help (for example, a company register setting out, conclusively or non-conclusively, relevant data on a company and its authorised representatives). In order to automate this process, the software would need to be able to access, read and consider the data set out in such registers (to establish, for example, whether a specific person is authorised to enter into a specific (type of) transaction on behalf of a company, on its own or jointly with certain other persons). However, this access to public registers may not provide all the data needed to assess capacity and authority, as authority is often delegated to authorised signatories via "simple" powers of attorney or board resolutions which are not necessarily publicly filed or disclosed. The parties to the smart contract might therefore agree on private data sources, such as a database listing all relevant signatories of one or both parties, being made accessible for the software to parse. Assuming this is possible, it could represent a significant step towards automating currently manual processes.

At one end of the spectrum, if sufficiently intelligent, the software might be able to access and parse a board resolution or a list of directors itself but, at the other end, the software may simply seek an input confirming (perhaps in legal terms "representing") that the resolution was duly passed or that the person(s) "signing" the smart contract and purporting to be directors, are in fact directors of the company and have authority to bind it.

Within the boundaries of the applicable legal framework, the parties may themselves determine the required or desired level of assurance to be built into the software. This level could be driven by commercial considerations, risk and/or statutory requirements. For example, it might be a statutory requirement that the board of a public company approve transactions over a certain value and so the parties may decide to build this condition into the software expressly.

2.1.4 Identification and verification

One practical challenge that may arise when using DLT is how to identify the other contracting party, if the distributed ledger enables anonymous or pseudonymous transactions. This may be the case where transactions are recorded by reference to an IP address or wallet address.

Requirements for identification and verification of identity

Depending on the jurisdiction and the type of smart contract, the parties may want, or may be required, to identify and verify the identity and signature of the other contracting parties, their representatives and other relevant persons or entities. For instance, this may be required in order for parties to satisfy applicable AML, CTF and KYC requirements (see section 2.8 below). Parties may also want to identify relevant persons or entities for internal risk purposes.

The consequences of non-compliance with identification and verification requirements will depend on the relevant legal system. In general, such non-compliance may give rise to a legal or regulatory breach and risk of associated enforcement proceedings and sanctions. A failure to identify or verify the identity or signature of relevant persons could also call into question their capacity and/or authority and so, under some legal systems, it could impact the legal, valid and binding nature of the contract itself.

Conversely, a legal system may determine that non-identification of the other contracting party does not affect the existence of a binding and valid contract, at least to the extent that the identities of the contracting parties are sufficiently determined or determinable (when using DLT, this could be by means of the contracting parties' IP or wallet addresses or their cryptographic keys).

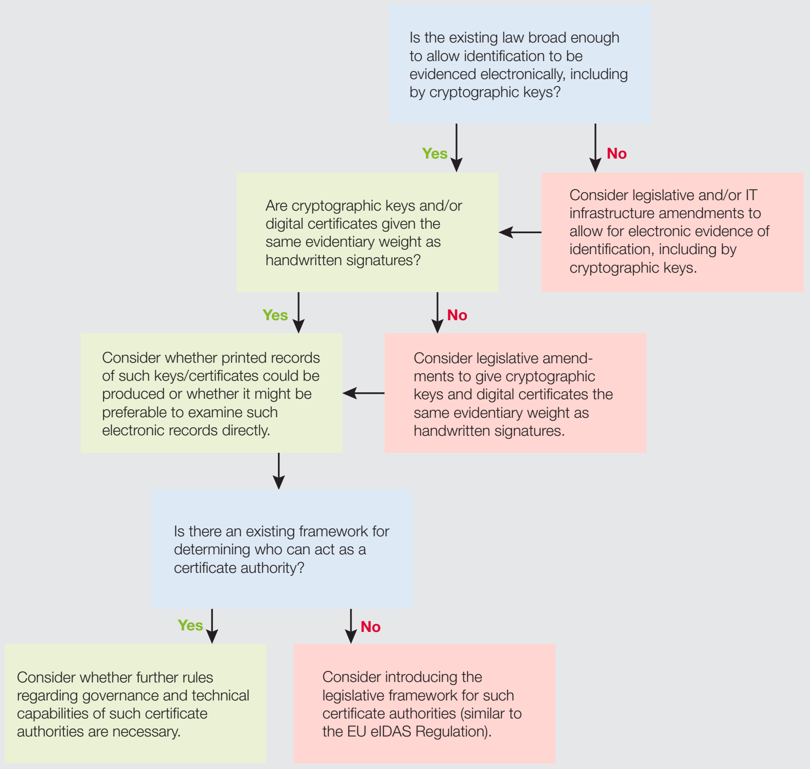
21 Within a DLT framework, two keys are assigned to each participant; a public and a private key. Each participant's public key uniquely matches their private key. While the public key is generally available and contains information about the participant (such as the account number, etc.), the private key is necessary to digitally sign the transaction. Therefore, only a person that has the private key matching the participant's public key can carry out a transaction via DLT. Based on the public key and its corresponding (hash) values, transactions, contractual parties and/or machines can be clearly identified within a DLT network at any time.

Considerations for lawmakers

As a first step, lawmakers should determine what sort of evidence (if any) courts would accept under the existing legal framework in support of a claim that the smart contract has been properly entered into by the person indicated by the smart contract. In particular, they should consider whether evidence of identification by electronic means (such as cryptographic keys) is permissible in principle. If not, existing laws may need to be amended to expressly permit such evidence to be considered by the courts.

In addition, lawmakers will need to determine the weight that should be given to such evidence by the court and consider how such evidence may be presented in practice. For example, they should consider whether cryptographic keys or digital certificates should be treated as equivalent to the use of a wet-ink handwritten signature and in what circumstances such evidence might be challenged or overridden. Practical considerations include whether printed records of such cryptographic keys or digital certificates could be produced or whether it might be preferable (or even necessary in some cases) to examine such electronic records directly or rely on testimony of the parties or a witness who has examined such records.

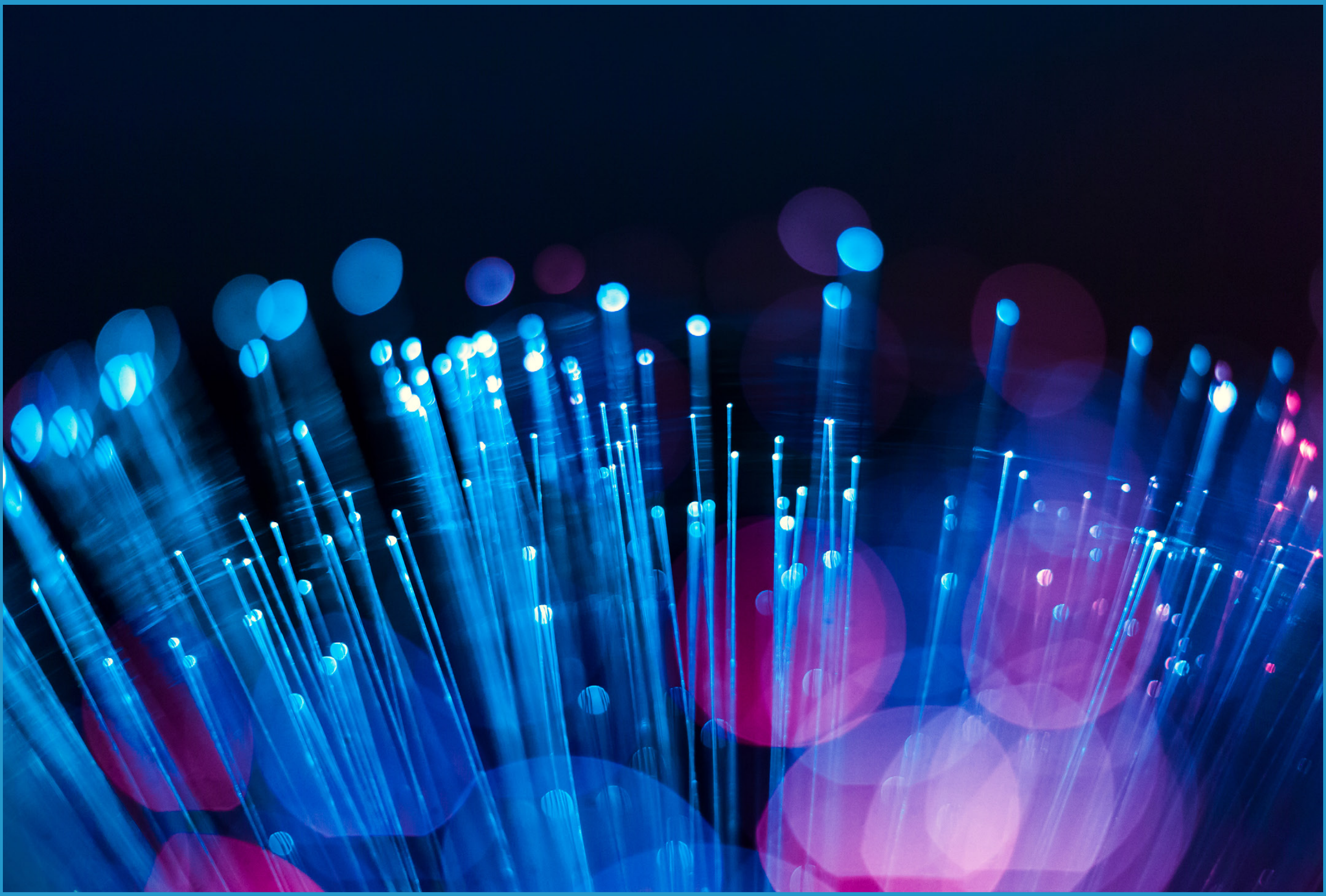
When determining whether and how such evidence should be accepted and the weight it should carry, lawmakers should consider whether there is an existing framework for determining who can act as a certificate authority.[22] A cryptographic key or digital certificate may carry more evidential weight if it is provided or issued by a certified provider or authority whose credentials can be checked. If there is no such existing framework, lawmakers will need to create rules governing qualification and the technical assurance that such bodies must provide.


Is the existing law broad enough to allow identification to be evidenced electronically, including by cryptographic keys?
Yes
No
Are cryptographic keys and/or digital certificates given the same evidentiary weight as handwritten signatures?
Consider legislative and/or IT infrastructure amendments to allow for electronic evidence of identification, including by cryptographic keys.
Yes
No
Consider whether printed records of such keys/certificates could be produced or whether it might be preferable to examine such electronic records directly.
Consider legislative amendments to give cryptographic keys and digital certificates the same evidentiary weight as handwritten signatures.
Is there an existing framework for determining who can act as a certificate authority?
Yes
No
Consider whether further rules regarding governance and technical capabilities of such certificate authorities are necessary.
Consider introducing the legislative framework for such certificate authorities (similar to the EU eIDAS Regulation).

22 For example, the eIDAS Regulation sets out such a framework. It applies to "trust service providers" (broadly, entities that provide, verify, validate or preserve electronic signatures, seals, time stamps or website authentication certificates). These providers must meet various requirements under the eIDAS Regulation in order for their services to be recognised or "qualified" under the eIDAS Regulation.

Certificate authorities for smart contracts

In the context of electronic transactions which are executed using public-private key cryptography, a certificate authority is a trusted third party that is able to certify the ownership of a particular public key. A certificate authority arrangement could be used, therefore, to certify that a public key used to enter into smart contracts belongs to a particular person (and, only that person would be able to "sign" the contract by applying their certified public key together with its corresponding private key). If applied to smart contracts, this technology could provide a degree of assurance as to the identity of the relevant parties to the smart contract. At present, the technology does, however, have some vulnerabilities; for example, because certificate authorities are not generally transparent about the certificates that they issue and so it can be hard to immediately detect instances where a certificate authority issues rogue certificates which could be used in targeted attacks.[23]



23 For further information on how certificate authorities might be adapted for smart contracts and how some of the drawbacks of the current model could be addressed, please see: "*SCPKI: A Smart Contract-based PKI and Identity System*", Mustafa Al-Bassam, University College London, available at: http://www0.cs.ucl.ac.uk/staff/M.AlBassam/publications/scpki-bcc17.pdf

Practical considerations for smart contracts

If smart contracts are concluded in conventional ways, such as in person and on paper (i.e., in a non-Concluding Model), identification and verification of parties to the contract can be carried out just as for any non-smart contract. However, even in such cases, certain automated actions following the conclusion of the contract may require separate identification and verification – for example, certain payments or instructions.

Identification and verification of identity may be more difficult under the Concluding Model where there is automated conclusion of contracts. In our view, the extent to which identification and verification of identity are possible in such a model will likely depend on the features of the software and the data sources available. In other words, this is a problem that is likely in search of a technology solution.

