The LawTech Delivery Panel

LawTech UK

UK Jurisdiction Taskforce of the LawTech Delivery Panel

Public consultation

The status of cryptoassets, distributed ledger technology and smart contracts under English private law

MAY 2019

Foreword

by Sir Geoffrey Vos, Chancellor of the High Court

Smart contracts will only finally take off when market participants and investors have confidence in them.

Mainstream investors still need to be convinced that their legal rights can be protected when they trade in cryptoassets and enter into smart contracts.

The UK Jurisdiction Taskforce (the "**UKJT**") of the LawTech Delivery Panel believes that perceived legal uncertainty is the reason for some lack of confidence. For that reason, the UKJT has launched this public consultation to identify the key legal questions that need to be answered. The answers to those legal questions will provide a dependable foundation for the mainstream utilisation of cryptoassets and smart legal contracts.

Parts of the technology sector have suggested that contracts written in code have no need for legal superstructure. The UKJT believes that investor confidence would grow significantly if it were clear that cryptoassets were property in English law, that they could be the subject of a security interest and that smart legal contracts give rise to binding legal obligations.

Once the consultation has closed and the results have been analysed, the UKJT will publish a legal statement in late summer providing the best possible answers to the critical legal questions under English law. It will then be possible to see whether any legislative change is necessary or appropriate.

We hope that many people in the technological and legal sectors will find time to participate in this consultation.

****

The UKJT brings together the Judiciary, the Law Commission of England and Wales and technology and legal professionals. The Financial Conduct Authority assisted as a technical advisor. The UKJT's members are:

Sir Geoffrey Vos (Chancellor of the High Court and Chair of the UKJT)
Lawrence Akka QC (20 Essex Street)
Richard Hay (Linklaters LLP)
Peter Hunn (Accord Project)
Mary Kyle (City of London Corporation)
Sir Antony Zacaroli (Justice of the High Court)

Observer:

Sir Nicholas Green (Chair of the Law Commission of England and Wales)

Technical advisor to UKJT:

Christopher Woolard (Financial Conduct Authority)

Contents

Consultation on the status of cryptoassets, distributed ledger technology and smart contracts under English private law[1]

1 Background to this consultation

The development of distributed ledger technology ("**DLT**"), cryptoassets, smart contracts and associated technologies has far-reaching implications for financial markets, both domestically and internationally.

Nevertheless, the experience of market participants at present suggests that a lack of certainty regarding the legal status of cryptoassets, DLT and smart contracts could be hampering this development.[2]

This uncertainty does not solely arise in the context of English law and the jurisdiction of England and Wales.[3] However, creating a measure of confidence in these issues would increase confidence in the use of cryptoassets, DLT and smart contracts and bolster the use of English law and the jurisdiction of England and Wales in transactions concerning cryptoassets, as well as in smart contracts more generally.

English law, as a well-developed flexible common law system, has the ability to provide the certainty and predictability that the commercial community demands, and is well able to adapt to deal with fast-changing technologies. Consequently, English law and the jurisdiction of England and Wales are well-positioned to provide the legal foundation for the development of these technologies.

2 Scope of this consultation

The LawTech Delivery Panel ("**LTDP**") was established by the UK Government, the Judiciary and the Law Society of England and Wales and has as its overarching objective the promotion of the use of technology in the UK's legal sector.[4] The UKJT is one of six taskforces established by the LTDP for the purposes of achieving this objective.[5]

---

1 In this consultation paper, references to "English law" should be read as references to the law of England and Wales.

2 For example, respondents to the Financial Conduct Authority's April 2017 "Discussion Paper on distributed ledger technology" were particularly interested in the use of DLT in the capital markets sector, in particular, underpinning market trading infrastructure, including the use of smart contracts. However, respondents said that, before they would consider using those solutions at scale, they would have to be clearer on issues such as the legal status of cryptoassets and the enforceability of smart contracts. The Discussion Paper is accessible here: https://www.fca.org.uk/publication/discussion/dp17-03.pdf (Accessed May 2019). See also paragraph 1.16 of the FCA's Feedback Statement published in December 2017, accessible here: https://www.fca.org.uk/publication/feedback/fs17-04.pdf (Accessed May 2019).

3 For example, we note that the European Union Blockchain Observatory Forum (in a report entitled "Scalability, Interoperability and Sustainability of Blockchains", published 6 March 2019) sets out in a list of recommendations the need to resolve the tensions between "GDPR and blockchain, the legal fiscal and accounting status of crypto assets, and the legal status of smart contracts, among others". The report is accessible here: https://www.eublockchainforum.eu/sites/default/files/reports/report_scalaibility_06_03_2019.pdf (Accessed May 2019).

4 For further background on the LTDP, please see: https://www.lawsociety.org.uk/policy-campaigns/articles/lawtech-delivery-panel/ (Accessed May 2019).

5 LTDP taskforces have been established in the following areas: Ethics, Commercial Dispute Resolution, Investment, Education, Regulation, and UK Jurisdiction (i.e. the UKJT).

The objective of the UKJT is to demonstrate that English law and the jurisdiction of England and Wales together provide a state-of-the-art foundation for the development and use of DLT, smart contracts and associated technologies.

In pursuit of this objective, the UKJT is co-ordinating the preparation of an authoritative legal statement ("**Legal Statement**") on the status of cryptoassets and smart contracts under English private law. The intention is that the Legal Statement will either demonstrate that English private law already provides sufficiently certain foundations in relation to the relevant issues, or will highlight particular areas of uncertainty that may be ripe for further clarificatory steps to be taken.

The purpose of this consultation is to seek input from stakeholders as to the principal issues of perceived legal uncertainty regarding the status of cryptoassets and smart contracts under English private law to inform what should be addressed in the Legal Statement.

In Annex 1 (*Questions to be addressed in the Legal Statement*), we have set out what the UKJT considers to be the principal issues. However, ultimately, for the Legal Statement to serve its purpose, it must address those issues that market participants themselves are most concerned with. This is why we hope that key industry stakeholders will find time to participate in this consultation.

In Annex 2 (*Overview and key features of DLT*), we outline the key features of DLT. This informs the UKJT's understanding of the key legal issues arising in this context and, ultimately, those that will be addressed in the Legal Statement.

In Annexes 3 (*Cryptoassets*) and 4 (*Smart contracts*), we provide further detail on the technical aspects of cryptoassets and smart contracts, again, each with the intention of informing an understanding of the issues to be addressed in the Legal Statement.

3 Overview of the key issues of legal uncertainty included in this consultation

As this consultation and the Legal Statement are focused on private law, the questions identified in Annex 1 (*Questions to be addressed in the Legal Statement*) are accordingly limited in scope.

Quite intentionally, they do not cover certain other areas of law insofar as they relate to cryptoassets or smart contracts, including (among others) their regulatory characterisation and treatment, matters of taxation, criminal law, partnership law, data protection, consumer protection, settlement finality,[6] regulatory capital, anti-money laundering or counter-terrorist financing. We recognise that these are important areas, and ones in which market participants may feel there exists a degree of legal uncertainty in some instances. However, the UKJT feels that other bodies or organisations are better-placed to provide the necessary clarity on these issues, and so they do not form a part of this project.[7] The questions also do not address certain areas of perceived legal uncertainty where too many potential factual scenarios would need to be considered in order for any helpful answers to be provided.

---

[6] For example, under the Financial Markets and Insolvency (Settlement Finality) Regulations 1999, as amended.

[7] We note there are several projects underway which seek to address considerations in these areas. For example, the Basel Committee on Banking Supervision has published a "Statement on crypto-assets" (see: https://www.bis.org/publ/bcbs_nl21.htm (Accessed May 2019)), in which it has said that it will, in due course, clarify the prudential treatment of bank exposures to crypto-assets, and the Financial Conduct Authority has recently conducted a consultation on the regulatory characterisation and treatment of cryptoassets (see: https://www.fca.org.uk/publication/consultation/cp19-03.pdf (Accessed May 2019)).

We set out below some background on the questions which we have included in Annex 1 (*Questions to be addressed in the Legal Statement*).

3.1 Legal status of cryptoassets

Many aspects of the status of cryptoassets as a matter of English private law are considered by some to be unclear. In particular, notwithstanding that a significant amount of work has been undertaken in relation to a number of these issues by various academic, professional and public bodies, it is understood to be of general concern to the market that an authoritative response be given to the questions of whether, and, if so, the circumstances in which, a cryptoasset may be characterised under English law as property. The questions relevant to this are therefore set out in paragraphs 1.1 and 1.2 of Annex 1 (*Questions to be addressed in the Legal Statement*).

Property law matters both to users of a DLT system and to third parties dealing with those users. If a cryptoasset is not property, it cannot be owned. If it cannot be owned, it cannot be purchased, sold, otherwise transferred in law or rights to it asserted if it is stolen. Neither can a trust be declared, or security created, over it. The concept of a cryptoasset being recognised as property is therefore critical to the application of private law to transactions involving cryptoassets. If a cryptoasset is recognised as property, it is then necessary to understand the legal nature of that property. Traditionally, English law recognises physical things (*choses in possession*) and legal rights (*choses in action*) as property".[8] If a cryptoasset is recognised as property, does it fall into one of these categories or does it fall within some other category of property under English law? This is also critical to the application of private law to transactions involving cryptoassets because it is necessary to determine the location of property (its *situs*), under most legal systems, in order to determine the correct law governing transfers (alienation) of the property concerned.

Consequently, the response to the threshold question of whether a cryptoasset may be recognised under English law as property either dictates to a large degree or is materially relevant to the outcome of a series of ancillary questions, including whether certain types of security may validly be granted over it and its treatment for certain purposes as a matter of English insolvency law. These questions are set out in paragraphs 1.2.1 to 1.2.6 of Annex 1 (*Questions to be addressed in the Legal Statement*).

