7

Section 7: Smart Contracts and Data Governance

Anne Rose (Mishcon de Reya LLP) and Marc Piano (Harneys (Cayman Islands))

PART A: Smart Contracts

Introduction

Smart contract technology, the process of digitising legal contracts and/or transactions using any combination of Smart Legal Contracts, Smart Contract Code, Internal Models and External Models as defined below, theoretically permits any written legal contract to be digitised into self-executing code. In turn, traditional transaction flows can be digitised in whole or in parts, using tokenised representation of transactional objects where required.

Several in-house and public projects already permit digitisation of contracts and transactions at least in part. Some of these projects are explored in this guidance. As at the date of this guidance, projects range across open and closed systems, using a combination of open source and proprietary platforms and processes. Each project and the nature of the legal contracts and transactions involved has unique requirements and objectives. Taken together with the benefits and drawbacks of automating elements of English law, each project approaches the use of smart contracts in digitising and automating legal contracts and transactions differently.

This section was written with a coding sub-group and with the help of expert evidence for which the Group is grateful. The scale, level of development and public accessibility varies for each of the projects explored. However, all experts who gave evidence on their projects demonstrated development far beyond proof of concept and are well placed to give evidence on the issues forming the subject of this guidance.

Objectives of the coding sub-group

The coding sub-group has four objectives:

1. Identify the extent to which different types of existing, primarily document-based, legal transactions are and/or may in future be carried out by or through smart contracts, and/or DLT technology and/or cryptoassets (in whole or in part);

2. Identify the current and/or future role of legal professionals in such transactional processes with a focus on the technical elements;

3. Identify, using recent examples, transactional flow and parties involved from a technical perspective; and

4. Identify, using recent examples, areas of risk, opportunity, responsibility, liability and value add for legal professionals and law firms in respect of the technical elements of such transaction processes.

Experts and evidence

The Group convened on four evidence telephone sessions between November 2019 and February 2020, at which expert evidence was heard from each of:

— **Niall Roche (Head of Distributed Systems Engineering, Mishcon de Reya LLP)**

— **Ciarán McGonagle (Assistant General Counsel, International Swaps and Derivatives Association (ISDA))**

— **Akber Datoo (Founder and CEO, D2 Legal Technology (D2LT))**

- **Aaron Wright (Professor, Cardozo School of Law and Co-Founder, OpenLaw)**

Definitions

Drawing from definitions provided by Ciarán McGonagle:

- **Smart Legal Contract (SLC):** a written and legally enforceable contract where certain obligations may be represented by or written in code; and

- **Smart Contract Code:** code that is designed to execute certain tasks if pre-defined conditions are met. Such code may or may not be intended to give effect to legal provisions or have legal ramifications. In some cases, such code is required for the internal function of an SLC, or communication between smart contracts (whether pursuant to contractual provisions or not).

Two potential SLC models:

- **Internal Model:** the provisions that can be performed automatically are included in the legal contract, but are rewritten in a more formal representation than the current natural language form; and

- **External Model:** the coded provisions remain external to the legal contract, and represent only a mechanism for automated performance.

Digitising legal contracts and/or transactions may use any combination of SLCs, Smart Contract Code, Internal Models and External Models.

Findings

The findings of the Group are divided into four parts:

1. Advantages and disadvantages of SLCs;

2. Data governance;

3. Digitisation considerations; and

4. Additional comments.

1. Advantages and disadvantages of SLCs

In summary, the advantages and disadvantages of SLCs are:

Advantages

- **Increased accuracy and potential transparency of contractual terms:** the logic and information in each contract may be visible to all participants in the blockchain network (although, where relevant, some or all contractual terms can be made confidential, visible only to the transacting parties and hidden from the wider network). This transparency combined with automatic execution facilitates an environment of trust and removes manual errors.

- **Efficiency in automating performance:** standard-form SLCs can be written so as to permit limited negotiation of commercial and legal terms. This is particularly beneficial for high-volume contracts and transactions. Negotiated contracts and related transactions can be quickly deployed and concluded by making the assembly of contracts dependent on variables or computable logic provided by the contracting party. Tokenised value or objects can be quickly transferred with an automatically generated audit trail.

- **Less scope for misinterpretation or competing interpretations:** subject to good data governance, standardised definitions and provisions in SLCs will automatically execute in accordance with their agreed terms. Where provisions of an SLC or elements of a transaction occur off-chain, appropriate on-chain or off-chain dispute resolution mechanisms can resolve issues arising from competing interpretations more efficiently than traditional methods, the availability and applicability of on-chain and off-chain dispute resolution methods are explored in more detail at Section 12.

- **Potential evidential value of deployed contracts, electronic outputs and audit trail of tokenised representations of subject matter or value:** computer code is more definitive, precise and immediate than traditional paper-based contracts. Electronic outputs – such as documents, inter-contract activity and external outputs – together with automatic generation of an audit trail of transfers of tokens, can help to minimise disputes around fulfilment of contractual terms and ownership of title.