That said, specific procedures could be provided for by agreement of the parties or the terms of a distributed ledger. Generally, the means of electronic verification of identity under the applicable legal framework may provide a solution to some of these considerations. Within the European Union, Regulation (EU) No. 910/2014 ("**eIDAS Regulation**")[24] contains such a legal framework. As well as dealing with more general questions relating to electronic identification of persons, it provides that "qualified electronic signatures" are to be considered equivalent to handwritten signatures.[25]

Other suitable approaches to effectively link the digital identity of a person or legal entity with its real world identity may be developed. For example, given that participants in DLT-based systems "sign" transactions using a unique private key, one possibility is that these unique private keys could be recognised as a form of electronic signature. At the very least, it would validate that a person with access to the relevant private key "signed" the transaction but, beyond this, if each authorised signatory were assigned their own unique private key, parties may use these keys as a proxy for signatures of authorised signatories. In this way, manual specimen signature lists might be eliminated and signature verification automated.

Some analysis would be required as to whether the use of cryptographic keys to "sign" DLT-based transactions would also meet the requirements to be a "qualified electronic signature" under the eIDAS Regulation. This would generally be a question of fact, depending on the characteristics of the DLT system in question and the way in which cryptographic keys are used under that system.

Use of third parties

The contracting parties may also utilise or require a third party (such as a certificate authority or the operator of a digital register, if one exists) to provide a means of identification, verification and validation of capacity and authority, where this is permissible, possible and practical (for AML/CTF/KYC considerations specifically, see section 2.8 below).

2.1.5 Smart contracts as "general terms and conditions"

Most jurisdictions have specific laws relating to the use of "general terms and conditions" or "standard (business) terms". In the context of smart contracts, it is necessary to consider firstly when a smart contract may be characterised as general terms and conditions.

Existing legal rules and principles governing general terms and conditions may already be broad enough to capture smart contracts as general terms and conditions. In many jurisdictions, the trigger will be whether the smart contract terms are pre-formulated for a large number of contracts and/or where one contracting party deals with another using its non-negotiable standard agreement.[26] In this context, it is important to distinguish between standard contractual terms and standardisation of the software that may be used for smart contracts; use of standardised software should not necessarily mean that a smart contract qualifies as "general terms and conditions".

Smart contracts are less likely to be treated as general terms and conditions if they are sufficiently negotiated between the parties or if they have not been provided by one contracting party to the other. For instance, when using a public DLT platform, the platform terms could be regarded as having been provided to the parties by the platform's operator or

24 Regulation (EU) No. 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC, OJ (EU) L 257/73.

25 The eIDAS Regulation provides for several forms of electronic signature, where "qualified electronic signatures" are those electronic signatures which provide for the highest degree of reliability because they are created by a special electronic signature creation device which meets certain technical criteria and use digital certificates to certify the validity of the electronic signature (see "Certificate authorities for smart contracts" in this section 2.1.4).

26 Note that in some legal systems, contractual terms may qualify as general terms and conditions even if the user has not pre-formulated them himself and if he wants to use them only once.

core group of developers that are responsible for the code design (i.e., a third party or parties) rather than by one user of the platform to another. Nevertheless, a court would likely analyse this question carefully, based on existing legal principles for determining whether a contract should be treated as general terms and conditions.

It is also common for requirements relating to general terms and conditions to apply so as to protect consumers. Whilst in some jurisdictions, requirements relating to general terms and conditions also apply in business-to-business relationships, the requirements with respect to general terms and conditions provided by a business to consumers are typically stricter.

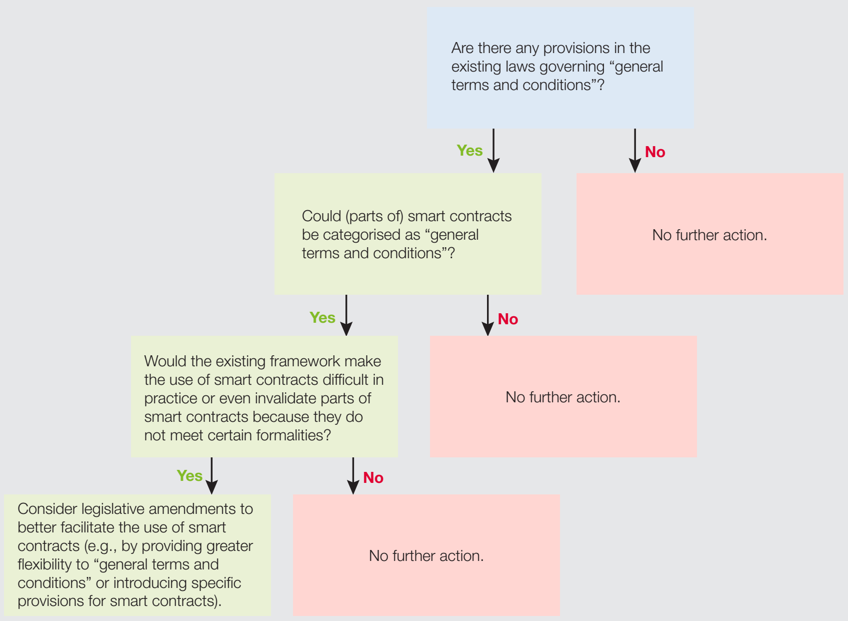
If existing rules would generally preclude use of smart contracts, or specific types of smart contracts, lawmakers may want to consider amending these existing rules to facilitate the use of smart contracts. When considering this, lawmakers will likely weigh the benefits of using smart contracts (such as the increased efficiency and reduction in risk of human error that automation could bring about) against effective protection of consumer rights, which is currently considered a high priority within and beyond the EU.

Considerations for lawmakers

As a first step, lawmakers should identify whether existing laws govern the use of general terms and conditions and their fairness, etc. If so, lawmakers should first consider whether (and, if so, when) smart contracts could be categorised as general terms and conditions according to the principles of the existing framework. Secondly, lawmakers should consider whether the existing framework would make the use of smart contracts difficult in practice, for example, because the terms and conditions must be delivered and/or presented in a particular medium or because terms and conditions written in a programming language might be considered "unclear" and therefore non-compliant with a requirement for the contract to be "fair, clear and not misleading".

If there are such requirements, lawmakers should consider whether those existing laws should be amended to better facilitate the use of smart contracts qualifying (in whole or in part) as general terms and conditions. This could take the form of amendments to provide for greater flexibility in the rules applicable to all types of general terms and conditions (i.e., for both smart and non-smart contracts). Alternatively, this could be done by developing specific conditions for use of smart contracts (for example, that a natural language translation must be provided for any part(s) of a consumer contract that are expressed in a programming language).

Lawmakers should also consider whether the existing laws governing the use of general terms and conditions could have the effect of partially invalidating the terms of a smart contract (for example, if some of its terms are considered to be unfair) and whether existing law would seek to "fill" the resultant gap. If this is the case, lawmakers may need to consider whether this is viable in a smart contract construction and whether alternative results might need to be provided for (see sections 2.3.4 and 2.3.6 for further discussion on this issue).


Are there any provisions in the existing laws governing "general terms and conditions"?
Yes
No
Could (parts of) smart contracts be categorised as "general terms and conditions"?
No further action.
Yes
No
Would the existing framework make the use of smart contracts difficult in practice or even invalidate parts of smart contracts because they do not meet certain formalities?
No further action.
Yes
No
Consider legislative amendments to better facilitate the use of smart contracts (e.g., by providing greater flexibility to "general terms and conditions" or introducing specific provisions for smart contracts).
No further action.

Secondly, it is helpful to consider whether there are any practical features of smart contracts that make it more difficult to comply with requirements applicable to general terms (or might otherwise justify special treatment of smart contracts). In general, the laws relating to general terms and conditions may take the form of mandatory terms or requirements (meaning it is not possible for the parties to deviate from such requirements). For example, some rules may prohibit the use of potentially onerous or unexpected terms or require them to be drawn to the attention of a consumer and/or require the user of the general terms and conditions to write and deliver them in a way which is fair, clear and not misleading. Practical challenges may arise in meeting these types of requirements, particularly for smart contracts using the Integrated Model, where the contract itself it written in computer code.

To the extent that a smart contract is characterised as "general terms and conditions" under the applicable law, clauses that violate mandatory legal requirements applicable to such general terms and conditions may be void and the resulting gaps in the contract may (or may not) then be filled in accordance with relevant provisions or principles of applicable law. Sections 2.3.4 and 2.3.6 below discuss the potential impact of void clauses and errors or gaps in smart contracts further.

2.1.6 Other applicable laws

More generally, smart contracts will need to comply with applicable mandatory provisions or principles of law. This may include not only those that exist at the time the contract is entered into, but sometimes also those that may be developed later. Some laws, such as sanctions laws, may also apply to existing contracts and, in some instances, law and regulation can have a retroactive effect. Therefore, parties may need the ability to amend the contract following a change in regulation to take that change into account and enable the parties to continue to use the smart contract. Smart contracts may therefore need to include mechanisms for making these sorts of ongoing adjustments. Failure to do so may result in the smart contract or clauses within the smart contract being rendered void. Whilst this is true of any contract, the automated nature of smart contracts means that there are particular practical issues to consider when seeking to amend smart contracts (see section 2.4 for further discussion on amending smart contracts).

As highlighted at section 2.1.5 above, mandatory contractual provisions or requirements often derive from consumer protection laws, but other laws can also be relevant. Many legal systems will include mandatory requirements relating to the performance of certain types of contract – i.e. requirements which the parties must adhere to if they want the law to give effect to or recognise the performance. For example, legal systems may contain laws setting out mandatory requirements relating to the enjoyment by a tenant of the relevant property, such as a requirement for the landlord to provide the tenant with the means of access to the property. If provisions in a smart contract do not comply with such requirements, they may be rendered unenforceable or void. In this example, a court might decide that a smart contract clause, which provides for a leased office to be automatically locked (by some software) in the event that the rent is not paid, is void on the basis that it unlawfully interferes with the tenant's legal right of possession.

Validity of contractual terms will always have to be assessed for each individual clause and legal relationship and taking into account all relevant circumstances. Where a contractual term is found to be invalid or void, the resulting gap may need be filled in line with the rules and doctrines applicable in the relevant legal system, as considered further at sections 2.3.4 and 2.3.6 below.

2.2 Requirements of form and public registration

Another important question for smart contracts is whether and how they can accommodate requirements relating to the form of the contract, or requirements for the contract or related actions to be publicly registered in some way.

2.2.1 Legal background

Many legal systems require certain types of contracts, notices, instructions, etc. to take a prescribed form or to comply with certain formalities. This may therefore limit the freedom of parties to conclude contracts, deliver notices or give instructions in any form including, for example, electronically or orally. The parties may also be allowed or mandated by law to agree such requirements between themselves.

The required formalities and the consequences if these requirements are not met will be particular to each legal system. Common requirements include that the contract must be in writing, must be delivered or stored in a particular medium and/or must be signed by one or more parties. These requirements may serve a multitude of purposes, for example, as evidence, information or to provide a warning. Under consumer protection laws, notifications to consumers often need to be provided in a form of text that is sufficient to inform or warn consumers.

Considerations for lawmakers

As regards DLT, as a first step, lawmakers must determine whether the relevant provisions under the existing laws are broad enough to implicitly allow records to be made and stored using DLT. If the relevant provisions are not broad enough to recognise such DLT-based records, lawmakers should consider whether modifications to existing laws are necessary to provide for this[27].

In some cases this may not be necessary, if the existing law does not expressly prohibit the use of such records or otherwise call into question their validity. However, in some instances, lawmakers may need to amend relevant laws to expressly acknowledge that a digital record electronically registered using DLT is to be considered a valid record, provided that it meets certain requirements. Lawmakers will need to stipulate the requirements and the purpose for which such record will be valid.

Legislation may also be required to specifically recognise or permit the use of DLT for a specific purpose. By way of example, France has introduced legislative reforms which propose the recognition of a DLT register and its records for the representation, transmission and pledge of unlisted securities at the discretion of the relevant issuing company[28]. Again, in other cases, it may not be necessary to specifically provide for such recognition to the extent that the relevant legislation establishing the register or its status at law is already sufficiently technology-neutral. Lawmakers will need to make an assessment about this.

In this context, lawmakers would also need to consider relevant issues relating to DLT-based registers, such as the level of security that should be required, the degree of public access and reliance which would be afforded if the register were available only in such form, as well as compatibility with data protection requirements.

As an additional step, lawmakers should consider whether to introduce requirements relating to the involvement of third parties (such as notaries, public authorities or courts) in the smart contracting process, taking into account that these third parties can play particular roles in a transaction, such as informing or warning the parties about certain implications of the transaction.
In this case, lawmakers should also seek to ensure that any such requirements are flexible enough to allow use of technology where appropriate; for example, by allowing a third party to validate a smart contract through application of a digital certificate (rather than by physically signing or applying a stamp or seal, as might be done for a traditional contract).