Equally, if a cryptoasset is capable of being recognised as property under English law, there is a series of additional questions as to other characterisations under English private law which may also be relevant. These questions include whether a cryptoasset may be characterised as a "documentary intangible" or as being "negotiable" (i.e. in the sense that a transferee may, by the mere transfer of a cryptoasset, acquire better title to that cryptoasset than that of its transferor), and whether cryptoassets may be recognised as "goods" for certain statutory purposes. These questions are set out in paragraphs 1.2.7 to 1.2.11 of Annex 1 (*Questions to be addressed in the Legal Statement*).

The UKJT also understands that there is uncertainty among market participants as to whether DLT records of cryptoassets are capable of amounting to a "register" for the purposes of evidencing, constituting and transferring title to certain types of securities under English law.

[8] We do note, however, that in certain circumstances the courts have been prepared to recognise that certain intangible things, such as carbon emission allowances, which also do not clearly fall into either category of personal property are, nevertheless, capable of being recognised as property under English law. See *Armstrong DLW GmbH v Winnington Networks Ltd* [2012] EWHC 10 (Ch), for example.

This question is set out in paragraph 1.2.12 of Annex 1 (*Questions to be addressed in the Legal Statement*).

In Annex 3 (*Cryptoassets*), we discuss the meaning of the term "cryptoasset", building on the general overview of DLT provided in Annex 2 (*Overview and key features of DLT*). In doing so, we explain certain key features of cryptoassets within some commonly used DLT models, with the aim of providing the authors of the Legal Statement with a description of certain key common features, given the multiplicity of potential models which exist.

3.2 Enforceability of smart contracts

As noted by the Law Commission of England and Wales, to ensure that the English courts and English law remain competitive choices for business, there is a compelling case for reviewing the current English legal framework to ensure that it facilitates the use of smart contracts.[9]

It is understood that market participants attempting to replicate contractual arrangements written in prose using smart contracts, are principally concerned that the circumstances in which smart contracts are capable of giving rise to binding legal obligations be clarified. This question is set out in paragraph 2.1 of Annex 1 (*Questions to be addressed in the Legal Statement*).

The UKJT is aware that some market participants may question the merits of this exercise, given that some among them may view one of the benefits of smart contracts as being that they are sometimes considered to remove the need for parties to rely on a legal framework to enforce their rights against each other. However, if smart contracts are capable of giving rise to binding legal obligations, it will be important for parties to be aware of the circumstances in which this will be the case (notably if the parties' intention is not to create legal relations). It will also be important for parties to know if and how their rights might be enforced in the event that technology does not work as expected.

Again, depending on the answer to that principal question, a series of ancillary questions arises. Notably, how the general principles of contractual interpretation would be applied by an English court in the context of a smart legal contract and the circumstances in which a statutory signature or "in writing" requirement may be met in the context of smart legal contracts. These questions are set out in paragraph 2.2 of Annex 1 (*Questions to be addressed in the Legal Statement*).

In Annex 4 (*Smart contracts*), we discuss how the market currently understands the term "smart contract", again building on the general overview of DLT provided in Annex 2 (*Overview and key features of DLT*). In doing so, we explain at a high level certain of the key features of smart contracts, and how they differ as between different implementations. As with Annex 3 (*Cryptoassets*), the aim of this section is to inform and help circumscribe the answers that will be provided in the Legal Statement.

3.3 Application of English law

Readers may note that, with the exception of the question posed in paragraph 1.2.3 of Annex 1 (*Questions to be addressed in the Legal Statement*), the question of the extent to which English law would be the applicable law in relation to dealings or other arrangements involving cryptoassets or smart contracts is not dealt with directly.

---

[9] The Law Commission of England and Wales, Thirteenth Programme of Law Reform (Dec. 2017), paragraphs 2.38 to 2.39. See: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/668113/13th-Programme-of-Law-Reform.pdf (Accessed May 2019).

The UKJT recognises that this is an area of much uncertainty for market participants. It is also of the view, however, that this is an issue which is highly fact-dependent, which limits the effectiveness of any attempt to provide a broad Legal Statement on the topic.

That said, the UKJT does consider that there would be value in setting out guidance within the Legal Statement as to the steps that may be taken by developers and participants to reduce uncertainty by ensuring, where desired, that English law will govern the relevant dealings or other arrangements, such as transactions within a DLT system or smart contracts to be deployed within such system.

4 Consultation questions

Your input is sought in relation to the following questions:

1. Do consultees agree that the questions in Annex 1 (*Questions to be addressed in the Legal Statement*) to this consultation paper cover the principal issues as to the legal status of cryptoassets and smart contracts under English private law?
2. If you disagree, what alternative questions or issues relating to the legal status of cryptoassets and smart contracts do you think need to be addressed?

5 Consultation process

This consultation will remain open for responses until 21 June 2019. Once this consultation has closed and the results have been considered, it is intended that the Legal Statement will be published in late summer of 2019. It will then be possible to see whether any further steps are necessary or appropriate.

Written responses to the consultation questions should be provided by:

(i) completing the form set out at the following link: https://www.lawsociety.org.uk/news/stories/cryptoassets-dlt-and-smart-contracts-ukjt-consultation; or

(ii) by email to UKJT@justice.gov.uk.

The UKJT will also be hosting a public event in order to receive feedback on the consultation questions in person. Further details on the public event, including details regarding registering for the event, can be found here: https://www.lawsociety.org.uk/news/stories/cryptoassets-dlt-and-smart-contracts-ukjt-consultation.

Annex 1
Questions to be addressed in the Legal Statement

Each of the following questions is posed as a matter of English law.

1 Legal status of cryptoassets

1.1 Principal question

Under what circumstances, if any, would the following be characterised as personal property:

1.1.1 ***a cryptoasset; and***

1.1.2 ***a private key[10]?***

1.2 Ancillary questions

General law

1.2.1 If a cryptoasset is capable of being property:

(i) is that as a *chose in possession*, a *chose in action* or another form of personal property?

(ii) how is title to that property capable of being transferred?

1.2.2 Is a cryptoasset capable of being the object of a bailment?

1.2.3 What factors would be relevant in determining whether English law governs the proprietary aspects of dealings in cryptoassets?

Security

1.2.4 Can security validly be granted over a cryptoasset and, if so, how?

1.2.5 If so, what forms of security may validly be granted over a cryptoasset?

Insolvency

1.2.6 Can a cryptoasset be characterised as "property" for the purposes of the Insolvency Act 1986?

Transferability and negotiability

Under what circumstances, if any, would a cryptoasset be characterised as:

1.2.7 a documentary intangible;

1.2.8 a document of title;

1.2.9 negotiable[11]; or

1.2.10 an "instrument" under the Bills of Exchange Act 1882?

[10] To the extent it is considered as distinct from the relevant "cryptoasset".

[11] In the sense that a transferee may, by its mere transfer, acquire better title than that of its transferor.

Goods

1.2.11 Can cryptoassets be characterised as "goods" under the Sale of Goods Act 1979?

Register

1.2.12 In what circumstances is a distributed ledger capable of amounting to a register for the purposes of evidencing, constituting and transferring title to assets?

2 Enforceability of smart contracts

2.1 Principal question

In what circumstances is a smart contract capable of giving rise to binding legal obligations, enforceable in accordance with its terms (a "smart legal contract")?

2.2 Ancillary questions

2.2.1 How would an English court apply general principles of contractual interpretation to a smart contract written wholly or in part in computer code?

2.2.2 Under what circumstances would an English court look beyond the mere outcome of the running of any computer code that is or is part of a smart contract in determining the agreement between the parties?

2.2.3 Is a smart contract between anonymous or pseudo-anonymous parties capable of giving rise to binding legal obligations?

2.2.4 Could a statutory signature requirement[12] be met by using a private key?

2.2.5 Could a statutory "in writing" requirement be met in the case of a smart contract composed partly or wholly of computer code?

[12] For example, in the context of a disposition of an equitable interest (under s53(1)(c) Law of Property Act 1925 (LPA)) or of a legal assignment (under s136(1) LPA)?

Annex 2
Overview and key features of DLT

1 Introduction

DLT is the general term which refers to technologies designed to maintain digital records which are synchronised between participants in a computer network in such a way that identical digital records are (or are able to be) held locally by each participant (or a subset of the network participants), with specific rules relating to the circumstances in which those digital records can be created, updated and then synchronised between participants.

Two core design aims for most DLT implementations are to create mechanisms whereby the relevant digital records: (i) cannot be duplicated in such a way as to permit them to be re-used (i.e. "double-spent"); and (ii) are capable of exclusive control. Most DLT implementations purport to achieve these core aims through the combination of the rules dictating how digital records are created and updated, and the synchronisation of those digital records between participants.

With the aim of informing the answers to the Legal Statement, in paragraph 2 of this Annex below we offer a description of different aspects of DLT, and in paragraph 3 of this Annex below we seek to draw out certain of the key features of DLT which will be central to any legal analysis.

At the outset of this exercise, we acknowledge its limitations. Quite apart from the challenges presented by terminology and taxonomy, these technologies are constantly evolving and what may be a key feature of many DLT implementations at present may well not be in the near future. It should also be noted that, with technological advances, so too the key features of DLT systems that are of relevance to the legal analysis today, or the manner in which they affect the conclusions of that analysis, may change.[13] Nevertheless, it is considered there is value in at least attempting to describe certain aspects of DLT, in order to inform the Legal Statement.

2 Description of DLT

2.1 Network of participants

DLT systems generally rely on computer networks, where participants connect to other (but not necessarily all other) peers. Other key aspects of DLT are forms of software which make use of this computer network.

Participants in a DLT network take many different forms and participate in the computer network to different degrees. Some participants conduct all functions contemplated by the relevant software, whereas others may perform a far more limited role.

"Nodes" are instances of the specific network software being run. That software allows the relevant computer to process and communicate information to other nodes and network participants. So-called "full" nodes usually each store an entire copy of the distributed ledger, whereas so-called "validator" nodes usually also validate proposals to update recorded data (as to this, see paragraph 3 of this Annex). Certain other participants may only able to send messages to other participants on the network, without storing a local copy of the data or being able to validate updates.