- **Scope for efficient dispute resolution using novel and inherent dispute resolution mechanisms:** elements of a contract or transaction in dispute may be isolated and resolved quickly and efficiently without necessarily affecting the wider contract or transaction. Importantly, a smart contract can escrow or parties can pre-authorise the transfer of funds at issue and an arbitrator can render a decision and direct payment to one or both parties, thereby decreasing the need for post-litigation enforcement proceedings.

- **Interoperability:** contractual data can be imported and exported into an SLC, which can be useful to keep track of contracts and manage risk. If deployed at scale, for example in relation to derivatives contracts where the collection, storage and dissemination of data is imperative to assessing risk, it is conceivable that a particular jurisdiction utilising SLCs would be able to have a more detailed view of the economy by analysing and aggregating contractual information in an anonymised manner.

Disadvantages

- **Over-automation:** not all elements of a legal contract that can be automated should be, such as provisions over which parties may wish to retain discretion to amend or waive from time to time. Over-automation due to poor digitisation planning or otherwise may inadvertently restrict the flexibility that is often expected and exercised over some contractual provisions, and expose parties to unintended risk.

- **Full automation is not always possible:** some terms implied by English law which require subjective assessment of the parties' intentions, or which must allow external intervention or determination, are not easily automatable. Attempts to do so may result in contracts being unenforceable or not fully reflecting the intentions of the parties. Digitisation scoping must seek to identify and address these issues.

- **Unsuitable contracts or transactions:** highly complex, one-off transactions contingent on many external parties and factors may not be suitable for automation, along with more “relational contracts”, which are assembled by the parties to memorialise an agreement to engage in commerce as opposed to precisely defining the rights and obligations of members.

- **Systems interoperability:** where there are SLCs and transactions dependent on external actors or systems, it may not be possible to fully automate or complete electronically. Proper digitisation considerations will identify and address these issues and facilitate off-platform fulfilment of relevant contractual provisions.

- **Inflexibility to amend contracts or waive provisions due to immutability:** where an automated term is expressed incorrectly, it may be that parties are unable to prevent or reverse performance, particularly given the immutability of DLT records.

- **Necessity to pre-fund accounts due to the automation of movements of value:** while SLCs have the potential to be able to automate movements of value (for example, collateral movements in the context of collateralised derivatives agreements) and so create several operational efficiencies, in order to achieve this automation it may be necessary for counterparties to pre-fund specific accounts/wallets which are linked to the smart contract code. This may not be practical or efficient in all markets, as it may mean that any such pre-funded value would not be capable of being used by its owner while it remains in the pre-funded account.

- **The "oracle problem":** to achieve the extensive automation which SLCs could be capable of, many SLCs need to be able to rely on objective sources of external data which both parties can trust (the so-called "oracle problem"). For example, with respect to an SLC which is designed to trigger a payout in the event that one party to a contract enters into insolvency proceedings, the smart contract would need to rely on an external data point which is capable of accurately confirming that a winding-up petition (or equivalent) has indeed been filed in relation to that party. These oracles may not always be available.

Data governance

A working definition of data governance from the Data Governance Institute is *"the exercise of decision-making and authority for data-related matters"*. By extension, data governance involves marshalling and unifying consistency and accuracy of data used in digitisation projects, such as defined terms, mechanical clauses, representations and warranties, covenants, standards, and rights and obligations.

Data governance forms a fundamental prerequisite of any digitisation project. Data governance failure can result in contractual uncertainty, legal or regulatory breaches, failure of automated provisions and unnecessary disputes arising.

Any digitisation project should therefore involve a data governance audit at the outset. This can include an internal glossary to ensure common standards within an organisation, an audit of any data subject to digitisation, standardisation of relevant data, and portability across documents and platforms. In particular, legal agreement terms play a crucial role in respect of smart contracts, and any data inputs and outputs need to have appropriate data governance to ensure certainty and completeness of contractual terms (which in the context of a smart contract, can often manifest themselves through data variables).

Effective data governance measures will assist in efficient contract and transaction digitisation and reduce risk to all parties.

More information on data governance is set out in Part B of this Section.

Digitisation

Stakeholders (being transaction parties, businesses, and service providers including law firms or other intermediaries) in seeking to wholly or partially automate legal contracts and transactions undertake a form of digitisation project.

General scoping and project management considerations for digitisation projects will apply. These considerations are beyond the scope of this guidance, and detailed resources on the topic are already widely available.

However, the sub-group does recommend additional considerations specific to legal contract and transaction digitisation.

Choice of platform

Digitisation need not necessarily involve the development of an entirely new platform or protocol. The sub-group heard evidence from each of ISDA, Mishcon de Reya and OpenLaw, each of which utilised different approaches to digitisation. ISDA has developed an industry-standard, digitised representation of derivatives transactions and events called the ISDA Common Domain Model. Mishcon de Reya, as part of the "Digital Street" project, utilised the open source Accord Project. OpenLaw developed a protocol to allow digitisation, execution and tokenisation of any legal document.