It will be incumbent on lawmakers to make a policy choice about whether and the extent to which changes should be made to existing requirements relating to form and/or third party involvement. Where the main purpose of a form requirement is to provide an evidentiary and/or transparency function, lawmakers might consider that use of a publicly accessible DLT-based register (or otherwise recording transactions on a public DLT platform) is deemed to satisfy this function. In this case, they may decide to amend existing legislation to dispense with the requirement or deem that it is satisfied through use of the ledger.
On the other hand, in circumstances where a form requirement is (mainly) there to provide some form of advisory or warning function, lawmakers may be more reluctant to relax such functions given their importance in ensuring that parties are appropriately protected (or, at least, not unduly at risk without knowledge or understanding).

Notably, real estate transactions may require agreements to be recorded by a notary and rights to be registered in the land register; whilst lawmakers may be more confident in establishing a DLT-based land register (provided that the register satisfies the transparency and evidentiary functions adequately and is able to maintain public trust), they are less likely to be comfortable with removing the requirement for the notary recording to the extent this currently also serves to warn the contracting parties and to provide them with advice.

27 See for example, the introduction of laws recognising DLT records in Vermont, available at: https://law.justia.com/codes/vermont/2016/title-12/chapter-81/section-1913; and in Delaware, available at: https://legiscan.com/DE/text/SB69/id/1627743

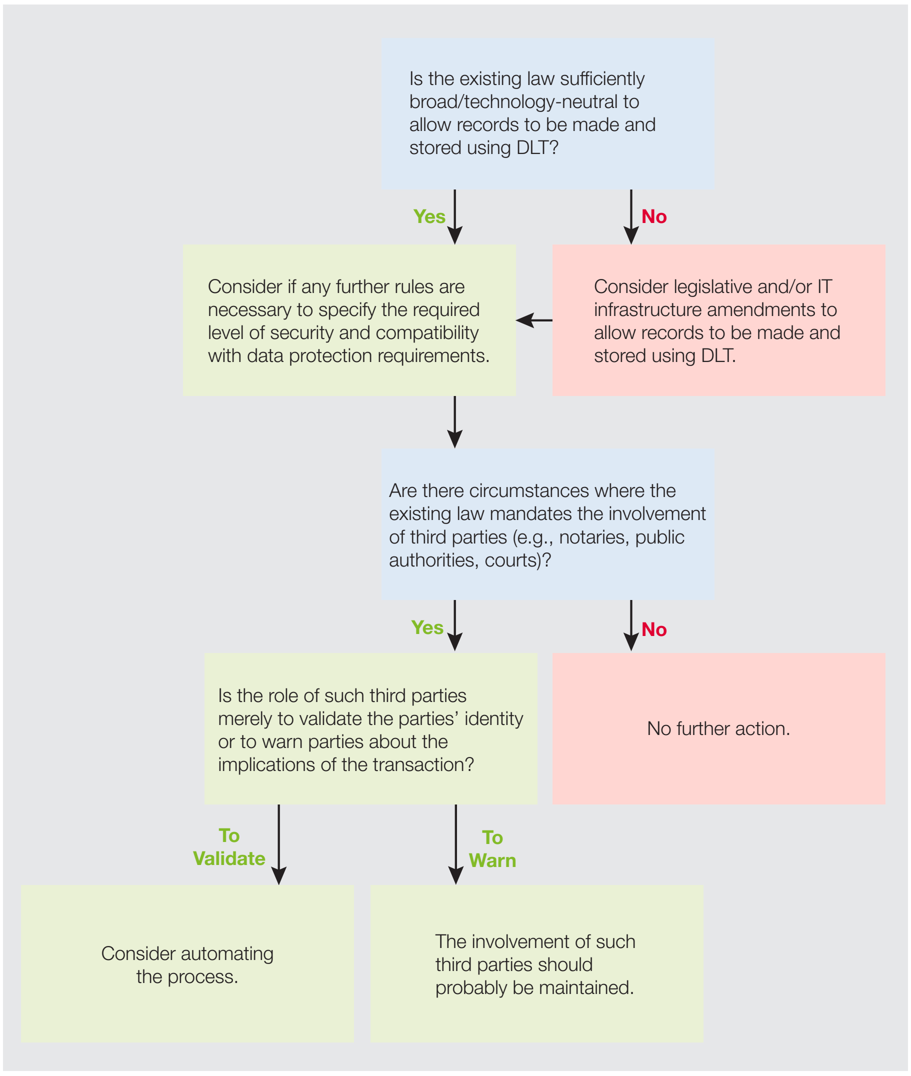
28 See https://www.legifrance.gouv.fr/affichTexte.do?cidTexte=JORFTEXT000036171908&categorieLien=id and "*France pioneers blockchain legal framework for unlisted securities*", available at: https://www.cliffordchance.com/content/dam/cliffordchance/PDFDocuments/Client%20Briefing%20-%20France%20-%20Blockchain%20for%20unlisted%20securities%20180750-4-2....pdf

In some cases, the involvement of third parties, such as a notary, may also be mandated. Whilst such requirements may serve to inform or warn the contracting parties of the impact of entering into the relevant agreement, the notary might also provide an advisory function for the parties to the transaction, answering specific questions they may have.

In many legal systems, these formalities include, or are supplemented by, public registration requirements, such as the requirement to register real estate-related transactions in a land register or company-related transactions in a company register.

2.2.2 Application to smart contracts
For smart contracts concluded by the parties in a conventional way (i.e., in a non-Concluding Model), it should generally be possible to satisfy relevant formalities in the same way as for a non-smart contract. This should be the case particularly for the Non-Integrated Model. However, the position may be different for contracts using the Integrated Model, where smart contracts are drafted (in whole or in part) in a programming language. For example, requirements that provide a protective function (such as warning the contracting parties about the implications of entering into the contract) may limit the ability of the parties to express certain contractual terms in a programming language.

Similarly, if the requirements involve actions of third parties or registration in a public register, contracts written in programming language may or may not be permissible. For example, whilst a notary may (be allowed to) accept


Is the existing law sufficiently broad/technology-neutral to allow records to be made and stored using DLT?
Yes
No
Consider if any further rules are necessary to specify the required level of security and compatibility with data protection requirements.
Consider legislative and/or IT infrastructure amendments to allow records to be made and stored using DLT.
Are there circumstances where the existing law mandates the involvement of third parties (e.g., notaries, public authorities, courts)?
Yes
No
Is the role of such third parties merely to validate the parties' identity or to warn parties about the implications of the transaction?
No further action.
To Validate
To Warn
Consider automating the process.
The involvement of such third parties should probably be maintained.

and record documents in a programming language, courts or authorities which operate public registers may not. Again, the question as to whether third parties are willing to accept and/or record documents in a programming language may depend on the purpose that these formalities or registration requirements serve.

It may be more difficult to meet relevant formalities where the smart contract is concluded by software (i.e., under the Concluding Model). For example, it may be difficult to meet requirements of written form or signature (depending on how such requirements are interpreted under the relevant legal system) if the software does not print a document which is then signed by the parties.

However, the legal framework may allow requirements of written form or signature to be satisfied by use of (qualified) electronic signatures or by use of cryptographic keys. Similarly, simple requirements for contracts to be made in writing or in text form may be met if electronic documents can be and are adequately stored, such as in a distributed ledger. For example, many legal frameworks require documents to be delivered or stored in what is known as a "durable medium"[29]. It is quite possible that a distributed ledger could qualify as "durable medium" and the indelible, tamper-proof nature of the records may be helpful in this regard.

2.3 Dealing with deficiencies and mistakes

As for any contract, some things may go wrong when negotiating, concluding, performing or enforcing a smart contract. Deficiencies or mistakes may arise in relation to various aspects of a smart contract, such as the offer or acceptance constituting the contract, the performance or enforcement of the contract, the software used to conclude, perform or enforce the contract or the parties' understanding of any of these aspects.

"Deficiencies or mistakes could apply equally to smart contracts and non-smart contracts: smart contracts also present certain additional difficulties."

Indeed, many types of deficiencies or mistakes applicable to smart contracts will be similar to those that may arise in respect of non-smart contracts, although the use of software and automation in smart contracts may give rise to some additional particularities and intricacies. The following examples of deficiencies or mistakes could apply equally to smart contracts and non-smart contracts:

- The offer and/or acceptance could be erroneous or one or both parties could have misunderstood what they have declared. Misunderstandings or mistakes could relate to the content or the legal implications of the offer or acceptance, the characteristics of goods to be delivered under the contract, etc.
- Misunderstandings of the parties may also relate to the purpose or content of the software that they intend to use to conclude, perform or enforce the contract.
- Parties may be mistaken as to whether they have capacity or authority to take a relevant action; for example, to issue or receive an offer or acceptance or to approve transactions.
- The contract or its wording could contain "gaps". This could be because a situation arises that the parties have not anticipated and the contract is silent as to what should happen. Alternatively, this may be because certain clauses are found to violate mandatory legal requirements such as consumer protection laws and are declared void (as discussed at sections 2.1.5 and 2.1.6 above). In the Integrated Model, these gaps could also relate to the software language parts of the contract.

In addition to these issues that are common to all types of contracts, smart contracts also present certain additional difficulties. For example, the following issues could arise:

- The smart contract software could contain errors or bugs in the contract code and, as a result, improperly execute the programme and associated actions or incorrectly assess whether the relevant conditions triggering an action have been met.
- The software may also improperly execute an action or incorrectly assess the conditions for other reasons (i.e., despite correct programming). For example, a smart contract may automatically request a loan where a company's bank account balance drops below a pre-programmed threshold. This may lead the software to request a loan in "error" if the software does not "know" that an *ad hoc* payment is due to be received, which would take the company's bank account balance above the relevant threshold, or if the bank has made an incorrect debit on the company's account. Similarly, the software used for a smart commercial real estate contract could be programmed to automatically lock the leased premises if the rent is not paid when due. However, the software may lock the premises in error, if it is unable to detect instances such as where the tenant discharges its obligation to pay rent by setting off own claims against the landlord or where the tenant has the right to withhold or reduce the payment in certain situations. Applicable laws may also prohibit the premises from being locked.
- The smart contract software may also not work properly on the relevant

29 The term "durable medium" is often defined as any instrument which enables the recipient to store information addressed personally to him in a way accessible for future reference for a period of time adequate for the purposes of the information and which allows the unchanged reproduction of the information stored (for example, see Article 4 of MiFID2 (Directive 2014/65/EU)).

distributed ledger (or within any other environment in which it runs) including for reasons relating to data sources, such as public registers, which the software is required to access (for example, those data sources may be incorrect, out-of-date, inaccessible or may no longer exist).

- The smart contract software may fail to work because it cannot take an action it is required to take (for example, a transfer of a digital asset from one party to another may not be possible to perform if the transferring party does not hold the relevant digital assets at the relevant time for performance).
- The smart contract software may also not work properly as a result of technical deficiencies or manipulations by the contracting or third parties (for example, as a result of a cyber hack).

2.3.1 Allocation of liability

These issues raise important questions of liability allocation: should liability fall to one of the parties, or should it be attributed according to culpability or based on the nature of the relationship (business-to-business or business-to-consumer)? Can liability be attributed at all in the case of parties who incur losses as the result of using open source software[30] or should such losses lie where they fall? Should liability be limited to direct losses or should liability extend to indirect losses (such as losses arising because, for example, a securities transaction failed to settle and the purchaser, needing those shares to launch a takeover bid for the relevant issuer, was unable to do so)? To what extent should the developer of the code that contains the errors or bugs be liable?

In our view, there is unlikely to be a single answer to these (and similar) questions which is appropriate for all smart contracts in all circumstances. The answers are likely to be driven by the specific context, including the nature of the parties, any governance codes or contractual relationships that may exist between the developer(s) of a distributed ledger operating system and the parties using it, the policy aims of the jurisdiction, the use to which the smart contract is put, etc. In the case of DLT-based smart contracts, this also may depend on whether the distributed ledger might be characterised in a particular way (e.g., as a joint venture, multi-party contract etc.) under the relevant legal system.

The way in which these issues are addressed and the questions of liability allocation would, in the absence of any specific provisions, fall to be determined under the existing legal system (contract laws, tort laws, etc.) and/or the contract itself (which may, where permissible, include clauses on liability, implications of deficiencies, etc.).[31] Indeed, the existing legal system and the contract may be perfectly sufficient to deal with such questions and allocations of liability without any need for further legislative reform specifically aimed at supporting smart contracts. We would not anticipate a legal system having a “gap” which meant that disputes regarding mistakes and deficiencies simply could not be adjudicated; this would be akin to a judge holding up his or her hands and saying “the law has nothing to say on this matter”, which has obviously not been the case to date when judges have been presented with novel facts and scenarios. Assuming the same is true of smart contracts, the question then becomes one of whether the outcomes provided for by the existing legal framework are desirable.

The outcomes which may result are manifold. Some issues will result in the invalidity, or modification, of the smart contract whereas other deficiencies or mistakes will be irrelevant or capable of remedy. In some cases, deficiencies and mistakes will trigger rights of one or both parties; for example, a right to challenge the offer or acceptance and/or to cancel, revoke, terminate or withdraw from the contract or to request damages.[32] In some cases, deficiencies or mistakes in smart contracts may need to be dealt with by stopping or suspending automatic performance of the contract (to the extent that such performance has not yet taken place) or by reversing performance that has already occurred (for example, by re-transferring payments or goods). The sections that follow consider different options, depending on the type of mistake or deficiency in question.