[13] For example, the impact of advances in quantum computing, and their effect on the permanence of cryptoassets, remains to be seen.

On certain DLT implementations, copies of the distributed database may be located all around the world, depending on the whereabouts of the nodes. Many DLT implementations are designed so that there is no limit on the number or location of participants who can act as nodes, meaning the geographical reach of some of these networks is potentially significant.

2.2 Permissionless vs permissioned; public vs private

The population of users that is able to: (i) act as a node on the network; and (ii) access the data recorded on the distributed ledger and contribute to updates of that data, is a key point of differentiation between different DLT models.

At a high level, a "permissionless" DLT implementation is one in which anyone can participate in the network without prior authorisation (i.e. anyone can operate a node on the network). In contrast, in a "permissioned" DLT implementation, prior authorisation is required in order to participate in the network. This prior authorisation may be provided in different ways, for example by all other participants in certain implementations, or, in others, from some form of central authority which has superior credentials to, and certain authority over, other nodes (often referred to as "master nodes").

Where a DLT implementation relies on a form of central authority, it may also be that such central authority has the ultimate say in how data is updated on the distributed ledger. This may be because it is the sole user with the ability to update the distributed ledger, or because it has a unique ability to change or override records which other users have previously validated.

The "permissionless"/"permissioned" distinction also differs from the "public"/"private" distinction. In a "public" DLT implementation, all users can view the records being added to the distributed ledger, whereas in a "private" DLT implementation, sight of the record is restricted by system design to a limited subset of users.[14]

A "public" DLT implementation is often also "permissionless" ("**Public and Permissionless**" models), and a "private" DLT implementation is often also "permissioned" ("**Private and Permissioned**" models).

2.3 The distributed digital records

As alluded to above in paragraph 1 of this Annex, distributed ledgers are digital records which are shared between a network of computers such that each participant (or relevant subset of participants) has (or can access) a complete record of the data.

2.3.1 Data structure

Different DLT implementations structure digital records on a distributed ledger in different ways. A common way in which the digital records are structured is in the form of a "blockchain". In a blockchain, digital records are structured in distinct data container structures known as "blocks", i.e. electronic parcels of data. These blocks form a sequential chain, with each block usually linked to the previous block.[15] In many forms of blockchain, this linking between blocks is achieved cryptographically, as each block will record the "hash" of the digital records within the previous block (or whichever set of

[14] See the Financial Conduct Authority's April 2017 "Discussion Paper on distributed ledger technology" p. 10. https://www.fca.org.uk/publication/discussion/dp17-03.pdf (Accessed May 2019).

[15] We note that in certain blockchains blocks are linked to several different blocks rather than just the block appearing immediately prior to it on the ledger.

blocks the new block is linked to).[16] [17] We explain the process of hashing in greater detail below in paragraph 3 of this Annex.

2.3.2 Transaction ledger vs account ledger

Regardless of how the digital records are structured on a distributed ledger, for most current DLT implementations the digital records which appear on the ledger are records relating either to "transactions" or to "accounts". A DLT implementation with a "transaction" ledger is often structured as an "unspent transaction output" ("**UTXO**") model, whereas a DLT implementation with an "account" ledger is known as conforming to the "account-based" ("**Account-Based**") model. We provide some additional detail below regarding what is recorded on the distributed ledger of a DLT implementation conforming to each of these models.

(i) **UTXO model**

In the UTXO model, what generally appears as the digital record on the distributed ledger is a "transaction" (or group of transactions) that has taken place on the network.[18] In this context, the term "transaction" refers to a digital record which:

(a) indicates the possibility for a specific other party (this party can be conveniently imagined as a "transferee") to create a new digital record upon the satisfaction of certain conditions (usually including a "cryptographic signature" – discussed further below in paragraph 3.2 of this Annex). Broadly speaking, this is referred to as the transaction "output" (or UTXO); and

(b) references the output of a prior transaction and provides the required cryptographic signature in order to create a new digital record. Broadly speaking, this is referred to as the transaction "input".[19]

It should be noted that the total set of outputs (i.e. digital records which indicate the possibility for a given participant to create further digital records upon the satisfaction of certain conditions) is not something which is recorded on the distributed ledger in the UTXO model. For example, in the Bitcoin system (which is an example of a UTXO model), the total balance of "Bitcoin" available for a participant to "spend" is not recorded on the Bitcoin distributed ledger.[20] Rather, the distributed ledger records the transactions that have taken place between users and gives every transaction a unique identifier.

(ii) **Account-Based model**

---

[16] One of the data points recorded in the previous block included in this hash will be the hash of the block recorded prior to the previous block; hence the reference to the cryptographic linking of blocks.

[17] Note that the first block (i.e. the genesis block) is not linked to any previous block.

[18] The distributed ledger in a UTXO model will record other data; however, for the purposes of this analysis, the key record is the group of transactions.

[19] Andreas M. Antonopoulos, *Mastering Bitcoin* (2nd Ed.) (2017), p.xxxi.

[20] With a Public and Permissionless implementation of the UTXO model, it is, however, generally possible for anyone to discover the balance of UTXO available at an address of a public key. With Bitcoin, for example, this can be viewed by searching a Bitcoin user interface such as www.homebitcoin.com/easybalance/ (Accessed May 2019).

Similar to the UTXO model, distributed ledgers in Account-Based models also track transactions between users of the network. In the context of an Account-Based system, the term "transaction" has a similar meaning to the meaning it has in the context of a UTXO model, albeit that the data comprising the transaction differs in that it does not reference any prior outputs or transactions. We discuss this in greater detail below in paragraph 5 of Annex 3 (*Cryptoassets*).

The key difference between the two models, however, is that the distributed ledger in an Account-Based model also maintains digital records known as "accounts".[21] An "account" is a digital record comprising, among other things, an "address", an account "nonce" and a "balance".[22] Breaking these components down:

(a) "**Address**". The "address" is a unique sequence of characters which can be used to identify an account in a DLT network. The address is the result of hashing a user's "public key" to compress and shorten the public key.[23] We discuss the concept of "hashing" and "public-private key cryptography" in paragraph 3 of this Annex below.

Note that participants in a UTXO model will also have an address. However, in the UTXO model, a participant's address is recorded on the distributed ledger only as part of a transaction.

(b) **Account** "**nonce**". A "nonce" is a computer science term for a random, unique number which can only be used once. Within an Account-Based model, this number will change following the construction and broadcasting of every transaction from an account.[24] In Ethereum, for example, every account has a publicly viewable nonce, which is increased in increments of one every time that a transaction is recorded on the distributed ledger. Other Account-Based models do not use an "incrementing" nonce, but instead the system generates an entirely random number for each new transaction.

In Account-Based models, the account nonce plays a crucial role in enabling network participants to verify that a transaction does not propose to update the same digital record twice. We discuss this further in paragraph 5.2 of Annex 3 (*Cryptoassets*) below.

(c) "**Balance**". The "balance" is the total set of digital records which indicate the possibility for the account user to create further digital records upon the satisfaction of certain conditions. In Ethereum, for example, the

[21] The Ethereum Homestead documentation summarises the difference between the Ethereum and Bitcoin blockchains as follows: "Whereas the Bitcoin blockchain was purely a list of transactions, Ethereum's basic unit is the account". See: http://www.ethdocs.org/en/latest/ (Accessed May 2019).

[22] We note that Account-Based systems often support "contract accounts", which facilitate the creation and deployment of smart contracts. These accounts will include digital records of "smart contract code", which we discuss in greater detail in Annex 4 (*Smart contracts*). This smart contract code is also something that appears on the distributed ledger in an Account-Based model.

[23] Generally, an address is a hashed version of the public key. See: https://hackernoon.com/a-closer-look-at-ethereum-signatures-5784c14abecc (Accessed May 2019).

[24] Note that the account nonce is a separate concept to the "nonce" used in any proof-of-work algorithm.

balance represents the amount of value (or "Ether") available for a participant to spend. This is another point of contrast with the UTXO model.

When a transaction is verified and synchronised between participants, the distributed ledger will also record a change in the accounts of the participants involved in the transaction.[25] This is discussed in greater detail in paragraph 5 of Annex 3 (*Cryptoassets*).

2.4 Changing the distributed record

Distributed ledgers are intended to be dynamic rather than static. This requires rules governing the creation and modification of the digital records which appear on the ledger.

2.4.1 Creating new digital records

The circumstances in which entirely new digital records can be added to the distributed ledger are a key design feature of almost every DLT implementation. These circumstances are specified within the system protocol that is run by network participants (or relevant subset of network participants) and may vary greatly between different implementations depending on what the system is designed for.

Some systems are set up so that new digital records are created only upon the satisfaction of certain conditions, which any participant may be able to satisfy. In several well-known examples of DLT implementations which structure data in the form of a blockchain, the system is designed such that new digital records are introduced in the first transaction in a block.[26] [27] Depending on the relevant consensus mechanism, this is part of the "reward" which the system awards to the participant (usually referred to as a "miner") who establishes the next valid block in the blockchain. We discuss this process further below in paragraph 3 of this Annex.

Equally, a system may be set up to introduce new digital records upon a payment of some form being made (i.e. a so-called "Initial Coin Offering"). Many are set up so that there is a limit to the total number of new digital records which can be created. Other systems are set up so that new digital records are naturally and consistently created as a function of time; some others at random, depending on the caprices of the system designers.

Some systems are designed so that new digital records are only created when a participant operating a master node authorises their creation. In certain Private and Permissioned DLT implementations, for example, a participant operating a master node may be able to create new distributed records when necessary in order to represent off-ledger assets. As an example, a central securities depository[28] may operate a Private and Permissioned network with the intention of facilitating the clearance of securities. In

[25] Andreas M. Antonopoulos and Gavin Wood, *Mastering Ethereum* (2018), p.110.