The requirements of contractual parties and advisors for a particular contract or transaction, or series thereof, will influence the approach that is right in the particular circumstances.

We would caution that the complexity and risks inherent to a digitisation project lend to a strategic and longer-term approach in platform choice and digitisation generally. It may not be efficient, for example, to digitise a contract or transaction specific to one particular platform if the likely volume or subsequent demand for digitisation lends to development of an in-house protocol or use of a different platform in future.

Finally, choice of platform should include due diligence on use of third-party protocols (whether open source or proprietary, and permissioned (private) or permissionless (public)) to assess suitability and risk relevant to the particular transaction(s) and intentions of the parties. As this technology space continues to evolve, regard should be had to development roadmaps, and continued suitability and support availability (where relevant) across the intended lifespan of the transaction and possible subsequent changes in relevant law and regulation, particularly for relatively novel protocols or offerings. Where a digitisation project includes critical reliance on third party services beyond a protocol itself – such as use of oracles – the role of those services and any recourse to responsible entities should be carefully considered. This may include analysis of sources, data and transaction flows and any standard terms of use of each third-party service. Reviews of terms and service should focus in particular on any representations and warranties as to service availability, accuracy and verification (or disclaimer thereof) of data flows where input data is sourced from third parties, liability clauses, and governing law, jurisdiction and dispute resolution. Where appropriate, it may be prudent to negotiate with critical third-party service providers to contract on bespoke terms.

Effective and efficient digitisation

Consideration must be given to which elements of a legal contract and transaction flow can and should be digitised, and which should not. It is not feasible to develop a set of general best practice guidelines, as these will be specific to the contracts, transactions and project objectives in each case. We can, however, provide examples of the different approaches taken from the evidence provided to the sub-group.

<u>ISDA</u>

ISDA's evidence focused on the work they are doing to develop a foundation for the development of smart derivatives contracts. ISDA's approach involves distinguishing between operational aspects (i.e. mechanical elements such as delivery or payment) and non-operational aspects (relating to time, or rights and obligations) within a derivatives contract.

Whilst many elements of derivatives contracts lend to digitisation, many do not. These include elements common to many contracts, such as representations and warranties, document delivery obligations, payment obligations subject to withholding, set-off or other deductions, transfer or assignment of contractual rights, events of default and insolvency events.

In its presentation to the Group, ISDA noted that: "This complexity and potential need for human intervention in respect of certain events, such as the triggering of an Event of Default, may mean that it may never be efficient or desirable to automate certain parts of a derivatives contract, even if it were technically possible."

D2LT – ISDA Clause Taxonomy and Libraries

D2LT's evidence detailed, inter alia, the legal agreement digitisation work it had completed for ISDA, designed to work together with the ISDA Common Domain Model. One of the issues the OTC derivatives industry faces was the huge variation in language of legacy ISDA Master Agreements between market participants. Although in some cases the language of particular clauses achieved different business outcomes, in many cases, the substance of the business outcome was identical – only the form/style of the legal drafting differed. This offered a significant impediment to efforts to automation, be it: (i) generation of new agreements; (ii) management of the contractual obligations contained within the agreements downstream (e.g. liquidity and collateral management); or (iii) use of AI and smart contract applications. Accordingly, the ISDA Master Agreement Clause Taxonomy was developed, which defines the various clauses contained within an ISDA Master Agreement, and enumerates the main business outcomes that parties negotiate within these agreements (determined with regard to twelve pre-defined design principles). Such standards are necessary to facilitate the automation of legal contractual obligations.

Subsequent to the D2LT evidence, D2LT have successfully completed similar work for two other capital markets trade associations, ISLA (The International Securities Lending Association) and ICMA (The International Capital Market Association) to create similar clause taxonomies and libraries for the GMSLA and GMRA documentation respectively. Furthermore, use cases have been identified across these trade associations to utilise these standards, such as in the automation of the close-out netting determination process[173], including the issuance of an NFT to represent legal opinions relied upon by the prudentially regulated trade association members for regulatory capital purposes.

3. "Digital Street" project

Similar considerations formed part of the development of the "Digital Street" project for HM Land Registry, through the open source Accord Project ecosystem. The Digital Street project furthers HM Land Registry's ambition of becoming the world's leading land registry for speed, simplicity and an open approach to data through the use of blockchain technology to develop a simpler, faster and cheaper land registration process.

The project did not digitise the Standard Conditions of Sale owing to their complexity. As an alternative, the Accord Project permits digitisation of clauses that are independent of any particular distributed ledger, enabling global interoperability. The project is therefore able to digitise such clauses, as they are conducive to digitisation, while enveloping compliance with, and fulfilment of, non-digitised clauses offline pursuant to established conveyancing protocols.