30 Open source software refers to software with source code that is freely available for anyone to view, use and edit. Open source software may often be developed in a collaborative, public manner. This means that if open source software is found to have deficiencies (e.g., errors or bugs), then there may be no readily identifiable developer or group of developers of that code who could potentially be held liable for the deficiencies.

31 In the financial services space, transactions under smart contracts are likely to be underpinned by a reasonably sophisticated matrix of contractual relationships, licences and liability allocation arrangements.

32 Some of these rights may also exist if there are no deficiencies or errors; for example, under many legal systems, consumers are granted rights to revoke certain contracts within a short period of time following conclusion.

Considerations for lawmakers

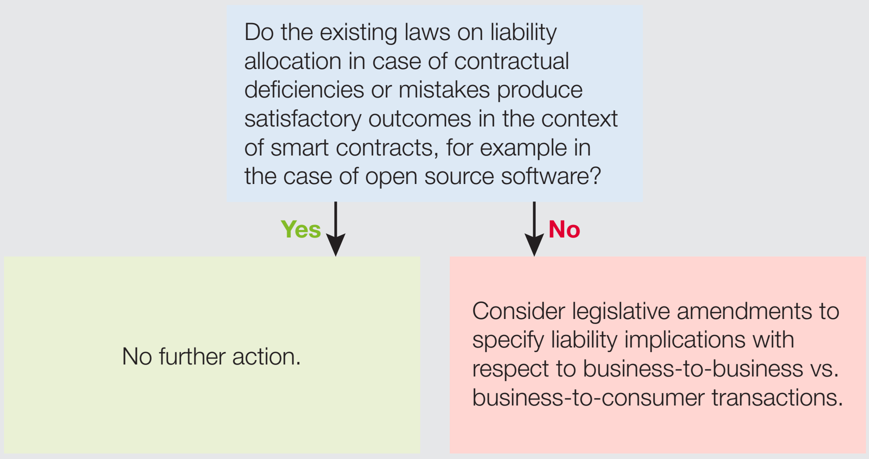
It is likely that existing legal systems and courts will have laws, rules, principles and tools to adequately deal with liability allocation in the event of contractual deficiencies or mistakes. However, it may be more difficult for lawmakers to assess whether the application of those laws, rules, principles and tools to smart contracts results in a desired outcome in respect of liability allocation.

As a first step in identifying where liability may be allocated in an undesirable way under a smart contract, lawmakers should consider adopting an approach common in software engineering and "test" the legislation in the context of different mistakes and deficiencies to determine likely outcomes for allocation of liability under a smart contract under the existing framework. To the extent that flaws or undesirable outcomes are identified, the law could be amended accordingly.

By way of example, if the existing legal framework does not specify how losses relating to a defect in open source software should be allocated, lawmakers may decide that it is appropriate to let losses lie where they fall for business-to-business contracts but not for business-to-consumer contracts, since consumers may generally be less able to bear such losses. In this case, lawmakers may decide to introduce specific laws setting out how liability should be allocated in this situation. Alternatively, lawmakers should be prepared to react to undesirable outcomes as cases arise in courts as they would ordinarily do so.

In any case, lawmakers may decide that the particularities of smart contracts mean that contractual parties should specifically address liability allocation (generally or in certain cases only). Alternatively, they may decide to empower certain authorities such as financial services regulators to make rules on liability allocation for smart contracts, to the extent possible and appropriate.

Lawmakers should consider whether they need to go further and mandate how liability must be allocated in certain circumstances, notwithstanding any agreement to the contrary. For example, in the case of smart contracts with consumers, lawmakers could provide that the non-consumer is deemed liable in the event of any bugs in the code, incorrect performance of the software, etc., and that this allocation of liability would override any conflicting term in the contract itself.


Do the existing laws on liability allocation in case of contractual deficiencies or mistakes produce satisfactory outcomes in the context of smart contracts, for example in the case of open source software?
Yes
No
No further action.
Consider legislative amendments to specify liability implications with respect to business-to-business vs. business-to-consumer transactions.

2.3.2 Ability of smart contracts to internalise resolution

Assuming that the legal framework allows for the parties to agree how deficiencies and mistakes will be dealt with, the question of whether and the extent to which smart contracts can automatically handle deficiencies and mistakes becomes an operational one, which will depend on the type of contract, deficiency and mistake, the specific implications and the features of the software. The software may, in particular, be able to handle deficiencies automatically where they were specifically anticipated by the contracting parties and/or where the software makes use of AI. In the former case, the smart contract could have a "default" outcome which the parties agree upfront in the event of certain deficiencies. For instance, if the smart contract is unable to transfer a digital asset because the relevant transferring party does not hold such digital asset, the contract may simply terminate and cease to run. Alternatively, it could be programmed to take steps to "buy in" the digital asset from a third party (similar to the buy-in mechanism used to remedy settlement fails in some securities transactions) or to compensate the purchaser if the price of the digital asset has increased (to take into account that it would be more expensive for the purchaser to "buy in" the asset from a third party itself).

However, we do not expect that it will be possible to anticipate and pre-programme a smart contract to take into account all potential scenarios. Therefore, we expect that, in general, some level of human input will be required when dealing with deficiencies. Consequently, automatability is likely to be affected where the parties have the right to choose between different ways of dealing with a deficiency or where the deficiency or mistake occurs outside the sphere of assessment for the software. For example, if the wrong type of physical commodity is delivered in (incorrect) performance of a smart contract, real world steps would need to be taken in order to correct the mistake. As smart contracts become more sophisticated, it may be possible to programme them to fix mistakes with minimal human intervention, particularly if associated digital infrastructure is developed to allow smart contracts to connect to and interact with electronic registers of assets, bank accounts etc.

Considerations for lawmakers

Lawmakers should, as a first step, determine whether existing laws establish a doctrine of mistake and identify the impact that this may have on smart contracts. Lawmakers should then consider as a policy matter whether deficiencies in the parties' understanding of the software should affect the validity of a smart contract.

In considering this, lawmakers should take account of whether there are circumstances in which this may be more justified. For example, where a counterparty is a consumer, lawmakers may consider it appropriate to allow the consumer to set aside a smart contract where the consumer had a mistaken understanding of the contract or may require the non-consumer to prove that the consumer was not mistaken.

Lawmakers should also consider whether different standards or requirements should apply in different circumstances. For example, a lower level of understanding of the software may be acceptable (i.e., meaning that a deficiency or mistake in a party's understanding would not affect the validity of the contract) under the Non-Integrated Model as compared with the Integrated Model.

Lawmakers should also consider whether it would be appropriate to impose a duty of good faith upon smart contract coding service providers to exercise reasonable or best efforts to make smart contract code consistent with the written intentions of the contracting parties (which may, for example, be set out in a natural language contract under the Non-Integrated Model).

2.3.3 Mistakes in understanding

Existing rules and doctrines should apply to any deficiencies or mistakes in one or both parties' understanding of the contract, offer, acceptance or related software. Some legal systems may, for example, consider some level of deficiency in the parties' understanding of the software as being irrelevant to the determination of the validity of the contract. Other legal systems may assess this differently for all smart contract models or for the Integrated Model (where the contract is drafted in whole or in part in software language). Some legal systems may also determine that deficiencies or mistakes in the understanding of the software language in the Integrated Model should be dealt with in much the same way as those relating to the understanding of non-native natural language contracts. A legal system may, for example, expect the parties to take reasonable steps to ensure they understand the content of the contract.

Depending on the legal system in question, the parties' understanding of a contract (or the legal consequences of that understanding) may then be based on an objective standard of what a reasonably diligent person in that party's position would be expected to understand or to have done in order to gain such understanding. In the context of smart contracts, a relevant question would therefore be whether a diligent party would be expected to engage an IT specialist to review draft smart contracts.

2.3.4 Errors or gaps in the contract

Similarly, errors and gaps in the contract (resulting, for example, from an incomplete agreement or violation of statutory provisions) would also likely be subject to existing rules and doctrines. These rules may stipulate that the gaps should be "filled" in line with the parties' true or apparent intentions or in line with statutory provisions.

The question as to whether it is possible to handle these deficiencies in an automated way will depend on the individual circumstances, legal framework and software. In some cases, human input may be needed. For instance, if a time limit for a certain action agreed in a smart contract is void under mandatory provisions because it is too short and must be replaced by an "appropriate" time limit, automated handling would require the software to be able to determine what that "appropriate" time limit should be with regard to the individual circumstances – this would be very difficult to achieve as it inherently involves the exercise of judgement. For instance, a court might decide that a term in an insurance contract that requires a consumer to notify the insurer of a claim within four hours of an insured event arising is too onerous, and that the customer should be allowed a longer notification period. The court's assessment of how much longer this should be may depend on the nature and circumstances of the claim.

In practice, it may be particularly difficult to deal with gaps arising from void clauses in smart contracts under the Integrated Model, as deletion of the void clause may fundamentally alter the software code and could have implications for the executability of the remaining code. In the Non-Integrated Model, similar questions may arise if voiding the clause requires changes to the software performing the smart contract.

These difficulties call into question whether legal techniques of striking out an offending clause and replacing it with the mandatory position may be inappropriate for smart contracts. The task would not be a simple linguistic one, but would likely require the software engineers to re-code, re-test, validate and re-run the programme. Instead, lawmakers may wish to consider whether alternative approaches or remedies should be developed for smart contracts. For example, instead of striking out an unfair penalty clause, an equivalent economic result could be achieved by awarding an appropriate amount of compensation.

Considerations for lawmakers

As a first step, lawmakers should consider whether existing laws contain provisions that seek to fill gaps or remedy errors in the contract. If this is the case, lawmakers may need to consider whether the current approach is viable for smart contracts, particularly in the case of gaps arising where a smart contract clause violates statutory protection or other requirements and so is found to be void.

Lawmakers may therefore wish to consider whether alternative approaches or remedies should be developed for smart contracts. For example, lawmakers might adjust the law to allow offending terms to continue to have effect but to provide for compensation to be awarded, such that the affected party would be put in the same position as if the relevant term had been declared void and struck out of the contract (and the resulting gap filled according to relevant legal principles and provisions).

2.3.5 Errors in the software

The implications of errors in the software will often depend on which party is responsible (and liable) for programming or providing the software. However, solutions will also be required for cases where both parties are responsible or where they agree on software produced or provided by a third party such as a DLT platform operator (which may seek to limit its own liability for errors in the software) or open source software.

When determining who is liable for such errors, courts would generally seek to apply the usual principles of contract and/or tort law (or similar laws and principles) under the relevant legal system. The outcome of this determination will usually be highly fact-specific. For example, if the parties use software provided by a DLT platform operator under a licence agreement, the courts would, in particular, need to consider whether the platform operator is liable for errors in the software under the terms of the licence agreement, or whether the platform operator has successfully excluded liability for such errors.

If parties use open source software and have no express contractual relationship with the software developer(s), the courts may need to consider further questions, such as whether developers owe a duty of care towards users of the software and whether the losses incurred were reasonably foreseeable or are otherwise losses for which damages may be awarded.[33] The precise tests will vary from jurisdiction to jurisdiction. In practice, it may not be possible to identify the developer(s) that are responsible for a particular error. Even if developers can be identified, they may be private individuals located anywhere in the world and/or may not have sufficient resources to meet a damages claim. Therefore, seeking to hold individual developers of open source software liable for losses under smart contracts may not be a viable, practical solution.

The courts may therefore turn to consider whether and the extent to which one party or another may be held liable for losses arising from the software error. Again, this is likely to depend on features of the relevant fact pattern. The courts may, for example, consider the relative sophistication and bargaining power of the parties, for example whether one party is a consumer or whether one party has effectively imposed use of the software on the other.

In the context of a DLT network, the courts may also consider whether the network participants (or a subset of those participants with certain roles or responsibilities) should be held jointly liable for software errors in the network's underlying software or code, for example if the DLT network is characterised as a form of partnership under the relevant legal system and/or if its participants may be jointly liable under the relevant laws or legal principles.[34]

The implications of a software error may also differ depending on the function of the software (for example, whether it concludes contracts in the Concluding Model and/or performs contracts in the Performance Model). The Non-Integrated Model and the Integrated Model may also be treated differently. Under the Non-Integrated Model, the software does not form part of the contract. Therefore, if an error is restricted to the software in a Non-Integrated Model the terms of the (correct) natural language contract are more likely to prevail.

As a related question and if not sufficiently determined by applicable laws, the parties should also determine how to deal with other instances where the software does not operate as they intended (such as in the case of cyberattacks, unavailable data sources or internet, corruption of data, etc.). For example, these types of events may qualify as *force majeure* events under applicable laws or as expressly agreed by the parties, such that a party suffering from the *force majeure* event will not be in breach of contract or

Considerations for lawmakers

Lawmakers should assess how liability would be allocated under the current legal framework where there is an error in the smart contract software, particularly under the Integrated Model. In particular, lawmakers should consider how existing legal principles might be applied to a situation where the error arises in open source software or in software developed and provided by a third party. If the outcome is unclear or might be undesirable from a policy perspective, lawmakers may wish to consider amending existing laws or introducing new laws to address these issues.