[26] See, for example, the Bitcoin Whitepaper: "the first transaction in a block is a special transaction that starts a new coin owned by the creator of the block." See: https://bitcoin.org/bitcoin.pdf (Accessed May 2019), p.4.

[27] In certain UTXO models, this first transaction in a block is called a "coinbase transaction".

[28] A central securities depository is an institution that holds financial instruments, including equities, bonds, money market instruments and mutual funds. It allows ownership of those instruments to be transferred in electronic form through updating electronic records which are often known as "book-entry records". See: https://www.fca.org.uk/markets/central-securities-depositories (Accessed May 2019).

such a network, the central securities depository would need to be able to create distributed records that represent the securities for which it is the registrar. In such instances, the central securities depository would typically operate the master node(s) and prohibit the operators of the other nodes from creating any additional distributed records (or, indeed, from validating any updates to those records).

2.4.2 Updating existing digital records

All DLT implementations have a set of rules for the updating of digital records recorded on the distributed ledger.[29] Broadly speaking, in most DLT implementations the process for updating the distributed ledger comprises five key stages:

(i) the construction of a proposal to update the distributed ledger with a new digital record (i.e. a "transaction message"). In many cases, the proposal will be to update the distributed ledger with a digital record which indicates the possibility for another party (i.e. the "transferee") to create a new digital record (but not introduce new value) if that party can satisfy certain conditions;

(ii) the propagation of that proposal through the relevant network, or to the relevant subset of users within the network (i.e. "broadcasting" the transaction message);

(iii) verification checks by other participants (who have received the proposal) to ensure that the proposal complies with the system's rules;

(iv) the further onward propagation (by the relevant network participants who have validated the proposal) of the proposal to other participants; and

(v) the recording of the updated data on the distributed ledger. This process is driven by a technique used to ensure that the relevant participants agree that the data should be recorded on the distributed ledger, known as a "consensus mechanism". We discuss this in greater detail below in paragraph 3.3 of this Annex.

2.4.3 Removing, reversing or deleting digital records

The ability for participants (or a subset of participants) to override digital records or remove them from the distributed ledger is a feature which may or may not be contemplated in a given DLT implementation. These are features which may assist in certain circumstances, for example, where cryptoassets are stolen or transferred in error. Where the system rules do not contemplate these features, it may be that the only way to remedy an error is to submit a "correcting" transaction (e.g. compel the transferee to re-transfer the digital record to the transferor).

Certain DLT implementations may be more likely to include these features. In certain Private and Permissioned networks, for example, it may be crucial for a master node to have the right to override transactions which fall foul of applicable law.

[29] Technically, when an existing digital record is "updated", a new digital record is created with that updated information. The distinction between this and the "creation" of entirely new digital records (discussed above in paragraph 2.4.1 of this Annex) comes down to whether new "value" is introduced into the system. In this context, "value" is a matter of system design rather than any objective wealth-based concept. New digital records which do not, as a matter of system design, introduce new value into the system can loosely be understood as "updates" to existing digital records, whereas new digital records which do introduce value can loosely be understood as "creations" of entirely new digital records.

3 Outline of key features of DLT

3.1 Authenticity, exclusivity and double-spend

With any "transfer" of valuable data, recipients will need to know that the data is authentic and not counterfeit; that the same data has not been transferred more than once (i.e. "double-spent"); and that such data is exclusively usable or "spendable" by the recipient.

Historically, these challenges have been dealt with in two ways: (i) by using physical certificates with unique identifiers; and (ii) by employing independent third parties to keep a record of the transactions.[30]

Electronic data which does not appear in a physical format is more or less (depending on the way in which it is held) susceptible of being copied. Without an infrastructure designed to protect against this, this is likely to destroy the possibility of a given electronic record having unique qualities, i.e. the scarcity of that digital record. It is also likely to mean that an individual cannot credibly claim to have exclusive control of that electronic record.

As discussed above, DLT systems are designed to offer solutions to these issues. Different DLT implementations attempt to solve these problems in different ways. However, at a high level, most use a combination of the following:

(i) a network validation or verification process, in which participants (or a subset of participants) check that a proposed change to the ledger complies with the system rules. This will usually involve checking:

  (a) that a proposal to update a given digital record comes from an individual who is, as a matter of system design, entitled to propose that update (i.e. to ensure "exclusivity"); and

  (b) that a proposal to update the same digital record has not already been made or recorded on the distributed ledger (i.e. to ensure no "double-spend"); and

(ii) the agreement among participants (or a subset of participants) that certain proposed changes (which have been verified) are then maintained on the distributed ledger.

We discuss each of these features below.

3.2 Validation process

3.2.1 Exclusivity

Different DLT implementations deal with exclusivity in different ways. For example, in the context of financial markets, a central securities depository may run a Private and Permissioned network for securities clearance and, while its clearing members may operate nodes on that network, the central securities depository's master node may want to preserve additional rights for itself (or its regulators) to validate transactions. In such systems, the central securities depository may be able to ensure exclusivity by acting as a trusted third party.

[30] Sarah Green, "Cryptocurrencies: The Underlying Technology", *Cryptocurrencies in Public and Private Law*, edited by David Fox and Sarah Green (2019), p.2.

However, many DLT implementations attempt to achieve exclusivity by utilising public-private key cryptography. At a high level, public-private key (or "asymmetric"[31]) cryptography involves a party (who we can imagine as the "transferor") encrypting that data in such a way that only another party (who we can imagine as the "transferee") who has access to something or information which can decrypt the data is able to decrypt it.

This process relies on "hashing" certain data inputs. At a high level, "hashing" is the function that transforms input data into a unique, fixed-length data string (known as the "hash") which, practically speaking,[32] is impossible to reverse to produce the inputs.[33] An illustration of how this works is provided below by reference to the well-known "Secure Hash Algorithm 256" ("**SHA256**") hash algorithm which, when applied to the sequence of letters "abc", has the following SHA256 hash (expressed in hexadecimal[34]):

ba7816bf-8f01cfea-414140de-5dae2223-b00361a3-96177a9c-b410ff61-f20015ad.[35]

This hash can be used as a unique identifier of the above text as: (i) if any punctuation or word were changed, a different hash would result; and (ii) in relation to the SHA256 hash algorithm, it is practically impossible to find another text that results in exactly the same hash.[36]

The transaction message created by a transferor will include a locking script which can only be satisfied by the transferee hashing their "private" key with certain other data to form a signature.

A private key is a string of data that is generated based on a random number (known as a "seed"). It is part of a participant's "public-private key" pairing, the other part being the participant's unique "public" key, with the "public" key being the hash of the private key. For a given DLT implementation, certain software enables participants to generate these key pairs.[37]

The locking script works by identifying within it the transferee's public key. The public key is usually freely viewable for any participant in the relevant network. However, due to the hashing process, practically speaking, it is not possible to derive the private key from the public key.

---

[31] It is referred to as "asymmetric" as the key that is used to encrypt the relevant data is different from the key used to decrypt it. See Imran Bashir, *Mastering Blockchain: Distributed Ledger Technology, Decentralization, and Smart Contracts Explained, 2nd Edition* (2018), p.81.

[32] We understand it is theoretically possible to discover the inputs to which the hashing algorithm has been applied. Were it to become easier to do this, there is a question as to whether any legal analysis would need to change to reflect this.

[33] See the EU Blockchain Observatory report, p.28. The report is accessible here: https://www.eublockchainforum.eu/sites/default/files/reports/report_scalaibility_06_03_2019.pdf (Accessed May 2019).

[34] Hexadecimal (or hex) is a system which uses 16 digits (being 0 1 2 3 4 5 6 7 8 9 A B C D E F) in order to represent binary sequences. Each hex digit reflects a 4-bit binary sequence.

[35] See: https://www.movable-type.co.uk/scripts/sha256.html (Accessed May 2019).

[36] Sarah Green, "Cryptocurrencies: The Underlying Technology", *Cryptocurrencies in Public and Private Law*, edited by David Fox and Sarah Green (2019), p.3.

[37] In the context of many DLT implementations, a user typically gains access to a public/private key pairing through a "wallet". A "wallet" refers to the software that allows users to generate master public-private key pairs from a seed number. This software can be used offline (i.e. without a connection to the associated DLT network). The term "wallet" is also, however, used to refer to an online application which displays all coin addresses (or public keys) of a user, and to provide this information accurately, the wallet needs to be online or connected to a blockchain file, which it uses as its source of information.

It is, however, easy to verify that a signature can only have been provided by someone with knowledge of the private key linked to the public key. Consequently, in DLT implementations which use public-private key cryptography, validating participants will need to check that a proposal from a participant to update the distributed ledger used the appropriate private key.

3.2.2 Double-spend

In addition to checking that a proposal to change the distributed ledger is made by an individual entitled to do so, validators must also check that this same proposal has not already been made.

The checks which are carried out for this purpose are very specific to the DLT implementation. We discuss these further below in the context of the UTXO model and the Account-Based model in paragraphs 4.3 and 5.2 of Annex 3 (*Cryptoassets*).

3.3 Synchronising the distributed record

All DLT implementations use a technique to ensure there is agreement between participants (or a relevant subset of participants) as to what digital records are maintained as part of the distributed ledger. This technique is often referred to as a "consensus mechanism". It is the consensus mechanism which determines whether a validated proposal to update a digital record (i.e. one which participants (or a relevant subset of participants) have verified as complying with the system's rules) forms part of the distributed ledger.

Many of the Public and Permissionless implementations rely on majorities of participants to agree on what is recorded on the distributed ledger. In such circumstances, if any participants alter or modify the content of any given record once it is recorded, it would also have to compete to persuade all nodes that the new content is correct.[38] It may be possible to do this, for example, if an individual has sufficient control over a majority (i.e. greater than 50%) of the nodes in the network – this is what is referred to as a 51% attack.[39] However, the larger the network the more difficult (and expensive) this becomes. Of course, this may be easier to achieve depending on the majority threshold required for the validation and distribution of new records.