The project further allows any disputes to be resolved offline, and the outcome to be recorded within the digitised transaction flow. As the project develops, the intent is to make clear to the parties which elements of the contract and transaction are fulfilled online and which will occur offline, without requiring separate processes running in parallel and fitting within the wider digitisation envelope.

4. OpenLaw

OpenLaw has developed an open source protocol for contract digitisation, execution, workflow management and tokenisation.

The protocol permits any legal document to be digitised according to the requirements of the parties. This approach affords flexibility for the parties to determine digitisation of contracts and transactions according to their agreed

[173] https://arxiv.org/abs/2011.07379 - Datoo A, and Clack CD (2021): Smart close-out netting

parameters for any particular transaction. However, we observe that this requires such parties and their legal counsel to have undertaken diligent digitisation scoping on a contract and transaction basis to ensure that digitised contracts and transactions are legally enforceable and commercially viable.

While OpenLaw is aimed at lawyers, for the time being they must be trained or be self-taught in the use of the mark-up language necessary to create programmable legal agreements capable of execution (e.g. basic logic actions and calculations). The solution currently utilises the Ethereum platform to manage the contract execution actions, but can be generalised to other systems and does not need to rely on a blockchain. On execution, the smart contract related evidence, if incorporated into an agreement, is recorded and managed on the Ethereum blockchain.

The solution provides contract management support and automatically saves contracts on third-party cloud hosting platforms such as Dropbox, Google Drive, and Microsoft One Drive.

OpenLaw provides a public "library", but also permits parties to run their own private instance to enable peer-to-peer contracting. Parties that run an OpenLaw instance can pass contractual information between one another without the need to share that information with third parties.

Any limitations of the proprietary mark-up language were not discussed in the evidence session, but users of OpenLaw must give careful consideration to the use of the mark-up language to effect complex multi-party agreements.

Additional comments

Legal contracts and transactions best suited for smart contract digitisation are those which:

- already occur at scale, using standard-form documents and standardised transaction flow;
- operate within a range of known or knowable variables and events, each of which can be accommodated during the digitisation and automated transaction process;
- can access external third-party data (through sources known as "oracles") available in a standard and processable form from trusted sources, where required; and
- produce deliverables or outputs in forms that can be accommodated as part of the digitisation process.

Legal counsel will play a central role in digitisation of contracts or transactions as both counsel and likely project managers. They will therefore be required to fully scope any digitisation project from both a legal and project management perspective. This will involve choice of platform, extent of digitisation, anticipating any technical or legal issues which may arise, and identification and coordination of stakeholders. As an additional safeguard, a well-scoped independent code audit can assist with objective confirmation that the code-dependent constituent elements give proper effect to legal and commercial terms, identify unintended mechanics and security risks, and generally provide comfort to all relevant parties that the code implements the desired transaction according to the agreed terms that reflect the parties' intentions.

Legal counsel should always consider whether digitisation can fully allow implied terms, application of principles derived from precedent, facilitation of industry or market standards, and the flexibility to amend contracts where required due to changes in law, regulation or where contingent on external input, such as third-party expert determinations.

Inadequate digitisation scoping may risk breach of contract or frustration due to unanticipated issues arising from automatic execution. This may heighten transaction risk for the parties and unnecessarily strain commercial relationships.
Legal counsel may be exposed to liability when facilitating a digitised contract or transaction where full consideration has not been given to the digitisation and transaction flow process, and unintended consequences arise. We note that there is no judicial determination on these specific points as at the date of this guidance. We do not offer any legal opinion on likely risk or determination on these points, however the changing risk landscape for lawyers is addressed in more detail in Section 12.

Automating transaction elements best concluded off-chain

As seen above, digitisation is not an "all or nothing" process and is not without risk. Digitisation of contracts and transactions can involve a hybrid partial digitisation and off-chain fulfilment of some contractual provisions not suitable for digitisation. For some contracts and transactions, this hybrid approach may be unavoidable to ensure contractual soundness and proper reflection of commercial intent.

This means that, where relevant, any digitisation must be able to facilitate and record off-chain compliance (or breach and any relevant remedies) as part of the digitised contract and transaction flow. This influences digitisation scoping, choice of platform, transaction flow and record generation. In some cases, the additional work required for full or partial digitisation may outweigh any time and cost efficiencies gained from digitisation, particularly for highly complex or one-off transactions.

Dispute resolution considerations

As at the date of this guidance, numerous on-chain dispute resolution mechanisms are available. These may have the equivalent effect of an arbitration clause in a traditional contract.

However, any digitisation must carefully consider whether these mechanisms provide sufficient scope to resolve the full range of potential disputes that may arise in a digitised contract or transaction.

The soundness and enforceability of these mechanisms has not yet been challenged or given judicial consideration. For example, mechanisms that are only able to determine digitised matters and not off-chain matters, or are contingent on pre-appointed arbitrators who are no longer available, may be open to challenge.