Lawmakers or regulators may wish to consider developing smart contract templates, which could include a field for the parties to indicate how they have agreed that liability will be allocated in the event of a coding error. Lawmakers or regulators may also consider whether it would be appropriate to provide a warning message in the event that the parties do not expressly agree on allocation of liability for a coding error (and so leave this field blank).

Lawmakers should also consider whether existing laws provide for force majeure events which might excuse one or more parties to a contract from performing their obligations. If there is such a framework, lawmakers should assess whether the description of those force majeure events would include cyberattack, unavailability of data sources or internet access, corruption of data, etc.

Either way, lawmakers should assess whether this is appropriate in the context of smart contracts – given that they may be more fundamentally affected by these sorts of events than other contracts – and whether express changes to the law should be made to more readily allow these types of events to qualify as force majeure events in the context of smart contracts.

33 Note that open source software is often provided under a General Public Licence or Open Source Software Licence, which typically includes broad limitation of liability language. However, this language may not always be effective to exclude or limit liability, for example if there are overriding mandatory principles or requirements of local law, such as requirements relating to general terms and conditions as discussed at section 2.1.5 above. In this context, the courts would also need to consider whether a contract exists at all and if so, whether the limitation of liability was validly included in the contract.

34 For further discussion on how liability for software errors or deficiencies might be addressed under different legal systems, see section III.D of "*The Distributed Liability of Distributed Ledgers: Legal Risks of Blockchain*", Dirk A. Zetzsche, Ross P. Buckley and Douglas W. Arner ([2017] UNSWLRS 52), available at http://www5.austlii.edu.au/au/journals/UNSWLRS/2017/52.pdf

otherwise liable for a failure to perform arising from that event.

2.3.6 The added complication of DLT – void transactions

Using DLT raises particular questions and issues. For example, distributed ledgers are inherently resistant to modification of recorded data and any change in the database requires the creation of a new "transaction". This raises the question of how to deal with applicable laws stipulating that, as a result of certain deficiencies, a smart contract is void from the outset.[35] If the distributed ledger does not allow the deletion, "overwriting" or amendment of recorded transactions, a possible solution could be to allow for "reverse transactions" which seek to restore the parties to the position they would have been in had the deficiency not occurred.

However, any possible solution will have to take into account whether applicable laws require the original situation to be fully restored from an economic perspective only or also from a legal perspective. For example, there is a legal difference between the ownership records showing that there has been a transfer of an asset from Party A to Party B and a "reverse transaction" thereafter, and the position where, following the transfer of an asset from Party A to Party B, it is determined that this transfer was void from the outset and so the records should show that it never took place[36]. The latter is potentially more difficult to achieve in the case of a distributed ledger (although not necessarily impossible – as discussed at section 2.4.1 below, the ledger could be altered or overridden by consensus or certain other means).

Considerations for lawmakers

As a first step, lawmakers should identify whether existing laws require that, in some circumstances (e.g., where the transaction is illegal or impossible, or one of the parties does not have the necessary legal capacity), parties be put in the position they would have been in, had the contract or transaction never occurred. If that is the case, in view of the immutability of DLT, lawmakers may need to consider alternative remedies which achieve the same or similar outcomes for the parties (e.g., compensation or reversal of the transaction). However, this would still need to be balanced against the consideration of whether, in some circumstances, there may not be an adequate substitute for a contract being treated as void from the outset – perhaps because a party may wish the contract to be expunged from public records on the basis that it was never entered into, because for example, such transaction may indicate bad judgment, be embarrassing or affect credit ratings, etc.

Lawmakers should also determine what impact such alterations may have on other third parties, such as those who purchase an asset following its prior purported acquisition under a void smart contract. Again, consideration would need to be given to whether alternative remedies could be sufficient in such circumstances.

This is primarily a practical issue rather than a conceptual legal issue. In general, existing legal requirements and principles will determine whether or not a contract is void and what the consequences of this should be. However, if applicable laws require the parties to be put back into their original legal positions, use of DLT may make it difficult to achieve this outcome in practice due to the immutability of the ledger. If so, lawmakers may consider that it is desirable to expressly provide for alternative solutions in these situations. These may already exist and be used in other situations. For example, existing laws may provide for a subsequent third-party purchaser to retain title to an asset that was the subject of a prior void transaction in certain circumstances[37] and instead provide for an alternative remedy, such as compensation, for the original purported seller.

2.4 Amendments to smart contracts

The parties may also agree, or one party may have the right, to amend a smart contract. For example, the parties may agree that the borrower has the right to suspend its payments under a loan agreement for a certain period of time and that this does not trigger the consequences for non-performance or late performance provided for in the contract or under applicable laws. This right may be agreed or arise at the outset. The parties may also seek to amend the smart contract at a later date (i.e., once the smart contract has been formed and is being performed), to introduce such right. Whether and to what extent smart contracts and, in

35 This may affect a "chain of contracts" – e.g., a chain of purchase transactions where the first contract is void and one or other of the contracting parties may, hence, under applicable laws be considered to have disposed of the relevant assets without being entitled to do so – notwithstanding that, applicable laws in most legal systems will contain rules and doctrines to handle such situation from a legal perspective.

36 By way of example, this might be the case where the transaction is illegal or impossible, or because one of the parties lacked the necessary legal capacity.

37 For example, if there is a bona fide purchaser of the asset which does not have notice of the prior void transaction.

A fork in the road...

On 17 June 2016, a smart contract code running on the Ethereum DLT was “hacked” or, more precisely, certain vulnerabilities in the way the smart contract had been coded were exploited. As a result, there were calls by users of the Ethereum DLT to change the records of the Ethereum DLT to effectively reverse the transactions which had made use of this hack. However, there were both supporters and objectors to this proposed approach – the latter taking the view that changing the Ethereum DLT was not a “remedy” to a software defect; the software had performed correctly and exactly as coded, and it simply gave rise to results which some had not anticipated and which were not desirable. Arguably, this transaction reversal would undermine the immutability and tamper-proof nature of DLT records – which are seen as key benefits of DLT. The divergence in views between the user community ultimately led one group of users to make changes to their version of the DLT software and roll back “rogue” transactions. Other users did not, which resulted in a “fork”, where two versions of the ledger lead off in different directions. These were called the Ethereum fork (rolling back the rogue transactions) and the Ethereum “classic” fork (accepting the rogue transactions as being valid).

particular, their encoded aspects may provide sufficient flexibility to accommodate such changes (or even handle them automatically) depends on the specific contract, the change that is needed and the features of the software. The situation may be easier where the parties anticipate that a specific change may occur and can provide for this in the smart contract itself from the outset.

One of the features of smart contracts which operate on a distributed ledger is the immutable nature of the code once it has been written – this may act as an effective limit to amendments. However, there may be practical answers to this issue – namely overrides, termination and other techniques, which we examine below.

"One of the features of smart contracts which operate on a distributed ledger is the immutable nature of the code once it has been written – this may act as an effective limit to amendments."

2.4.1 Overriding the ledger

Whilst it is often stated that a distributed ledger is immutable, that may not always be the case – the relevant protocols could allow for changes to be made, whether by majority consensus or possibly through some other override. There have already been examples of ledgers being changed in response to circumstances that were not envisaged at the outset – see "a fork in the road…" at page 37. This ability to effect changes to records could be specifically provided for within a DLT environment, subject to certain safeguards.

Whilst forks have their opponents[38], this technique of rolling back to previous versions of the database or collectively agreeing to remove elements of data previously recorded nonetheless remains a possibility. It might prove to be useful, for example, if it becomes necessary to "delete" data for legal or regulatory reasons. For instance, this may be the case for personal data included in the ledger – see "A note on GDPR" at section 2.7. Clearly, however, many considerations would need to be taken into account and appropriate safeguards put in place if this technique is to be adopted as a desired feature of a DLT system.

The smart contract itself could also provide for the possibility of amendment or update by the parties – this might be a unilateral ability for one of the parties to override the smart contract, by signing the changes using their private key, or it could require both parties to sign using their private keys for the override to be effective. However, such changes may need to be countenanced at the outset of the smart contract and an appropriate software mechanism for making changes would need to be built in.

2.4.2 Termination and replacement

Another alternative (although probably more cumbersome), may be for the parties to terminate the smart contract and put in place a new smart contract with the new terms. Clearly, any accrued rights and liabilities under the original smart contract would need to be adequately dealt with and there are likely to be myriad legal consequences such as triggering certain compliance requirements – for example, if the smart contract is a derivative transaction, those lifecycle events of termination and the entry into the new contract may require reporting to regulatory authorities or other market utilities.

This process of termination and replacement of a smart contract with a new contract may also affect how the arrangements are characterised or treated. For example, if the smart contract was a foreign exchange contract which is closed out and re-opened on a daily basis, the regulatory framework may treat such transactions either as a "rolling spot" derivatives transaction or possibly as a series of unregulated spot foreign exchange transactions.

2.4.3 Other techniques

There may be other techniques that could be used by the parties (or perhaps imposed by a court if the changes required are to reflect a court judgment or mandatory provisions of law) to achieve an analogous outcome. For instance, the parties could agree to give effect to the amendment outside the smart contract. In the context of a smart loan contract, for example, if the lender agrees to reduce the interest rate payable but the smart contract continues to effect repayments on the basis of the previous, higher interest rate, the lender could make a repayment of the excess to the borrower. This could even be achieved by entering into a new smart contract for a

38 In particular, forks could lead to the emergence of two different operating systems. With two separate DLT systems, developers and companies building applications, as well as users, must decide which system to use and support.

counter-directional payment transaction, which is conditional on receiving the loan repayments under the original smart contract. It is important to note, however, that whilst the economic outcome generally should be the same as reducing the loan repayment, there are important differences from both a factual and a legal perspective. For example, the counter-directional payment entails the borrower taking credit risk on the lender (i.e., the risk that the lender might become insolvent, meaning that the borrower might not receive the full amount it is due) which would not happen where the interest rate is simply lowered. In addition, legal implications may arise, for example under regulatory requirements, tax laws, changes to the position of the borrower and lender in court proceedings or in the case of insolvency of one or both parties.

2.5 Governing law and jurisdiction

Questions relating to governing law and jurisdiction are not specific to smart contracts. Where there is a cross-border element to the contract, applicable conflict of law rules and dispute resolution, arbitration and choice of court requirements must be considered – these are complex and there is significant jurisprudence on them. We should, however, not discount that applicable requirements may limit the parties' freedom to agree on any governing law, jurisdiction, court or other dispute resolution mechanism – for example, if consumers are involved, the governing law, jurisdiction and court may need to reflect the residence of the consumers. In addition, as in the case of non-smart contracts, laws of several jurisdictions may apply to various actions taken in connection with a smart contract. For example, the smart contract itself could be governed by French law. However, if the services to be provided under the contract will be delivered to or from Spain or if the contract relates to transfer of Spanish securities or property, Spanish laws relating to cross-border trade, tax and/or ownership of securities and property may apply to the transactions under the contract.

If a distributed ledger used for a smart contract has nodes located in multiple jurisdictions, this could give rise to some particularly complicated questions of governing law and jurisdiction, depending on the subject matter of the contract and on the connections to particular jurisdictions and the nexus deemed relevant under applicable conflict of laws rules[39]. For example, where these rules provide that the applicable law is the law of the jurisdiction where an asset is located, this may be hard to define, particularly for native and other intangible assets whose location may be determined by the location of the record of ownership. If this record is a distributed ledger, this could effectively be any jurisdiction in the world.

These issues are not necessarily new. The location of where an activity is considered to take place, where ownership records are held or where data is received, sent, processed and stored may currently be different from the location of the financial institution, corporate or individual in question, particularly in the context of the growing use of cloud services. See "A note on GDPR" at section 2.7.

In many jurisdictions, the courts will generally seek to uphold the parties' express choice of law governing a contract (including a smart contract) to the extent possible.[40] In the absence of an express choice of governing law, courts would generally apply existing legal rules or principles to determine the law governing the smart contract. For contractual obligations, this will often depend on indicators, such as the habitual residence of the seller or service provider,[41] which are generally capable of being determined for smart contracts, just as for non-smart contracts. The courts in some jurisdictions may also apply mandatory overriding provisions of law to contracts involving a person located or habitually resident in that jurisdiction, regardless of the governing law of the contract. Such mandatory provisions may include consumer protection laws (e.g., distance selling requirements), data protection laws or laws relating to general terms and conditions.

However, some conflict of laws rules, such as those determining the law applicable to questions of asset ownership and other proprietary rights, may be difficult to apply or may not apply at all in the context of transactions and assets recorded using DLT. For example,

39 This also assumes that the location of the nodes is easily identifiable, for example based on their IP address, and that users do not attempt to conceal their physical location through a VPN or similar means.