There are a range of different consensus mechanisms. Two well-known examples are:[40]

(i) "**Proof of work**". This relies on a user proving that adequate computational resource has been spent before a record of transactions can be accepted as part of the distributed ledger. This function is performed by "miners", who are network participants who compete to establish new distributed records.[41]

In the context of many blockchain implementations which rely on proof of work as the consensus mechanism, once a transaction is verified as conforming to the system rules, it is included in a proposed block by a node for "mining" (although at this stage it remains

[38] Sarah Green, "Cryptocurrencies: The Underlying Technology", *Cryptocurrencies in Public and Private Law*, edited by David Fox and Sarah Green (2019), p.4.

[39] Where a DLT implementation relies on a trusted central authority or master node to assist with securing the integrity of data that is recorded, it may have less need to rely on this form of cryptographic linking between blocks, as the central authority may itself maintain the integrity of the ledger by acting as the sole party entitled to update it.

[40] Note that the mechanism by which consensus is reached varies depending on the particular implementation and is unspecific to whether it follows the UTXO model or the Account-Based model.

[41] In most DLT systems, all miners will also be nodes, but not all nodes will be miners.

an “unconfirmed transaction”). “Mining” is the process by which blocks are added to the blockchain. The block is validated by a miner performing a “proof of work” computational puzzle. The puzzle is intended to require many computational steps without shortcuts.[42] Once the puzzle is solved, the successful miner can broadcast the block with the puzzle solution in it and it is very easily verified by other nodes (who may or may not be miners) on the network. Once verified by the majority of nodes, the block is added to the blockchain.

(ii) “**Proof of stake**”. This relies on a user proving that it has a stake in the system that is sufficient to prevent it from acting in a malicious/fraudulent manner. Users competing to establish new blocks must construct a particular type of transaction which locks up their balance as a form of “deposit”. Validators then take turns proposing and voting on the next valid block, and the weight of each vote depends on the size of the validator’s deposit, with each validator shouldering the risk that they lose their deposit if the block they vote on is rejected by the majority of validators.[43]

In the context of many Private and Permissioned networks, as restrictions are placed on the population of entities who may operate nodes, a consensus mechanism dependent upon a majority of nodes may be less important. Further, where such networks include one or more master node(s), the master node(s) may act as the final and absolute truth for the network rather than relying on other nodes to establish consensus on what is recorded on the ledger.

3.4 Summary

Most DLT implementations attempt to ensure that certain digital records cannot be copied and can only be modified by someone entitled to modify them. In many implementations, public-private key cryptography enables an individual to provide evidence of an entitlement to modify the relevant data, and it also excludes anyone other than the participant who has knowledge of the relevant private key from being able to modify further the relevant data. The consensus mechanism and distributed nature of the database can prevent double-spend, as any attempt to spend the same data twice would not achieve the consensus necessary for validation and recording.

The Bitcoin Whitepaper sums up the combined effect of the different technologies as follows:

> *We have proposed a system for electronic transactions without relying on trust. We started with the usual framework of coins made from digital signatures, which provides strong control of ownership, but is incomplete without a way to prevent double-spending. To solve this, we proposed a peer-to-peer network using proof-of-work to record a public history of transactions that quickly becomes computationally impractical for an attacker to change if honest nodes control a majority of CPU power.*[44]

[42] Anthony Lewis (1 September 2015), “A Gentle Introduction to Bitcoin”, See: https://bitsonblocks.net/2015/09/01/a-gentle-introduction-to-bitcoin/ (Accessed May 2019).

[43] Andreas M. Antonopoulos and Gavin Wood, *Mastering Ethereum* (2018), p.321.

[44] See: https://bitcoin.org/bitcoin.pdf (Accessed May 2019), p.8.

Annex 3
Cryptoassets

1 Introduction

Cryptoassets are no longer merely a technological concept. Increasingly, they are entering the consciousnesses of governments, regulators, bankers, accountants, lawyers and academics.

Many regulatory authorities have assessed, from a regulatory perspective, various types of cryptoassets (and services related to them) for their respective jurisdictions, which has required such authorities to attempt to define what they are.[45]

However, for the purposes of English private law, there is no general definition of a cryptoasset, and the UKJT recognises that there may exist a great deal of uncertainty as to what is meant, or what is being described, when the term is used.

We do not seek to address the semantic uncertainties of the term "cryptoasset" in this consultation paper by advancing a legal definition of the term. However, for the purposes of informing the answers to be provided in the Legal Statement, we set out below high-level descriptions of certain of the key technical features of technologies which are commonly understood to support cryptoassets.

2 Technical features

Broadly speaking, the term "cryptoasset" is often used to describe something which is, or of which at least a component is, represented by certain data (often, although not necessarily, recorded on a distributed ledger) which, by virtue of the design of a broader system, can only be updated upon the satisfaction of specific conditions. As discussed in paragraph 3 of Annex 2 (*Overview and key features of DLT*), these conditions usually involve: (i) public-private key cryptography to evidence the authenticity of the participant proposing the update; and (ii) a mechanism to ensure the same data has not been copied or updated (i.e. "spent") twice.

As noted in paragraph 2.3 of Annex 2 (*Overview and key features of DLT*), there are two broad categories of DLT implementation: the UTXO model and the Account-Based model. Broadly, all DLT implementations will conform to one of these models, and both models can support either a Public and Permissionless network or a Private and Permissioned network. We provide further descriptions of the data that is stored, and how it is updated, in both models below in paragraphs 4 and 5 of this Annex.

Consequently, from a technical perspective, despite different DLT networks being designed for different purposes, the distributed records that appear in both which are commonly understood to comprise (whether in whole or in part) cryptoassets may share substantially similar technical features. We illustrate this below in paragraph 3 of this Annex.

---

[45] The Cryptoassets Taskforce (comprising HM Treasury, the Financial Conduct Authority and the Bank of England), for example, has advanced a broad definition of a cryptoasset as a type of "cryptographically secured digital representation of value or contractual rights that uses some type of DLT and can be transferred, stored or traded electronically". See paragraph 2.10 of the Cryptoassets Taskforce: final report (October 2018). See: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/752070/cryptoassets_taskforce_final_report_final_web.pdf (Accessed May 2019). Many regulators have gone on to draw a distinction between different types of cryptoassets, being "exchange tokens", "security tokens" and "utility tokens". See also the Financial Conduct Authority's consultation on the regulatory characterisation and treatment of cryptoassets (accessible here: https://www.fca.org.uk/publication/consultation/cp19-03.pdf (Accessed May 2019)).

3 Types of cryptoasset and other distributed records

The current most common purpose of DLT implementations is to facilitate an electronic means of exchange between different participants on the same network (a so-called "exchange token"). In such implementations, the data referred to as the "cryptoasset" does not provide any rights to anything outside of the "network". Bitcoin and Ether are examples of this type of cryptoasset.

However, the data infrastructure used within a system designed to facilitate "exchange tokens" can also be used as a means to facilitate the representation of legally enforceable rights of a participant. For example, systems may be set up where certain distributed digital records are intended to represent: (i) ownership of an off-ledger, real-world asset, such as a commodity; (ii) an entitlement to repayment of a specific sum of money; or (iii) an entitlement to a share in future profits of a company or project. The latter two examples are often referred to, in the context of financial services regulation, as "security tokens".[46] With such a use case, a separate mechanism is required in order to link the distributed digital record with the rights that distributed digital record purportedly represents. For example, where a DLT system is designed with the purpose of representing a bond in so-called "token" format, a separate legal document containing the covenant of the bond issuer to pay the "holder" of the cryptoasset would be required and would need to specify, for example, that the person or entity who has knowledge of the relevant private key corresponding to the DLT record is entitled to payment from the bond issuer, or otherwise define how the "holder" may be identified and assert its payment entitlement.[47]

In such use cases, what is meant by the term "cryptoasset" may be less clear. The term is often used to refer to the combination of the distributed digital record and the legally enforceable rights (created separately) which that distributed digital record is intended to represent.[48] However, the distributed digital records themselves (quite apart from the separately created enforceable legal rights) may share substantially similar technical features to the distributed digital records that appear in the context of an "exchange token".[49]

Consequently, any analysis of the legal status of "cryptoassets" should involve a consideration of these distributed digital records separately to the rights purportedly represented by them.

---

[46] The Financial Conduct Authority ("**FCA**") defines "security tokens" as those cryptoassets that meet the definition of a "specified investment" as set out in the Financial Services and Markets Act 2000 (Regulated Activities) Order 2001, and possibly also a "financial instrument" under Directive 2014/65/EU (MiFID II). For example, these cryptoassets have characteristics which mean they are the same as or akin to traditional instruments such as shares, debentures or units in a collective investment scheme. Security tokens are the type of cryptoasset which falls within the FCA's regulatory perimeter. For further details please see the FCA's Guidance on Cryptoassets, CP19/03, accessible at: https://www.fca.org.uk/publication/consultation/cp19-03.pdf (Accessed May 2019).

[47] As Private and Permissioned networks provide participants with greater levels of control over who participates, how cryptoassets come into existence and how they are transferred, Private and Permissioned DLT implementations are often seen as being more suitable for the development of a "security token" structure. However, in theory, it is certainly possible for a cryptoasset in a Public and Permissionless network to amount to a "security token", for example, if a separate legal mechanism provides that ownership of a cryptoasset in such network entitles the participant who knows the relevant private key to a payment, or otherwise defines how the "holder" may be identified and assert its entitlement.

[48] See, for example, the reference to "security tokens" in paragraph 2.11 of the Cryptoassets Taskforce: final report (October 2018). See: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/752070/cryptoassets_taskforce_final_report_final_web.pdf (Accessed May 2019).