Reliance on any dispute resolution mechanism must also consider the ability to enforce any decisions issued through them, as well as any scope for appeal. Unlike traditional arbitration protocols, there is also no recognised set of clauses for proper incorporation, operation, appeal or enforcement.

Further, the novel nature of these mechanisms may themselves be the source of dispute, increasing legal costs and risk for both parties.

As at the date of this guidance, we consider that on-chain dispute resolution mechanisms lack any recognised standards or judicial treatment which might make them a viable alternative to traditional dispute resolution options. Both on-chain and off-chain dispute resolution mechanisms are addressed in more detail in Section 12.

Part B:
Data governance requirements for smart contracts

Akber Datoo (D2 Legal Technology (D2LT))

Introduction

The potential of smart contracts has attracted a lot of attention and excited many. By relying on a DLT such as a blockchain, it is possible to run code reflecting contractual arrangements between parties that is resilient, tamper-resistant and autonomous.

Smart contracts extend the functionality of DLT from storing transactions to "performing computations".[174]

Indeed, it has been said that these may create contractual arrangements that are far less ambiguous than agreements written in legal prose, due to the fact that their performance is contained within the very essence of the smart contract, rather than being a separate step, as is the case with "traditional" legal contracts. However, even leaving aside the challenge that the smart contract code may not be in a human-readable form and may instead create standardised contracts that few are able to truly understand[175], the data governance challenges behind creating correctly performing smart contracts should not be underestimated, and form an area that lawyers will need to focus on very carefully.

What is a smart contract?

At a very simple level, smart contracts are coded instructions which execute on the occurrence of an event. However, there is no clear and settled meaning of what is meant by a smart contract. The idea of smart contracts was first perceived in 1994 by computer scientist and legal theorist, Nick Szabo, who defined it as "a set of promises, specified in digital form, including protocols within which the parties perform on these promises". However, at the time, smart contracts remained a somewhat abstract term and of limited value, as they ultimately relied on stakeholders trusting another entity to execute the smart contract. The advent of DLT and blockchain has enabled smart contracts to come back to the forefront of development and innovation, since they rely on consensus algorithms rather than trust in an intermediary. Taking a well-known example, the Bitcoin blockchain is technically a limited form of smart contract whereby each transaction includes programs to verify and validate a transaction (each being, effectively, a small smart contract).

For the purposes of this Section and as a foundation on which to base the discussion, we use the Clack et al. definition of a Smart Contract:[176]

> *"A smart contract is an automatable and enforceable contract. 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".*

This definition is broad enough to encapsulate a wide spectrum of smart contracts, including both types identified by Josh Stark, namely (i) "smart code contracts" (where legal contracts or elements of legal contracts are represented and executed as software); and (ii) "smart legal contracts" (where pieces of code are designed to execute certain tasks if predefined conditions are met, with such tasks often being embedded within, and performed, on a distributed ledger).

Smart contracts offer event-driven functionality triggered by data inputs (which may be internal or external), upon which they can modify data. External data can be supplied by "oracles" (trusted data sources that send data to smart contracts). Smart contracts can track changes in their "state" over time, and can act on the data inputs or changes in their state, resulting in the performance of contractual obligations.

It should be noted that where smart contracts are implemented on a DLT such as a blockchain, they natively already provide for a degree of data quality assurance with respect to the data they store. For example, on a blockchain, hashes are used to link the blocks on the blockchain preventing tampering of the data, and cryptographic signatures are used to provide for provenance and non-repudiation. Smart contracts cannot directly

[174] Nick Szabo, 'Smart Contracts: Building Blocks for Digital Markets' (Extropy: The Journal of Transhumanist Thought, 1996) vol 16

[175] Smart contracts are usually classified as fitting into either the "External Model" or the "Internal Model". In the case of the former, the legal contract remains in the traditional agreement form, but external to this legal contract, certain conditional logic elements of the contract are coded to occur automatically when relevant conditions (based on data inputs) are satisfied. In contrast, with the "Internal Model", certain conditional elements of the legal contract are rewritten in a formal logic representation, and this logic is executed automatically based on the data inputs to that logic.

[176] Clack et al, 'Smart Contract Templates: Foundations, Design Landscape and Research Directions' (Barclays Bank, 3 August 2016) <http://www.resnovae.org.uk/fccsuclacuk/images/article/sct2016.pdf> Accessed 19 May 2020

query the distributed ledger to retrieve data – they only have access to the payload of those transactions explicitly directed to them as addressees, to data stored in their own, local variables, or to data held by other smart contracts and made available through suitable functions. Also, smart contracts cannot access data (for example, through an API or querying an external database) outside the blockchain, since otherwise different results might be obtained with the passage of time – therefore causing issues in respect of repeatability. Therefore, data required for the operation of a smart contract is obtained through the use of an oracle smart contract (an "oracle"), which, using standard transactions that are recorded on the distributed ledger, allow external data sources to push data in (either for example, upon explicit solicitation or periodically)[177].