40 For EU jurisdictions, for example, this is provided for under Article 3 of Regulation (EC) No. 593/2008 of the European Parliament and of the Council of 17 June 2008 on the law applicable to contractual obligations (Rome I Regulation). Similarly, the Rome II Regulation (Regulation 864/2007/EC) allows parties to choose the law governing non-contractual obligations within its scope of application (under Article 14) and provides uniform rules for determining the law applicable to such non-contractual obligations in the absence of such a choice.

41 For example, under Article 4 Rome I Regulation.

The Hague Convention of 5 July 2006 on the Law Applicable to Certain Rights in Respect of Securities Held with an Intermediary[42] would not seem to apply to securities recorded on a distributed ledger, as the securities would not be held with an intermediary (this being one of the key features of DLT). Further, if relevant conflict of laws rules specify the location of the relevant account as determining the applicable law, they may give no clear answer as to which law should apply in the context of transactions and assets recorded using DLT, where there is no centralised account ledger or register of ownership. Therefore, there may be gaps or deficiencies in existing conflict of laws frameworks that lawmakers may wish to fill or resolve.

Of course, the ways in which other jurisdictions address conflict of laws issues may impact cross-border disputes. Where jurisdictions take inconsistent approaches to conflict of laws questions, this can result in further uncertainties. Therefore, co-ordinated international efforts may ultimately be necessary or beneficial in order to address these conflict of laws issues in a consistent manner across jurisdictions (as is the aim of the Hague Securities Convention in the context of intermediated securities). However, this is likely a longer term ambition, rather than a short term goal.

Parties might also seek greater certainty in other ways, which may involve some kind of compromise in terms of the decentralised and disintermediated nature of a DLT-based holding structure. For example, parties might hold assets via a private distributed ledger with a centralised operator (rather than using a public distributed ledger with no centralised operator). In this case, applicable provisions may stipulate that the location of the operator determines the law applicable to proprietary claims relating to such assets. Alternatively, a party might enter into an agreement (which could be outside of the DLT) whereby it would appoint an intermediary – a custodian of sorts – to hold DLT-based assets on its behalf such that the intermediary would be recorded as the owner of the assets in the relevant distributed ledger. The party would then have a claim against that intermediary in respect of those assets (rather than a direct ownership right to the assets themselves). In this case, the party may be able to better determine the governing law of the claim it has vis-à-vis the intermediary and achieve greater certainty in terms of the applicable law. However, as noted above, this does involve a compromise as the disintermediated nature of the DLT-based holding structure is lost.

42 The Hague Securities Convention, available at https://www.hcch.net/en/instruments/conventions/full-text/?cid=72

Code is law

Some fervent proponents within the IT community take a purist view that only the code inherent in a distributed ledger should apply to and govern contracts within such distributed ledger, and that no other (state) law should do so ("code is law"). In this paradigm, the jurisdiction of state powers and authorities would be excluded and only the providers of the code could determine the applicable rules. However, this is unlikely to be generally accepted in practice and the position conflicts with (current) limits of technology to sufficiently regulate and decide all possible matters.

More fundamentally, states and their courts are unlikely to accept such a position (at least unless a unique independent authority is set up) since it undermines their sovereignty and basic principles of democracy and the rule of law. In addition, stakeholders may prefer that a state or an independent authority sets rules and applies them to specific cases.

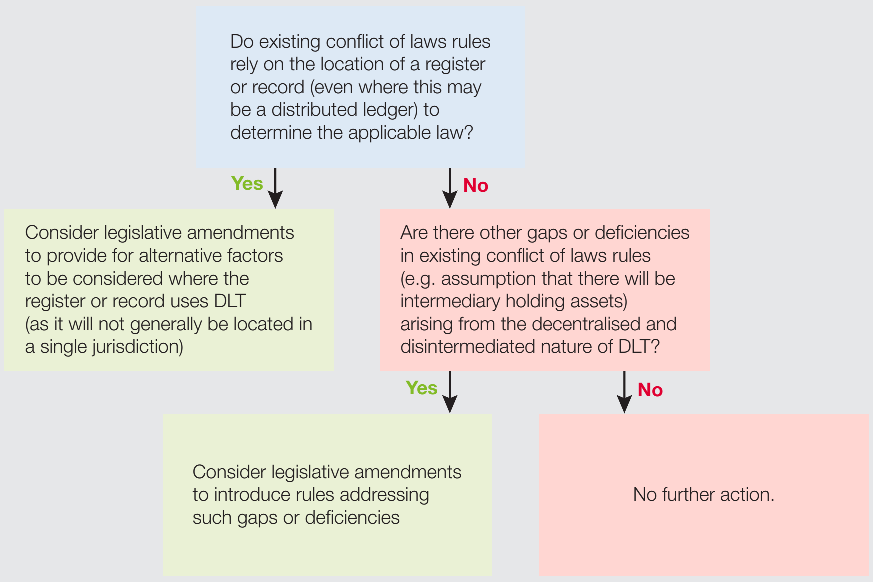
Considerations for lawmakers

At a high level, similar types of conflict of laws questions will arise, and legal principles apply, for smart contracts as for non-smart contracts. For example, parties may have the ability to choose the law governing contractual and non-contractual obligations under the existing legal framework. In the absence of an express choice of law, the existing framework may then set out the factors that courts should consider when determining the applicable law.

However, conflict of laws analysis can be complex and additional issues and uncertainties may arise in the context of smart contracts that use DLT, particularly if applicable rules use the location of an intermediary, account or other record to determine the applicable law (given the disintermediated and decentralised nature of DLT).

As a first step, lawmakers should consider what conflict of laws rules would apply to DLT-based smart contracts and what practical factors would be used to determine the applicable law, including in circumstances where the parties have not made an express choice of governing law. Lawmakers should also consider whether there are mandatory laws in their jurisdiction (i.e., which counterparties cannot override by contractual agreement) where the domestic law would apply and domestic courts would have jurisdiction. Lawmakers should then consider the conflict of laws rules and other factors that would determine when such mandatory laws would apply.

Where these factors relate to the location of certain data (such as an asset register, personal data, etc.), lawmakers may need to test whether such factors would continue to be appropriate in the context of a DLT system where nodes storing such data may potentially be located in multiple jurisdictions. Lawmakers may need to amend these rules to provide for consideration of alternative factors that do not relate to the location of the data or, if data location factors are retained, they may need to be adjusted to take into account that the relevant data may be located in multiple jurisdictions at the same time. Lawmakers should also consider whether there are other gaps or deficiencies in existing conflict of laws rules arising from the decentralised and disintermediated nature of DLT, for example, if existing rules only cover situations where there are intermediary holding assets. If so, lawmakers may wish to amend existing laws to address such gaps or deficiencies.


Do existing conflict of laws rules rely on the location of a register or record (even where this may be a distributed ledger) to determine the applicable law?
Yes
No
Consider legislative amendments to provide for alternative factors to be considered where the register or record uses DLT (as it will not generally be located in a single jurisdiction)
Are there other gaps or deficiencies in existing conflict of laws rules (e.g. assumption that there will be intermediary holding assets) arising from the decentralised and disintermediated nature of DLT?
Yes
No
Consider legislative amendments to introduce rules addressing such gaps or deficiencies
No further action.

2.6 Dispute resolution

Generally, smart contracts may be reviewed by courts or arbitration tribunals under the relevant legal system, just as any non-smart contract. The relevant legal system also determines whether and to what extent the contracting parties may agree on or exclude particular dispute resolution mechanisms (such as arbitration). However, smart contracts are somewhat particular to the extent they are automated. As described further below, automation may impact how parties access the courts and comply with relevant procedural requirements. It may also affect which party bears the burden of proof in practice, and may limit the current exclusive powers of state courts to judge and legally enforce contractual claims and other rights and obligations.

2.6.1 Burden of proof for smart contracts

In many jurisdictions, applicable procedural rules will require each party to assert its own claims in court and to apply for its legal enforcement with a court. As a general rule, a claimant will often have to prove the facts on which it seeks to rely and will bear the risk that such facts cannot be proven. For example, if Party A fails to make a payment under a non-smart contract that obliges Party A to make the payment to Party B, Party B would have to assert and, generally, prove in court that it has a valid payment claim and that Party A has not paid. In addition, Party B would have to apply to the court for legal enforcement of a court judgment finding that Party B has a valid payment claim, if Party A still does not pay. Hence, Party B would bear the legal risk that it cannot prove its claim and the risk that Party A may enter into insolvency prior to legal enforcement of the judgment.

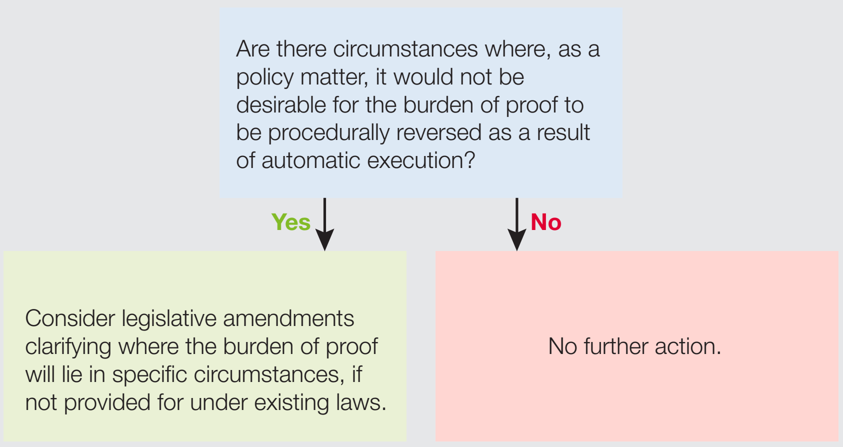
Whilst these general principles also apply to claims relating to smart contracts, the circumstances of the claim (and hence the burden of proof and associated risks) are likely to be reversed as a result of automation. If, in the above example, the payment were automated under a smart contract, Party B would automatically receive payment and so it would not have to assert its payment claim in court. Rather, Party A would now have to assert and, generally, prove in court its claim for repayment of the relevant amount if it considers that the payment should not have been made. As a result, Party A would bear the legal risk of having to prove its claim as well as the risk that Party B might enter into insolvency before the repayment is made. There may also be other legal consequences for Party B, such as an inability to exercise rights of retention or set-off either within or outside of litigation.

The party benefitting from this procedural "reverse" and shift in risk (Party B in the above example) may therefore be more willing to enter into a contract with a

Considerations for lawmakers

As a first step, lawmakers should determine the circumstances in which, as a policy matter, it would not be desirable for the burden of proof to be procedurally reversed as a result of automated execution. Once those circumstances are identified, lawmakers should consider introducing new laws, or clarifying existing laws to make clear that a smart contract arrangement in such context must not result in the effective reversal of the burden of proof in these circumstances or at least to mitigate these risks if the existing laws do not already sufficiently provide for this.

For example, and using the illustration above of a payment automatically being made by a consumer (Party A) to a non-consumer (Party B), the law could expressly provide that, where a consumer claims that it should not have made an automatic payment under a smart contract, the non-consumer counterparty would have to refund the consumer automatically and immediately and then seek to prove that the payment was in fact justified.


Are there circumstances where, as a policy matter, it would not be desirable for the burden of proof to be procedurally reversed as a result of automatic execution?
Yes
No
Consider legislative amendments clarifying where the burden of proof will lie in specific circumstances, if not provided for under existing laws.
No further action.

pseudonymous third party. However, this procedural "reverse" and shift in risk resulting from the automatic performance of a contract may not be desirable in certain cases. For example, certain types of parties (such as consumers) may be less able to assume such risk. Hence, legal systems may seek to prohibit any change to the usual burden of proof to the detriment of a consumer, or, at least may prevent this from being stipulated in "general terms and conditions".