[49] We note that several well-known DLT implementations are commonly thought of as not relying on "native" cryptoassets as part of the consensus mechanism, unlike Bitcoin or Ethereum. However, as these DLT implementations utilise either the UTXO model or the Account-Based model, the data recorded on the distributed shares similar features to that recorded on the distributed ledger in DLT implementations which do rely on "native cryptoassets". An example of this is R3's Corda, which utilises the UTXO model, but is not commonly thought of as relying on a "native" cryptoasset.

Given the similarities with respect to what appears on the distributed ledger, it may be that the legal analysis with respect to the distributed digital records utilised in a "security token" structure is the same as the legal analysis for the distributed digital records utilised in an "exchange token" structure.

The same point can be made with respect to DLT implementations where similar distributed digital records are intended to be used to:

(i) evidence (rather than constitute) a participant's title to an off-ledger, real-world asset. For example, some financial institutions may operate a DLT network (likely a Private and Permissioned network) for use as their books and records to record transactions and account positions; or

(ii) confer on participants the right to access a specific product or service that is provided on the DLT network or outside it (i.e. so-called "utility tokens"[50]).

4 Cryptoassets (or distributed records) in the UTXO model

In the case of certain cryptoassets, what is really being referred to is an "unspent transaction output" or "UTXO". Below, we draw on the explanations provided above in Annex 2 (*Overview and key features of DLT*) and elaborate on them in the context of UTXOs.

4.1 Representation as an "output" within a transaction

A UTXO is represented by a unique string of data, manifested as a readable sequence of characters.[51] It is viewable as a data entry, along with several other related data entries, within the digital record comprising a "transaction" appearing on the distributed ledger. It is usually identified in the "output" field within a transaction record.

Typically, the output field will contain the following sub-fields:

(i) **Value**. This is the numerical value which, as a result of the transaction, is being placed at the disposal of whoever knows the private key associated with the public recipient address. It is important to note that, in constructing a transaction message, the constructor could theoretically put whatever value they so desired; at the stage of transaction message construction, the value is not constrained by the values of UTXO already available for the message constructor to spend. The constraint on this is imposed by the other participants in the network who, in conducting the transaction validation check, would reject the proposed transaction as invalid on the grounds that the "outputs" do not have any previous associated "input" (see paragraph 4.3 of this Annex below). This rejection will necessarily happen as a result of the system protocol.

(ii) **Locking script**.[52] This is a script which stipulates what conditions must be fulfilled for that value to be placed at the disposal of another participant.

In a standard transaction where one party ("**P1**") aims to place value at the disposal of another participant ("**P2**"), this will be a requirement for P2 to provide: (i) P2's public key

[50] See paragraph 2.11 of the Cryptoassets Taskforce: final report (October 2018). See: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/752070/cryptoassets_taskforce_final_report_final_web.pdf (Accessed May 2019).

[51] David Fox, "Cryptocurrencies in the Common Law of Property", *Cryptocurrencies in Public and Private Law*, edited by David Fox and Sarah Green (2019), p.143.

[52] In the Bitcoin system, this is referred to as "*scriptPubKey*".

that, when hashed, matches the address embedded by P1 in the locking script (which will be P2's address); and (ii) a signature to prove that P2 knows the private key corresponding to P2's public key. Essentially, the locking script creates a cryptographic encumbrance on the specified value and can only be redeemed by the introduction of a solution to the locking script.[53] This solution will be provided as part of the transaction "input" which P2 uses when it wants to transfer the value associated with this output (see paragraph 4.2 of this Annex below).

It is important to note that each transaction recorded on a UTXO distributed ledger contains UTXO represented by a unique string of data. As this string of data is unique, technically speaking a "new" UTXO is created by each transaction, albeit that the transaction may not bring new "value" into the system.[54]

4.2 UTXO "consumed" as an input

In paragraph 4.1 of this Annex above, we outlined how UTXOs are represented as outputs within transaction data recorded on a distributed ledger. However, for a transaction to be valid in the UTXO model, it must also include "inputs".[55] Inputs are essentially "to-be-consumed outputs".

To illustrate this, it may be helpful to assume that a valid transfer of UTXOs has taken place between Alice and Bob, who are two participants in a UTXO system, such that those UTXOs are available for Bob (or whoever knows Bob's private key) to spend. If Bob wishes to "spend" those UTXOs by transferring them to another participant, Carol, he will need to construct a transaction message which contains inputs which reference those UTXOs. This will involve Bob constructing an input consisting of:

(i) **a transaction identifier**. This is a data entry which is used to refer to a particular output, which consists of:

  (a) **a transaction identification number ("TIN")**. This is used to uniquely identify the particular transaction that led to the creation of this input, i.e. the original transaction from Alice to Bob; and

  (b) **an output identification number ("OIN")**. This is an "index" which identifies the specific output which was transferred from Alice to Bob and which is to be consumed in the transaction being assembled by Bob (i.e. this identifies a particular output (or set of outputs) to be spent in Bob's transfer to Carol); and

(ii) **a signature script**. This is the computational script for satisfying the conditions stipulated by the locking script set by Alice when she made the original transfer to Bob, i.e. the conditions which enable Bob to consume the previous outputs as inputs for his transfer to Carol. As noted in paragraph 4.1(b) of this Annex above, in a standard transaction, to satisfy the conditions set by Alice in the locking script, Bob's signature script will be required to contain:

---

[53] Andreas M. Antonopoulos, *Mastering Bitcoin* (2nd Edn) (2017), p.24.

[54] We refer to paragraph 2.4 of Annex 2 (*Overview and key features of DLT*) and, in particular, footnote 29, where we draw this distinction.

[55] Transaction messages must also include certain other data. However, for the purposes of this consultation paper, we have not set out a detailed explanation of this additional data.

(a) Bob's public key which, when hashed, matches the address embedded in the locking script set by Alice; and

(b) a signature to prove Bob's knowledge of the private key corresponding to that public key. This will appear as a hash of Bob's private key and certain data from the transaction between Alice and Bob. Essentially, Bob uses his private key to encrypt certain transaction data, and that transaction data can be decrypted using Bob's public key. This allows anyone to check that Bob did indeed have knowledge of the private key, which is a key part of the transaction verification process.[56]

For Bob's transaction to Carol to be validated by nodes on the UTXO network, in addition to the input data entry, he will need to include an output data entry containing the data set out in paragraph 4.1 of this Annex above, i.e. a "value" to be transferred to Carol and a locking script to determine how Carol is able to spend that value.

4.3 Transaction validation

As noted in paragraph 3.2 of Annex 2 (*Overview and key features of DLT*), the transaction validation feature is designed to help to provide participants with a degree of exclusivity in relation to UTXOs and prevent them from being "double-spent". In most UTXO models, the transaction validation exercise will involve checking that (among other things):

(i) every transaction must prove that the sum of its inputs is greater than the sum of its outputs;

(ii) every referenced input must be valid and not yet spent; and

(iii) for every input, the transaction must have a signature which proves that the transferor has knowledge of the private key corresponding to the public key referenced in the relevant locking script for the outputs consumed by this input (this is the "signature script" which we discussed above in paragraph 4.2 of this Annex).

Note that no verification is made as to whether the total amount of UTXO available for a user to spend is greater than (or at least equal to) the amount proposed to be transferred in the transaction (i.e. no "account" balances are checked).

As discussed above, the population of participants who conduct this validation exercise may differ significantly between different DLT implementations, depending on what purpose the network is designed for. In certain Private and Permissioned models, recipients may be required positively to "accept" transactions and, in some models, the operator of the master node may also have the right to decline any transaction or reverse a transaction previously validated and recorded on the distributed ledger.

4.4 Transactions involving multiple inputs and outputs

The summaries above in paragraphs 4.1 and 4.2 of this Annex assume a chain of two transactions (i.e. from Alice to Bob and then Bob to Carol) which each purport to transfer the same value and, each time, only to one recipient address. However, in practice, transactions will

[56] The following description provided in the Bitcoin Whitepaper is helpful in illustrating this: "*We define an electronic coin as a chain of digital signatures. Each owner transfers the coin to the next by digitally signing a hash of the previous transaction and the public key of the next owner and adding these to the end of the coin. A payee can verify the signatures to verify the chain of ownership*." See: https://bitcoin.org/bitcoin.pdf (Accessed May 2019), p.2.

often involve: (i) outputs from several different transactions; (ii) outputs being sent from more than one address of a user; (iii) outputs being sent to different recipients, each with a different address; and (iv) the transfer of only a portion of the value of a spendable output. We discuss these further below.

4.4.1 Outputs from several different transactions, referencing the same public address

Here, Alice wants to transfer UTXOs with a value of 10 to Bob from one public address. Alice's public address is referenced by two separate UTXOs with a combined value of 10, which have been received as a result of two separate prior transactions with each UTXO with a value of 5. However, Alice wants to send the two separate UTXOs with a combined value of 10 in one single transaction.

In constructing her transaction message, the "input" will consume both separate UTXOs. As a result, the transaction identifier for her transaction will include the TINs and OINs for both prior transactions. Assuming both outputs to be spent reference the same public address of Alice, Alice will be able to satisfy each associated locking script using the same private key associated with her public address. The output entry (being the new UTXO) which Alice creates will specify a value of 10 and a locking script which sets the conditions to be satisfied by Bob, referencing Bob's public address.

Once the transaction is verified and subsequently recorded in a block, Bob will be able to spend a single UTXO with a value of 10. Any transaction he constructs to transfer that UTXO will only need to reference the one TIN of the transaction from Alice to Bob, and the OIN of that same transaction, which identifies the one UTXO with a value of 10 included in the transaction.

4.4.2 Outputs from several different transactions, referencing different public addresses

Here, Alice wants to send UTXO with a combined value of 10 to one public address of Bob, however she has two separate UTXOs, each with a value of five and each of which references a different public address of Alice. Alice will need to construct two transaction messages and satisfy the two locking scripts using two separate private keys. Each transaction will have to specify outputs with a value of five, and the locking script used in both would reference Bob's public address.