The elevated role of data and data governance in smart contracts

In many ways, smart contracts are similar to today's written contracts, in that to execute a smart contract, one must also achieve a "meeting of minds" between the parties.[178] Once this meeting of minds has been reached, the parties memorialise it, which might be triggered by digitally signed blockchain-based transactions.

A traditional legal agreement will typically contain various details of events which the parties have agreed will result in certain consequences, and typically an obligation on a party to perform some action. By way of example, it might provide that:

> *"if the rate of defaults on the underlying portfolio exceeds 2%, the protection seller shall make a payment of £1,000,000 to the protection buyer".*

Such contractual obligations of course require a certain degree of certainty and specificity in order to ensure the "meeting of minds" required for the formation of a contract.

Smart contracts do, however, differ from traditional legal agreements through the smart contract's ability to enforce obligations through autonomous code. Promises in smart contracts, such as the example given above, are harder to terminate – especially in cases where no one single party controls a blockchain, and there may therefore not be any straightforward manner in which execution can be halted. Where transactions represent real-world business interactions between parties collaborating on a complex business process, the specific facts surrounding the operation of the business process become critical to the successful running of that business process, and accordingly, the data quality of those facts is key.

In the context of a smart contract, factual matters relevant to the contractual obligations are likely to be automatically assessed, removing the normal human assessment of the triggering event. In the example above, this would be the question of whether the rate of defaults has exceeded 2%, which may simply be an input from another system.

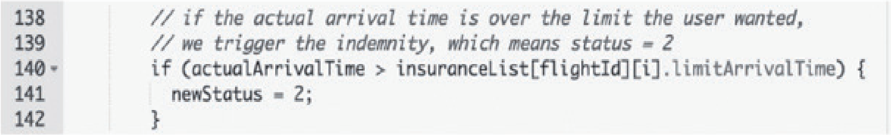
It is the fact that smart contracts seek to automate performance, and therefore need to automate the process of applying fact to a contract at hand, that elevates the importance of data governance from the traditional legal agreement context. A smart contract operates through Boolean logic – a form of mathematical logic that reduces its variables to "true" and "false".

AXA's "Fizzy" application is an example of a smart contract application for flight insurance, whereby the terms of the contract between the holder of the insurance and AXA are based around insuring against a flight delay of greater than 2 hours. The smart contract operates on the Ethereum blockchain network, and it continuously checks data from oracles in real time. Once the delay exceeds 2

[177] "Data quality control in blockchain applications", Cinzia Cappiello, Marco Comuzzi, Florian Daniel and Giovanni Meroni – available at: https://www.researchgate.net/publication/335399935_Data_Quality_Control_in_Blockchain_Applications

[178] Stephen J Choi and Mitu Gulati, 'Contract as Statute' (Michigan Law Review, 2006), Vol 104

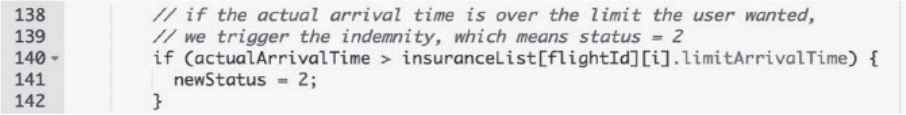
hours, the compensation terms are automatically triggered and given effect. Putting this into colloquial Boolean algebra, “if the plane is late by more than 2 hours, then compensation must be paid out”. The key code representing this logic is shown below[179] (note that the variable limit ‘limitArrivalTime’ is defined as 2 hours elsewhere in the code).



```
        // if the actual arrival time is over the limit the user wanted,
        // we trigger the indemnity, which means status = 2
        if (actualArrivalTime > insuranceList[flightId][i].limitArrivalTime) {
          newStatus = 2;
        }
```