2.6.2 Presenting arguments and evidence

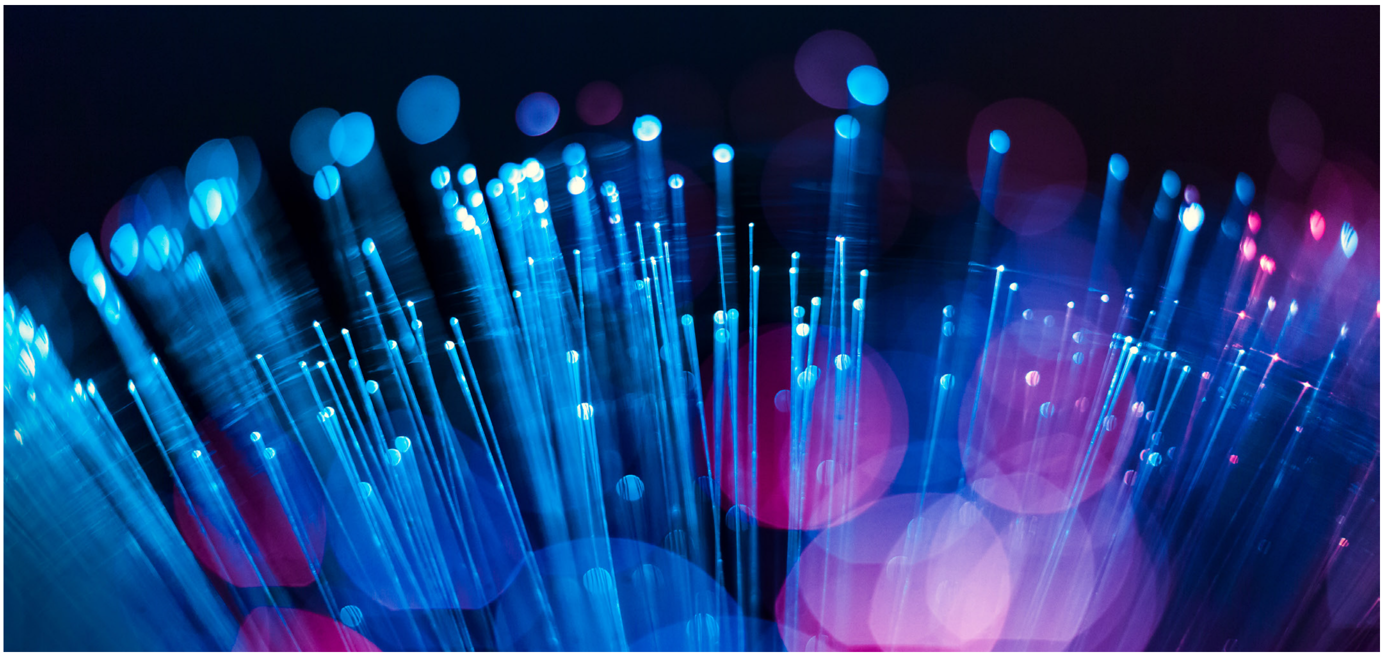
Particular considerations may apply when presenting arguments and evidence relating to smart contracts in court, to the extent the smart contracts are drafted in a programming language or concluded or performed by software (including in relation to any linked software data sources).

The applicable laws may already provide for the possibility to present certain digital evidence, and courts may require natural language translations of any parts of the contract drafted in a programming language or rely on a qualified expert opinion to understand and assess the software or computer code. For smart contracts, this may raise questions about the extent to which this analysis and assessment relates to legal issues (rather than factual or technical matters). This may be the case if, for example, the analysis and assessment of computer code amounts to an examination of what the parties have actually agreed. This is particularly likely to be the case in the Integrated Model where some or all of the terms of the agreement are expressed in programming language. A legal system may have distinct rules governing the assessment of facts and legal matters by independent experts or when independent experts may assess or provide advice on legal matters.

To the extent permitted under the applicable legal rules, smart contracts may also include terms limiting the types of evidence that are permissible and that may be considered in a dispute.

2.6.3 Arbitration and specialist technical courts

The relevant legal system may allow the contracting parties to agree to refer certain disputes relating to smart contracts to a neutral third party or parties for arbitration. By way of illustration, if a dispute arises, or if a party to a smart contract identifies a deficiency or other issue relating to the smart contract that it cannot agree or resolve directly with the other party, it could raise an objection and trigger an arbitration clause in the smart contract. Depending on the terms of the arbitration clause, this may automatically suspend performance of the smart contract whilst the arbitrator assesses and resolves the issue. The arbitrator (or arbitral tribunal) would then



assess and resolve the issue to the extent possible. The arbitrator's decision could then be fed directly into a DLT-based smart contract by means of an oracle.[43] A number of organisations have already started developing arbitration clauses or "libraries" that parties could include in their smart contracts.[44]

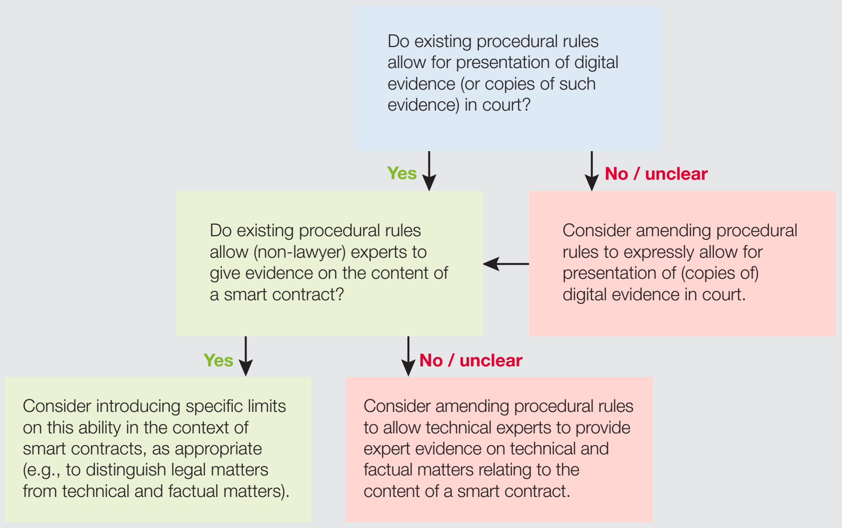
The arbitrator could also be (partially) automated and could, for example, have control over funds deposited by parties to a smart loan contract, which it could automatically release where it resolves a dispute, deficiency or mistake by deciding that one party owes an amount to the other under the contract.

In due course and once we start seeing greater use of smart contracts in practice, lawmakers may also wish to consider whether to establish courts that specialise in smart contract disputes, in the same way that, for example, specialist Admiralty Courts have been established in England to adjudicate maritime matters and the Technology and Construction Court to adjudicate on construction, engineering and technology disputes. Similarly, some continental European courts have established specialist chambers, for example, for construction, trade-related or medical disputes, which would be adjudicated by technical experts.

Considerations for lawmakers

As a first step, lawmakers should consider carefully the procedural rules regarding admissibility of evidence in court proceedings and whether there may be a need to provide for or amend existing rules on the presentation of digital evidence (or printed copies of such evidence) in court. For example, lawmakers may wish to consider whether procedural rules require (or should be amended to require) digital evidence to be presented in a specific manner or for printed copies of digital evidence to be certified or otherwise verified in some way to confirm that they are genuine and accurate copies of the digital evidence.

Lawmakers should consider the admissibility and status of natural language translations of smart contracts that are written in a programming language. For example, would (or should) existing procedural rules around translation of foreign language contracts apply to natural language translations of smart contracts? Lawmakers should also consider whether a natural language translation is sufficient or whether technical experts may need to be called upon to give expert evidence on the content of a smart contract written in a programming language. This could raise the question of whether and to what extent the expert may be providing evidence on matters of law rather than matters of fact. Lawmakers may therefore wish to consider whether to establish specific requirements and limits on the ability of (non-lawyer) technical experts to give this sort of evidence, in line with the jurisdiction's general approach to this issue.


Do existing procedural rules allow for presentation of digital evidence (or copies of such evidence) in court?
Yes
No / unclear
Do existing procedural rules allow (non-lawyer) experts to give evidence on the content of a smart contract?
Consider amending procedural rules to expressly allow for presentation of (copies of) digital evidence in court.
Yes
No / unclear
Consider introducing specific limits on this ability in the context of smart contracts, as appropriate (e.g., to distinguish legal matters from technical and factual matters).
Consider amending procedural rules to allow technical experts to provide expert evidence on technical and factual matters relating to the content of a smart contract.

43 As explained at section 1.2 above, an "oracle" is an interface connecting a distributed ledger to a trusted data source or other input.

44 For example, see CodeLegit Conducts First Blockchain-based Smart Contract Arbitration Proceeding (July 2017), available at: https://datarella.com/codelegit-conducts-first-blockchain-based-smart-contractarbitration-proceeding/

Considerations for lawmakers

Lawmakers should also determine whether, and the extent to which, existing laws would permit parties to refer disputes under smart contracts to arbitration. If not, existing laws may need to be updated, should lawmakers wish to facilitate the resolution of smart contract disputes through this mechanism.

Alternatively, if lawmakers do not consider arbitration to be appropriate (or even if they do), lawmakers should also determine whether decisions arising from existing dispute resolution mechanisms (such as court judgments) could be reflected directly in the smart contract. For example, this might involve court judgments becoming a supported data source or oracle input for DLT-based smart contracts. If not, lawmakers might consider making changes to existing rules regarding how judgments may be published and utilised by DLT-based smart contract systems.

2.7 Confidentiality

The contracting parties may wish to keep the existence of a smart contract and/or its terms and conditions private – and such privacy may also be required under applicable laws (e.g., data protection laws or the laws governing specific industry sectors such as the banking and insurance or health sector).

For smart contracts (just as for traditional contracts) this could be ensured by including terms relating to confidentiality in the smart contract, provided that the software does not run in the public sphere but, for example, runs on a private server or a private distributed ledger with restricted access for the contracting parties only.

In the case of a public distributed ledger or a private distributed ledger with an operator or with participants other than the contracting parties, information will potentially be available to all participants and/or the operator, subject to the distributed ledger's terms and conditions and software. Maintaining confidentiality would require the contracting parties to restrict access to, or sufficiently anonymise, relevant data. As a result, the content of, and possibly even the identities of parties to, transactions would need to be sufficiently encrypted. However, there is a degree of tension between maintaining confidentiality, on the one hand, and the concept of a public distributed ledger on the other hand.

A further tension exists in the field of personal data and distributed ledgers where data cannot typically be deleted in the conventional sense, as may be required by data privacy laws. Please see page 47 for a discussion of some of the key issues.

2.8 AML, CTF, KYC, anti-bribery and corruption requirements

In most jurisdictions, certain entities and persons have to comply with AML, CTF and KYC requirements set out in laws and other arrangements at a national, supranational and international level. Whilst these requirements pursue similar purposes in each jurisdiction, the details and the consequences of non-compliance may vary between jurisdictions.

Complying with AML and CTF obligations generally involves taking appropriate measures for preventing, detecting, and reporting money laundering and terrorist financing activities. KYC refers to the related obligation to identify and verify the identities of the contracting parties and certain other persons or entities, to inquire and assess the purpose of a business relationship or transaction, to continuously monitor and assess business relationships and transactions and to update documents and information collected as required. The applicable provisions may allow for outsourcing to, or reliance on, third parties to comply with these obligations. However, such outsourcing or reliance will often be subject to limits and conditions, such as the obligation to carefully select the third party or to regularly monitor and assess the performance of the obligations by the third party. Collecting certain information on contractual counterparties and other third parties may also be required under other laws. For example, anti-bribery and corruption laws may require a business to have an understanding of participants in its supply chain, or non-AML/CTF risk management requirements may need to be observed.

A note on GDPR

The EU General Data Protection Regulation ("**GDPR**")[45], which started to apply in May 2018, has brought about significant changes to EU data privacy rules and has prompted significant thinking about its impact on DLT-based systems. The GDPR rules apply in the context of storing and processing personal data – i.e., information relating to an identified or identifiable natural person, such as a name or details of a transaction in which they have engaged. The effects of the GDPR rules on DLT-based systems and smart contracts are too numerous to discuss in detail, but some of the key issues include the following:

- In relation to personal data stored on a ledger in connection with a smart contract (e.g., an individual's name and address), there is a dichotomy posed by the right of an individual to be forgotten under Article 17 of the GDPR on the one hand and, on the other, the immutability of the distributed ledger. Whilst such personal data may be cryptographically "sealed", the Article 29 Working Party, an EU advisory body, considers that such cryptographically sealed personal data is only pseudonymised and not fully anonymised. Accordingly, in its view, this type of pseudonymised personal data would remain subject to the GDPR. One practical answer may be not to store the identifying details that make data "personal" on the ledger – instead such data details should be stored outside of the ledger and be referenced by the smart contract where necessary (assuming the smart contract code has authority and is technically able to access the information). However, whilst this may sound like a simple solution, it comes with disadvantages in terms of maintaining multiple databases, the vulnerability of the off-ledger database to tampering, etc.
- DLT-based systems often have a wide geographical spread. This means that participants in the system, their computers, and persons whose data are being processed in connection with a smart contract could be located anywhere in the world. Thus, it is highly probable that personal data processed in connection with a smart contract will be subject to the regulations of various jurisdictions, and participants will need to concern themselves with not only the GDPR, including its international data transfer restrictions, but possibly also the data privacy regulations in all or many of the jurisdictions in which participants in the system, their computers, and persons whose data are being processed are located.
- To comply with the GDPR, it is crucial for the "controller" and "processor" in each DLT-based system to be identified. In general, the "controller" is the person that determines the purposes and means of processing personal data, whereas the "processor" only processes the personal data on behalf of the controller. Identifying these persons and their respective roles can be challenging, particularly given the decentralised nature of DLT-based systems and the ability of network participants to enter into smart contracts directly with each other (i.e., without the need for a common counterparty across all transactions), share resources on a peer-to-peer basis and add information to the ledger without requiring any authorisation from a central administrator. In general, any participant entering personal data in blocks of the ledger may be regarded as a "controller", under the GDPR, of the data it has provided or to which it has access through the system, unless it is a mere technology service provider supporting the system, in which case it is likely to be characterised as a processor.
- Even where personal data is not being stored on a ledger in connection with a smart contract, an individual counterparty's use of the same public key to "sign" several smart contracts may result in that counterparty becoming identifiable, as the public key is generally visible to everyone participating in the relevant DLT-based smart contracting system. Therefore, public keys may become personal data subject to the GDPR.