Once the two transactions are verified and subsequently recorded in a block, Bob will be able to spend the transferred output value of 10. Any transaction Bob constructs to transfer those outputs to Carol will need to include an input referencing both TINs of the transactions from Alice to Bob, and the OINs of the outputs (with a value of five each) of both of those transactions.

If Carol wants to transfer those UTXO with a value of 10 on to another third party, any transaction she constructs will just need to include an input which references the one TIN of the transaction from Bob to Carol, and the one OIN of the output (with a value of 10) of that transaction.[57]

[57] It is important to note that what appears on the distributed ledger is not what appears in a user's "coin" wallet. Most wallet software gives the impression that "coins" are sent from and to wallets, but really it is outputs and inputs moving from transaction to transaction. When a user's wallet records a balance of 100 "coins", what this really means in the UTXO model is that the user is able to spend unspent transaction outputs with a total value equal to 100.

4.4.3 Outputs from one transaction, only a portion of which is transferred

Here, Alice wants to send UTXO with a value of five to Bob. However, Alice has a single UTXO with a value of 10 (which was transferred to her as a single UTXO (with a value of 10) in one transaction and references one of Alice's public addresses). In the UTXO model, when a user transfers the unspent value from their address, every unspent output at that address which was received in the same transaction has to be retrieved, the amount to be transferred deducted, and then the remaining balance returned to a "change" address (i.e. a new address) of the transferor.[58] [59] The reasoning behind this is that each output from one transaction can only ever be referenced once by an input of a subsequent transaction. The return of the change balance to a "change" address is facilitated by the construction of another transaction which makes those excess outputs available for the transferor to spend using their private key.[60]

The process has been summarised as follows:

> *If a user wants to send a transaction sending X coins to a particular address, it may sometimes be the case that some subset of their UTXOs has a combined denomination of exactly X, in which case they can create a transaction that consumes those UTXOs and creates a new UTXO of value X owned by the destination address. When no such perfect match is possible, the user must include input UTXOs with a combined denomination greater than X, and add a second destination UTXO called a "change output" that assigns the excess coins to an address controlled by themselves.*[61]

Consequently, in order to effect the transfer described above, Alice must construct a transaction which consumes the entire UTXO with a value of 10, split between two separate outputs with two separate locking scripts on the outputs, each with an associated value of five. One locking script will be satisfiable by Bob's private key, and the other by Alice using her private key. The "change" address does not have to be the same address as that of the input (but it is likely to be).[62]

4.4.4 Outputs from one transaction, transferred to two different public addresses

This is essentially the same logic as that outlined in paragraph 4.4.3 of this Annex above, except that the two locking scripts will reference the two public addresses of the transferees, rather than one referencing Alice's "change" address.

4.4.5 Mixing outputs

In certain implementations, where a transaction makes a value available for a transferee to spend which is a combination of two separate outputs, once the transferee transfers

---

[58] With a portion of the transferred output going to miners in the form of unspent outputs (i.e. a "transaction fee").

[59] The Bitcoin Whitepaper summarises this process as follows: "*Although it would be possible to handle coins individually, it would be unwieldy to make a separate transaction for every cent in a transfer. To allow value to be split and combined, transactions contain multiple inputs and outputs. Normally, there will be either a single input from a larger previous transaction or multiple inputs combining smaller amounts, and at most two outputs: one for the payment, and one returning the change, if any, back to the sender*." See: https://bitcoin.org/bitcoin.pdf (Accessed May 2019), p.5.

[60] See: https://hackernoon.com/why-ethereum-when-we-already-have-bitcoins-blockchain-3359eb7e087e (Accessed May 2019).

[61] See: https://medium.com/@ConsenSys/thoughts-on-utxo-by-vitalik-buterin-2bb782c67e53 (Accessed May 2019).

[62] Most wallet software will usually use the same address as that of the input automatically.

that value to another user by referencing those outputs within an input and creating one combined output, it is likely to be impossible to determine which original transaction any subset of that transferred value originated from. However, we note that this may vary between implementations.

4.5 How do "new UTXOs"[63] come into existence?

As discussed in paragraph 2.4.1 of Annex 2 (*Overview and key features of DLT*), the circumstances in which new digital records (and, in the context of the UTXO model, new UTXOs) come into existence vary extensively between different implementations.

In every case, however, the introduction of value requires a transaction which includes a combination of "inputs" and "outputs", similar to any other UTXO transaction. What distinguishes this type of transaction from any other type is that it does not "consume" UTXO as inputs. Rather, it has only one input, and this input will not contain a reference to a prior output. This first transaction will, however, have an output (although only one), which will identify the recipient's public address in the locking script (meaning only the owner of the private key associated with that address is able to spend the value).

5 Cryptoassets (or distributed records) in the Account-Based model

In the case of certain cryptoassets, what is really being referred to is the "balance" or "value" recorded within an "account" that is available to whoever knows the private key associated with that account to "spend" by constructing a transaction message. Again, we draw on the explanations provided in Annex 2 (*Overview and key features of DLT*) and elaborate on them in the context of cryptoassets in the Account-Based model.

5.1 How are cryptoassets in the Account-Based model represented?

Much of the general description provided above in relation to UTXOs applies equally with respect to cryptoassets in the context of the Account-Based model. Like UTXOs, Account-Based cryptoassets are represented by a string of data, manifested as a readable sequence of characters.

Account-based distributed digital records are also generally generated by a transaction on the system, however, the data string which represents those cryptoassets (i.e. the data string recorded on the distributed ledger) is a data string that represents the entire balance of a user's account. This differs from the UTXO model, where the data string representing UTXO recorded on the distributed ledger only appears within a set of transaction data. This distinction goes to the heart of the difference between the UTXO model and the Account-Based model, which we discussed in paragraph 2.3.2 of Annex 2 (*Overview and key features of DLT*).

5.2 Transactions

Many of the core components required for constructing and effecting transactions are similar as between the Account-Based model and the UTXO model. However, there are two key differences as compared with the UTXO model:

[63] As discussed in paragraph 4.1 of this Annex above, strictly speaking, each transaction in a UTXO system generates a new UTXO, as the string of data representing each transaction output is unique even if its associated value remains the same. We refer to paragraph 2.4 of Annex 2 (*Overview and key features of DLT*) and, in particular, footnote 29, where we draw this distinction.

5.2.1 Transactions in Account-Based models do not rely on "unspent transaction outputs".

Nothing within the transaction message is required to reference any prior inputs, outputs or other prior transaction data in order to be valid. Essentially, all that is required for the transaction to be verified as valid by validator nodes is: (i) for the balance of the sending account to be sufficient to cover the balance being transferred; and (ii) for the signature to be valid. If both are valid, the sending account is debited and the receiving account is credited with the value.[64] [65]

One consequence of this is that, once the transferred value/balance is recorded to the recipient account's address, that value/balance will be indistinguishable from any other value/balance within the same account. Any subset of that value/balance can be spent using the cryptographic signature.

Equally, in contrast with the UTXO model, there is no "change address" in the Account-Based context either. This is a natural consequence of the fact that a transaction does not need to consume outputs as inputs in order to be valid.

5.2.2 A transaction which is recorded in a successfully-mined block leads to the balance of an account recorded on the distributed ledger being updated.

When a transaction is validated and included in a block, the distributed ledger will record a state change, adding the value that has been transferred to the balance of the transferee account address, and debiting the balance of the transferor's address. It is when that value has been updated within the transferee's account that it becomes available to be spent by whoever knows the associated private key. This contrasts with the UTXO model, where it is the point at which the relevant transaction is recorded on the distributed ledger that the associated UTXO becomes available to spend.

5.3 How do cryptoassets in the Account-Based model come into existence?

As with the UTXO model, distributed records representing a new value in an Account-Based model come into existence in a variety of ways. In contrast to the UTXO model, however, no input is required in order for the transaction creating a new value to be validated.

[64] See: https://h2o.law.harvard.edu/text_blocks/30595 (Accessed May 2019).

[65] One additional feature in many Account-Based models which is designed to protect against "double-spend" is for validator nodes to check the "account nonce" of any sending participant. In most Account-Based systems, for each new transaction constructed by the accountholder and recorded on the distributed ledger, a change in the account nonce will also be recorded on the distributed ledger as part of the account record. Where two transactions reference the same account nonce, this may mean that a user is attempting to spend the same data twice and validating nodes may reject the proposed transaction.

Annex 4
Smart contracts

1 Introduction

This Annex outlines at a high level the key elements of smart contracts, smart "legal" contracts and the overlap and interaction of these concepts with DLT.

2 What is a smart contract?

There is no single definition of a smart contract, with the term encompassing several different concepts. Developers and computer scientists would tend to use the term to describe computer code which, once initiated, will automatically execute certain tasks once pre-defined conditions are met, and the execution of which cannot be halted unless this possibility is contemplated by the relevant parties and embedded within the relevant computer code.[66]

In the context of DLT,[67] a smart contract is a computer code which runs on the nodes participating in the DLT network in order to execute a transaction or operation[68] that will (assuming the transaction/operation satisfies the relevant system rules and achieves the required consensus) result in a change that is recorded on the distributed ledger.

In the sense used in this paper, a smart contract refers to computer code that executes on the satisfaction of certain conditions. As discussed below in paragraph 5 of this Annex, a smart contract may or may not be, or be part of, a smart legal contract.