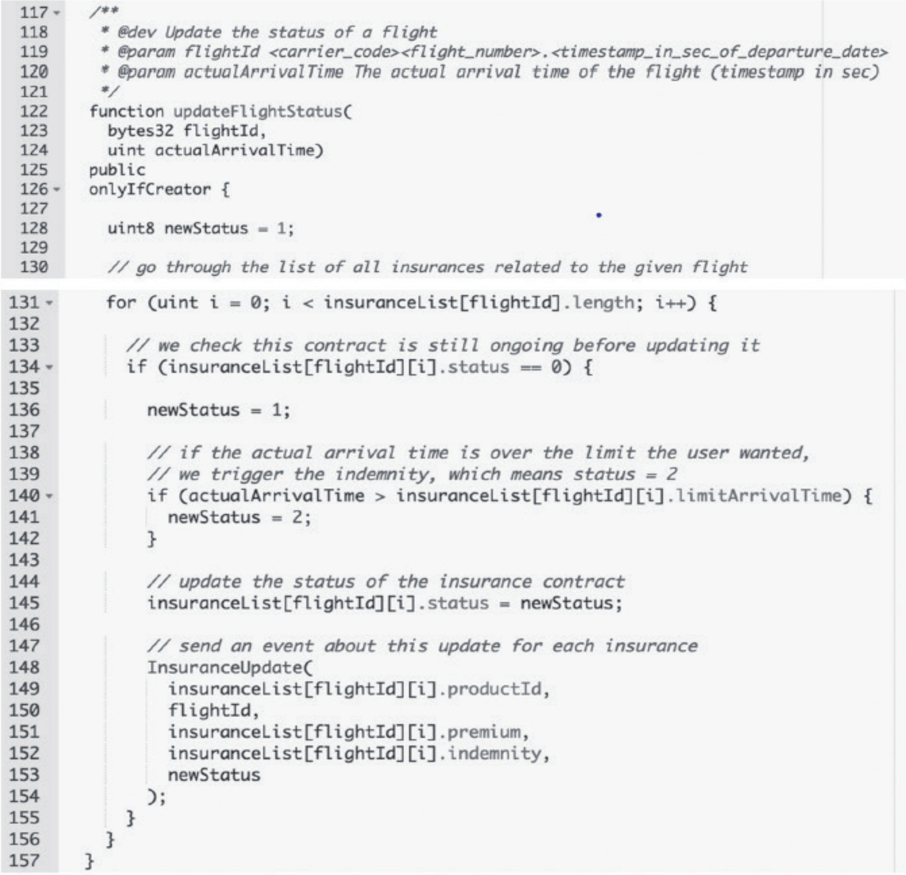
Figure 12.2 The core logic code for the Fizzy smart contract application

The core logic code for the Fizzy smart contract application



```
        // if the actual arrival time is over the limit the user wanted,
        // we trigger the indemnity, which means status = 2
        if (actualArrivalTime > insuranceList[flightId][i].limitArrivalTime) {
          newStatus = 2;
        }
```

Figure 12.2 The core logic code for the Fizzy smart contract application



```
/**
 * @dev Update the status of a flight
 * @param flightId <carrier_code><flight_number>.<timestamp_in_sec_of_departure_date>
 * @param actualArrivalTime The actual arrival time of the flight (timestamp in sec)
 */
function updateFlightStatus(
  bytes32 flightId,
  uint actualArrivalTime)
public
onlyIfCreator {

  uint8 newStatus = 1;

  // go through the list of all insurances related to the given flight
  for (uint i = 0; i < insuranceList[flightId].length; i++) {

    // we check this contract is still ongoing before updating it
    if (insuranceList[flightId][i].status == 0) {

      newStatus = 1;

      // if the actual arrival time is over the limit the user wanted,
      // we trigger the indemnity, which means status = 2
      if (actualArrivalTime > insuranceList[flightId][i].limitArrivalTime) {
        newStatus = 2;
      }

      // update the status of the insurance contract
      insuranceList[flightId][i].status = newStatus;

      // send an event about this update for each insurance
      InsuranceUpdate(
        insuranceList[flightId][i].productId,
        flightId,
        insuranceList[flightId][i].premium,
        insuranceList[flightId][i].indemnity,
        newStatus
      );
    }
  }
}
```

Figure 12.3 An example of the Solidity smart contract coding language (taken from the Fizzy smart contract)

An example of the Solidity smart contract coding language (taken from the Fizzy smart contract)

In many ways, the automated performance feature of smart contracts extends the need for “certainty and completeness of terms of a contract”, to “certainty and completeness of data specification of data variables inherent in a smart contract” (be

[179] Akber Datoo, ‘Legal Data for Banking: Business Optimisation and Regulatory Compliance’ (John Wiley, 2019)

this data input or contractual state data). This can only be addressed through the governance of such data.

Data governance

The term ‘data’ is typically used to refer to facts or pieces of information that can be used for reference and analysis. A phenomenal amount of data is created, stored and processed in the ordinary course of day-to-day life and business – and its proliferation is ever increasing. These are likely to form key data inputs into the conditional logic of a smart contract. However, the quality (typically through the lens of definition, accuracy and timeliness) of such data needs to be considered as this will likely impact the functioning of a smart contract and any automated performance, noting that this is not simply a question of whether the data is accurate, but must be viewed through a variety of data quality lenses such as timeliness, consistency and precision.

As a result, smart contracts need to ensure an appropriate data governance framework is in place in relation to any data variables relevant to it. This is a formalisation of authority, control and decision making in respect of these data variables. This is unlikely to be in the complete control of the parties to a smart contract, however there ought to be a meeting of minds as to acceptance of the data governance.

In the context of data relevant to a smart contract, it is fair to assume that this will be structured rather than unstructured data (noting, of course, that this is not a binary question, but rather data will sit along a spectrum of degrees of structure, defined by the purpose of a structure and intended use of the data). In the same way that traditional contract definitions are key to their reflection of the intentions of parties and envisaged outcomes, smart contracts, due to their automated performance features, are hugely reliant on the way in which data inputs flow through their conditional logic – requiring the drafters of smart contracts to carefully consider data governance parameters that might mean the logic is no longer appropriate, or in more sophisticated contracts, to provide for alternative logic based on data quality features of the data inputs at “run-time”.