The challenges posed by the GDPR for personal data and DLT systems are manifold and may require legislators to identify practical workarounds if smart contracts are to be facilitated. This is particularly the case for EU jurisdictions (where the GDPR has direct legal effect) but the wide extraterritorial scope of the GDPR and geographical spread of DLT systems means that its impact is likely to be wider than this.

45 Regulation (EU) 2016/679, on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, available at: https://eur-lex.europa.eu/legalcontent/EN/TXT/?uri=celex%3A32016R0679

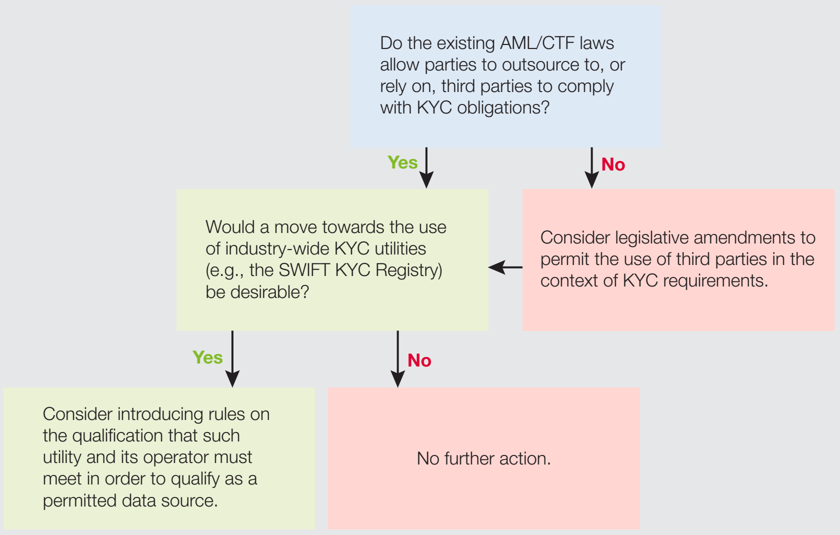
Considerations for lawmakers

As a first step, lawmakers should determine whether the existing legal framework permits outsourcing to, or reliance on, third parties to comply with KYC obligations under AML/CTF regulation. If not (or if it is not clear), lawmakers should consider introducing laws which permit the use of third parties in the context of KYC.

Where outsourcing to and/or reliance on third parties is generally permissible, lawmakers should identify limits and conditions on this use of third parties (or consider introducing appropriate limits and conditions). Typically, regulations provide that, whilst operational elements of KYC may be outsourced, the regulatory responsibility and liability remains with the regulated entity.[46]

As a positive step towards facilitating more efficient and cost-effective KYC processes, lawmakers should consider amending existing laws and regulations to facilitate the use of industry-wide KYC utilities (such as the SWIFT KYC Registry referred to below). In particular, lawmakers may need to adjust existing laws to provide that industry participants may not only outsource the operation of KYC but could also discharge their regulatory responsibility by relying on such utilities to perform the necessary KYC – specifically, in the context of DLT-based smart contracts, where such utilities are established and supported, they should either be run on the distributed ledger itself or should be supported as a permitted data source that is accessible via an oracle.

Where such utilities are to be permitted, lawmakers will also need to create rules governing qualifications that the utility and the utility's operator or administrator will have to meet in order to qualify as a permitted data source.


Do the existing AML/CTF laws allow parties to outsource to, or rely on, third parties to comply with KYC obligations?
Yes
No
Would a move towards the use of industry-wide KYC utilities (e.g., the SWIFT KYC Registry) be desirable?
Consider legislative amendments to permit the use of third parties in the context of KYC requirements.
Yes
No
Consider introducing rules on the qualification that such utility and its operator must meet in order to qualify as a permitted data source.
No further action.

If the entry into a smart contract is not automated (i.e., in a non-Concluding Model), it will generally be possible to carry out KYC in the same (or a similar) manner as for a non-smart contract. However, in some cases, the performance or enforcement of a smart contract may itself trigger new KYC obligations – for example, in the case of a wire-transfer exceeding certain limits. Hence, the software will need to be able to identify such triggers and to execute actions only if the relevant KYC requirements are met. In case of the Concluding Model, the software will need to ensure that contracts are concluded only if the relevant KYC information has been obtained and assessed. Depending on the software, the information required, and the data sources available (for example, a company register or a transparency register of economic beneficiaries), human input may be necessary.

Obtaining, assessing and updating relevant information is certainly more challenging if the smart contract runs on a distributed ledger. In case of a public distributed ledger, subject to its terms and conditions, it may be difficult to even identify the other contracting party, let alone obtain and assess the relevant information. Hence, complying with KYC-related obligations may require outsourcing of KYC operations to, or reliance on information obtained, or steps taken, by a third party such as the operator of a distributed ledger (if there is one). The third party would need to meet applicable requirements and would need to be technically able to collect and assess all relevant information. However, such outsourcing or reliance will not

46 For example, this is the case under Directive (EU) 2015/849 of the European Parliament and of the Council of 20 May 2015 on the prevention of the use of the financial system for the purposes of money laundering or terrorist financing (MLD4).

always be possible or permissible under the applicable requirements and may not be compatible with the decentralised and open structure of such distributed ledger. Alternatively, the participants may share all KYC information with all or certain other distributed ledger participants. For example, if all participants of a private distributed ledger are banks, they may agree among themselves (and, if relevant, each bank with any client it acts for) to share all relevant information among the participants of the private distributed ledger. However, sharing such information may not comply with the parties' intentions and may also violate applicable confidentiality and/or data privacy obligations. SWIFT has been considering how to address some of these issues as part of its work to develop an industry-wide KYC utility (the "**SWIFT KYC Registry**").[47]

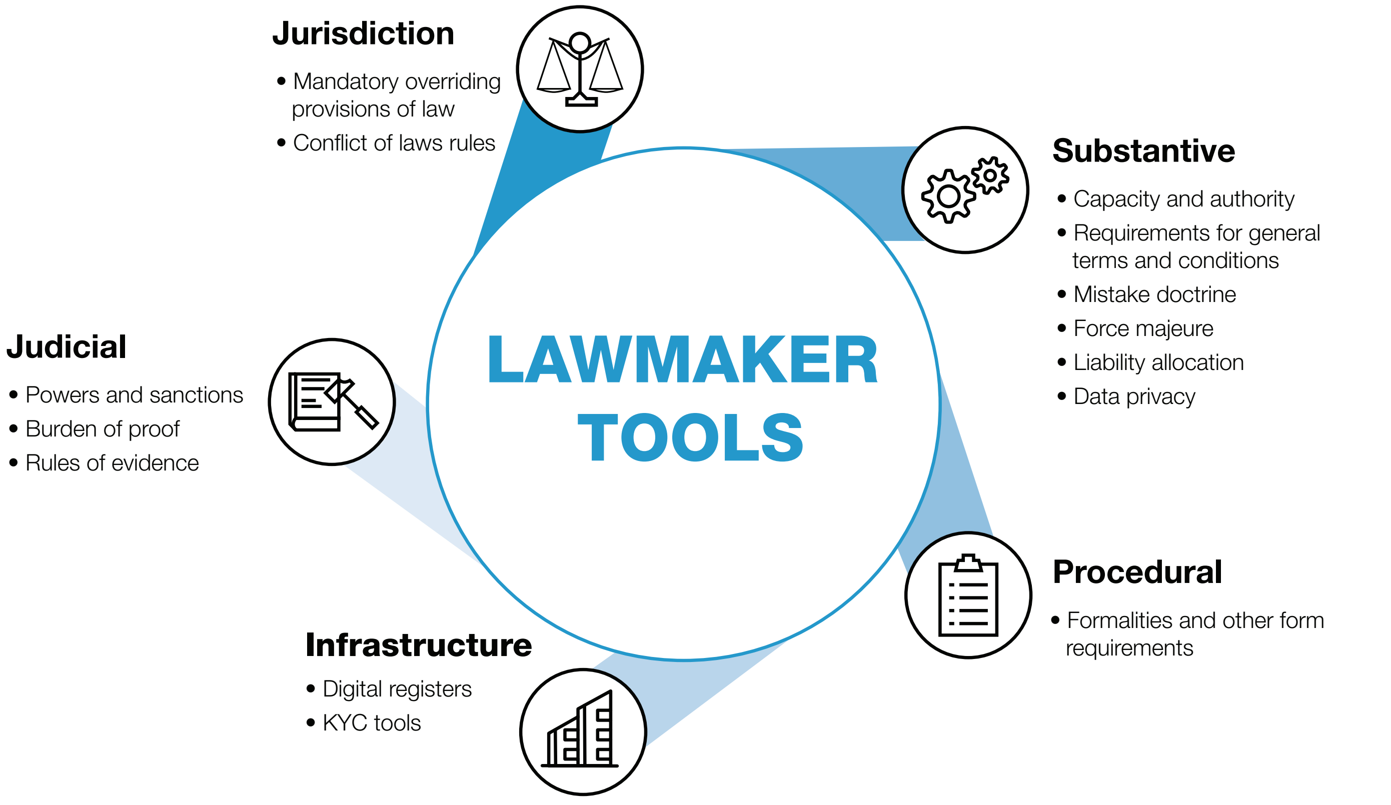
Use of smart contracts must also be sufficiently covered by parties' internal AML/CTF risk management measures and safeguards (which may add on to other general risk management and related requirements that apply, for example, to banks). As a result, the smart contract software may need to be linked to a party's internal software which is used for this purpose. This may be more difficult in the case of a distributed ledger, as participants would need to be able to connect their existing in-house systems to the distributed ledger. In some cases, these in-house systems may be old or use outdated technology, and so there may be technical challenges in building oracles that would allow these legacy in-house systems to access and analyse any data which is or is to be recorded on a distributed ledger, and to prevent any transactions that the in-house systems do not approve.

47 For further information about SWIFT's ongoing work on the KYC Registry, see: https://www.swift.com/oursolutions/compliance-and-shared-services/financial-crime-compliance/our-kyc-solutions/the-kyc-registry

CONCLUSION

Whilst existing legal frameworks may allow for use of smart contracts, this may not always result in a desirable outcome from a policy perspective. Parties may also face practical challenges in complying with existing legal or regulatory requirements, such as form requirements, in the context of smart contracts. If not addressed, these types of issues could prevent or slow down the widespread adoption of smart contracts. The recommendations set out in this paper should help lawmakers get more comfortable with the formulation of the legal and regulatory framework, which should facilitate the use of smart contracts, while ensuring adequate protection of consumers.

Lawmakers have various tools at their disposal to address these issues and create a legal and regulatory environment that promotes the appropriate use of smart contracts, as described in this paper and summarised in the diagram below. If these tools are used effectively, we expect that smart contracts could start to become mainstream in the next few years.



European Bank
for Reconstruction and Development

CLIFFORD
CHANCE

About EBRD's Legal Transition Programme

EBRD's Legal Transition Programme helps to establish laws, regulations and institutions that support the development of a predictable investment climate in our region, which includes central Asia, central Europe and the Baltic states, eastern Europe and the Caucasus, south-eastern Europe, the southern and eastern Mediterranean and Turkey. Its lawyers advise lawmakers and regulators on numerous areas of commercial and financial law, including legal and regulatory aspects of crowdfunding, blockchain, including smart contracts, and payment services.

Contacts

Jelena Madir
Chief Counsel
E: madirj@ebrd.com

Ammar Al-Saleh
Senior Counsel
E: alsaleha@ebrd.com

About Clifford Chance

Clifford Chance's global fintech offering extends across market-leading practices in Europe, the US, the Middle East, Africa, Asia and Australia. Clifford Chance's global team of fintech experts advises the clients that are shaping the converging financial and technology sectors. We have developed fintech expertise across M&A, technology, data privacy, financial regulation, capital markets, tax, antitrust, intellectual property and cyber security. Our roster of clients is also varied and includes the financial institutions that focus on technology, technology companies, emerging fintech and insurtech players, startups and consortia.

Contacts

Dr. Marc Benzler
Partner
E: marc.benzler@cliffordchance.com

Peter Chapman
Senior Associate
E: peter.chapman@cliffordchance.com

Laura Douglas
Senior Associate PSL
E: laura.douglas@cliffordchance.com

Dr. Kai Krieger
Senior Associate
E: kai.krieger@cliffordchance.com

Disclaimer

CLIFFORD
CHANCE

European Bank
for Reconstruction and Development

J20182008181602