3 Smart contracts and DLT

Distributed ledger technology is often seen as a component part of a smart contract platform. It is, of course, possible to facilitate smart contracts without distributed ledger technology. Prior to the advent of DLT (and indeed now), parties were able to program computers to automatically execute a payment (for example) upon the satisfaction of pre-defined conditions.[69] This would, however, generally have required both parties to have programmed their own computers, and to run separate instances of the program on their system, creating the risk that the separate implementations of the code do not match, unless a single, trusted entity is selected to run the code for both parties.[70]

The key benefit of DLT, however, is that it resolves the need for a single trusted entity to run that code as opposed to being run across mutually "distrusting" entities. The "distributed" nature of DLT allows the relevant code to be embedded in the distributed ledger, meaning that there can be no dispute about what the code is (although not necessarily what the intended legal effect, if any, of the code should be) that the parties have agreed to and there is no separate

---

[66] Vitalik Buterin has referred to smart contracts in this sense as "persistent scripts", which is a helpful term in that it does not necessarily conclude as to the legal status of that computer code.

[67] Although we primarily discuss smart contracts in the context of DLT, we note that smart contracts can be instantiated wholly on a distributed ledger-based model, partially, or may not be instantiated on a distributed ledger at all.

[68] Note that we do not refer here to "message calls", i.e. lookups in smart contracts. Such lookups are not included in a block. Each interaction with a smart contract will either be a message call (where no "state change" results, if the relevant system is Account-Based) or a transaction (where a "state change" results, if the relevant system is Account-Based).

[69] ISDA Whitepaper: "Smart Contracts and Distributed Ledger – A Legal Perspective", p.8. See: https://www.isda.org/a/6EKDE/smart-contracts-and-distributed-ledger-a-legal-perspective.pdf (Accessed May 2019).

[70] ISDA Whitepaper: "Smart Contracts and Distributed Ledger – A Legal Perspective", p.8. See: https://www.isda.org/a/6EKDE/smart-contracts-and-distributed-ledger-a-legal-perspective.pdf (Accessed May 2019).

implementation. Tying this back to the description of DLT we provided in Annex 2 (*Overview and key features of DLT*), the reason for this is that the "terms" of the contract, comprising code, would be recorded on the distributed ledger. Once each block is validated according to the relevant consensus method, the security benefits of the technology (as described above) will apply to it. As such, there will only ever be one agreed version of the code constituting the smart contract embedded on the ledger.

4 Smart contracts and DLT: how it works in practice

In the context of many DLT implementations, what is meant by "smart contract code" is scripting language that is capable of running on a "distributed virtual machine" or other runtime environment, such as a "container".[71] In essence, a distributed virtual machine is a computer of which nodes on the DLT network run a local copy. Smart contract code will likely be written in high-level code (e.g. JavaScript[72] or Solidity[73]) or in a domain-specific language.[74] Depending upon the implementation, in order for the code to run and effect a change on the associated distributed ledger, it often has to be converted/compiled into bytecode to execute on the distributed virtual machine runtime.

The process by which this code is recorded on the relevant blockchain is similar to the process outlined in Annex 3 (*Cryptoassets*) in relation to transfers of value between participants. The code will need to be included within a transaction, and that transaction will have to be recorded on the distributed ledger. In an Account-Based model, any embedded smart contract function can be run by submitting a calling transaction to the contract's address, which causes the distributed virtual machine to run the contract code with the transaction as its input.

5 Smart legal contracts

The concept of a "smart contract" is a broader concept than a "smart legal contract". A smart contract may or may not have legal ramifications as it is merely computer code, whereas a "smart legal contract" refers to a smart contract that either is, or is part of, a binding legal contract. Whether English law recognises such a thing as a smart legal contract in this sense will be addressed in the Legal Statement. In any given case, this is likely to involve the application of established tests under English law for determining whether a binding legal contract arises. The concepts of offer, acceptance and consideration are likely to be relevant in this context.

6 Models of smart legal contracts

There are a variety of potential smart legal contract implementations. However, the three implementations that are often referred to are:

(i) the "**Solely Code Model**", i.e. code standing by itself (i.e. without being housed within any form of natural language contractual architecture);

(ii) the "**Internal Model**", i.e. a contract written in a document comprising natural language and code; and

---

[71] A container is software that packages up code so that an application can run quickly and reliably from one computing environment to another. See: https://www.docker.com/resources/what-container (Accessed May 2019).

[72] Which is used in Hyperledger Fabric. See: https://www.hyperledger.org/projects/fabric (Accessed May 2019).

[73] Being one of Ethereum's high-level coding languages.

[74] For example, the "Ergo" domain-specific coding language (see further here: https://docs.accordproject.org/docs/ergo.html (Accessed May 2019)) or the "Digital Asset Modelling Language" (see further here: https://daml.com/ (Accessed May 2019)).

(iii) the "**External Model**", i.e. a contract entirely in natural language but including agreement for certain aspects of the contract to be performed using a program designed for this purpose.[75]

We discuss these different implementations in greater detail below.

6.1 "Solely Code"

This form of smart legal contract implementation is often referred to as "contract *as* code". There is some debate as to whether the "Solely Code Model" is capable of existing apart from the "Internal Model", i.e. whether a "Solely Code" smart legal contract must always be part of a broader agreement between two parties. Setting that to one side, if we assume that the parties had exchanged versions of a piece of code, agreed to how it is written and offered consideration to each other for recording the code on the relevant distributed ledger,[76] all other aspects of the arrangement between the parties would take place on the relevant DLT network.

The "contract" would be expressed in high-level code written in a computing language which is supported by each distributed virtual machine run by a node on the DLT network. This high-level code would then be compiled/converted into binary code so that it is capable of being executed by such node. Where the effect of the high-level code is such that, when the code runs, it executes a transfer or payment between the parties, that action would run on the distributed virtual machine and subsequently be recorded as a transaction on the distributed ledger.

6.2 Internal Model

This form of smart legal contract implementation is essentially a case of "contract *containing* code". This model would be achieved by taking the same computing code discussed in paragraph 6.1 of this Annex above and housing it within a broader natural language contractual architecture (e.g. a 2002 ISDA Master Agreement) executed by the parties. The parties would look to agree within the natural language contract that the operative aspects of the contract they are entering into are to be performed as set out in the computer code. The code would then need to be recorded on the relevant distributed ledger, which (as outlined above) would require one of the parties to initiate a contract creation transaction and have that transaction (along with the associated code) recorded on the distributed ledger. All performance and execution of the code so recorded would then take place on the DLT network as outlined in paragraph 6.1 of this Annex above.

This will typically be implemented in two main ways:

(i) a natural language document with a provision (or series of provisions) expressed solely in code, with no natural language expression or representation of that provision; only the machine-readable representation; and

---

[75] See the ISDA Whitepaper: "Smart Contracts and Distributed Ledger – A Legal Perspective" for further details on the distinctions. The paper is accessible here: https://www.isda.org/a/6EKDE/smart-contracts-and-distributed-ledger-a-legal-perspective.pdf (Accessed May 2019).

[76] Certain smart contracting languages (e.g. Digital Asset Modelling Language) require smart contracts to be signed by the obligor party in order to be valid as a prerequisite for execution. The offeror of a transaction must present the offer to the offeree for the offeree's acceptance, rejection or counteroffer. Counteroffers are presented back to the offeror for acceptance, rejection, or another counteroffer. If there are more than two parties to a transaction, the offer is presented to all participants in the same manner. Only when all participants have accepted the offer and have digitally signed the smart contract with their cryptographic signatures, or private keys, is the smart contract valid and ready for execution. See further here: https://daml.com/ (Accessed May 2019); and "The only valid smart contract is a voluntary one—easier said than done" https://medium.com/daml-driven/the-only-valid-smart-contract-is-a-voluntary-one-easier-said-than-done-726df37c04c (Accessed May 2019).

(ii) a natural language document with a provision (or series of provisions) expressed both in code and natural language, meaning that the code is effectively bound to a natural language representation.[77]

With both paragraphs (i) and (ii) above, in the event of a dispute, the parties would have the natural language contractual architecture to fall back on.

6.3 External Model

This form of smart legal contract implementation is often referred to as "contract ***supporting*** code". The model is similar to the model outlined at paragraph 6.2 of this Annex above, save that the relevant smart contract code would not be included within the written contract itself.

In this model, the smart legal contract exists as a written natural language document and is paired with code that is instantiated in a manner external to the written document. This may be, for example, a script which exists on a distributed ledger or a cloud-based system which is used to automate the execution of all or part of an agreement represented by the natural language document. There is some debate as to whether the "External Model" in fact represents a smart legal contract, or whether there is simply a contract (in prose), coupled with a smart contract (i.e. computer code) which performs certain aspects.

6.4 Concluding commentary on smart legal contract models

The models outlined above are intended to describe architectural paradigms, the aim of which is to demonstrate that there is a continuum between the contract as code at one end (i.e. the Solely Code Model), a hybrid binding of language and logic in the middle (i.e. the Internal Model), and the contract and code as separate, but related, entities at the other (i.e. the External Model).

It is important to note that it is also possible to have hybrid approaches. For example, a given implementation conforming to the Internal Model description could be executed in a non-DLT environment and could be designed to call smart contract code on a distributed ledger.

For example, a sales contract may include a line of code which reflects the natural language of a pricing provision, which has the effect of calculating the price payable upon the occurrence of some external factor, such as a particular exchange rate threshold. This form of smart contract conforms to the definition of an Internal Model. However, the contract may be written such that the effect of the code running is to trigger a call to some on-chain code with no natural language equivalent, and the running of this on-chain code could initiate a transfer of value between the two parties on the distributed ledger. This is a feature which is more consistent with an External Model and illustrates how the two models may be combined.[78]

[77] This is typically achieved by having both the natural language text and the code refer to a data model that sits behind both the natural language and the code. This data model may specify data types for variables expressed in the natural language of the written agreement, and these variables can then be used by the code. Essentially, this attempts to ensure a degree of consistency or validation between the natural language and code. An example of such a data model would be the Common Domain Model produced by the International Swaps and Derivatives Association. Please see here for further information: https://www.isda.org/2018/11/22/isda-cdm-factsheet/ (Accessed May 2019).

[78] This combined form of implementation may also avoid the need to run the entirety of code on the DLT network, which may have benefits for the parties in terms of privacy.