To the extent that “big data” is utilised as data in the smart contract context, there is of course likely to be a methodology developed to use such a data set in order to address any inherent “messiness” in the data. The extent of any techniques used to overcome such “messiness”, needs to be assessed in the context of their use within a smart contract’s conditional logic, and the logic may need to differ based on various aspects of the governance of such data (for example, the appropriateness of certain “less-conforming” data structures as inputs).

Enterprise data management theory typically defines the following roles:

— the data trustee;

— the data steward; and

— the data custodian.

The data trustee is ultimately responsible and is the overarching “guardian” of a particular data domain, defining the scope of the data domain, tracking its status, and defining and sponsoring the strategic roadmap for the domain. They would ultimately be accountable for the data, but would typically delegate the day-to-day data governance responsibilities to data stewards and data custodians.

The data steward is a subject-matter expert who defines the data category types, allowable values and data quality requirements. Data stewardship is concerned with taking care of data assets that do not necessarily belong to the steward(s) themselves, but which represent the concerns of others.

Data custodians are also accountable for data assets, but this is from a technology perspective (rather than the business perspective in respect of the data steward), managing access rights to the data and implementing controls to ensure their integrity, security and privacy (covered in Section 10 of this guidance).

Of course, the difficulty is that a smart contract is likely, in most cases, to operate outside of a single enterprise. Accordingly, provision must be made within the terms of the smart contract itself to ensure the data quality sought, perhaps through data governance requirements or data quality checks agreed between the smart contractual parties.

Dimensions of data quality

The dimensions of data quality that might be relevant to the data variables in a smart contract will of course vary based on the nature of the smart contract in question, and the specific business use of the specific data variable. These will typically be:
Accuracy: the degree to which data correctly represents the entity it is intended to model (for example, where a default rate of a large loan portfolio is a data input, the extent to which loans which are in a potential event of default state, rather than actual event of default, are excluded from the measurement).

- **Completeness:** whether certain attributes always have an assigned value in a data set (for example, how loans without default data are treated)

- **Consistency:** ensuring data values in one data set are consistent with values in another data set (for example, where the test of whether a loan in default differs across the data set).

- **Currency:** the degree to which information is current with the world it seeks to model and represent (for example, the degree to which assumptions have been used to arrive at the data point in question).

- **Precision:** the level of detail of data elements (both in terms of, for example, the number of decimal points to which a numeric amount is detailed, to the number of data elements within a particular data attribute in the data structure that may impact the data value – often based on its intended usage).

- **Privacy:** the need for access control and usage monitoring.

- **Reasonableness:** assessment of data quality expectations (such as consistency) relevant within operational contexts.

- **Referential Integrity:** expectations of validity in respect of references from the data in one column to another in a data set.

- **Timeliness:** the time expectation for the accessibility and availability of information (for example, the precise cut-off time in respect of which loan information will be included, and whether the data source is able to guarantee timeliness of inclusion of data by the time the data is utilized within the smart contract logic).

- **Uniqueness:** the extent to which records can exist more than once within a data set.

- **Validity:** consistency with the domain of values and with other similar attribute values.

Data required to assess the data quality of a data variable and quality control policies

There are four main methodologies to be considered in assessing the data quality of a data variable within a smart contract:

1. **A data quality assessment that does not require additional data.** In this case, the data quality can be assessed by considering and analysing the value of the data variable itself. For example, “a speed of a car is within acceptable bounds if it is between 0 and 60 miles per hour”.

2. **A data quality assessment that relies on historical values of the data.** For example, the temperature of an individual taken by an IoT device is only of sufficient quality if it doesn’t differ from any prior recording in the previous five minutes by more than two degrees Celsius.

3. **A data quality assessment that relies on a (single) value or feature of (possibly multiple) other variables.** For example, a property address assessed against a land register.

4. **A data quality assessment that relies on multiple other values or features of (possibly multiple) other variables.** For example, a temperature reading might be compared against prior readings of different subjects.

There are broadly five policies that can be adopted in respect of the data, allowing the verification of data quality at runtime:

1. **Accept Value:** within tolerances, even though the data quality may not be ideal, it may be accepted.

2. **Do Not Accept Value:** a breach of the agreed tolerance results in the non-acceptance of the data input. The consequence of this must be considered and agreed in the context of the contractual agreement between the parties.

3. **Log Violation:** it may be necessary to accept certain data inputs, despite some concerns regarding data quality, whilst flagging it as being of low data quality for informational purposes.

4. **Raise Event:** where a low data quality input represents a critical situation that requires an immediate action (be it by a person or system), the automated action might be to escalate and raise an event.

5. **Defer Decision:** a particular violation of a data quality threshold on an input might not be enough, in itself, to result in a definitive automated action, and the decision may simply be deferred.