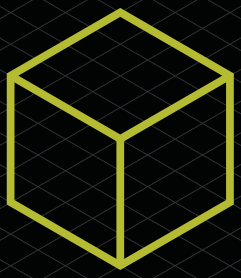
Part 1: Developing Technologies

Section 1

An Overview of DLT



Section 1: An Overview Of DLT

Tom Grogan, MDRxTech LLP; Water Hernandez-Cruz, Mishcon de Reya LLP

For readers less familiar with the concepts explored in this guidance, this section gives an overview of distributed leger technology (DLT). It shows how the use of ledgers has evolved, identifies some of the main characteristics of DLT, explores the mechanisms by which some distributed ledgers create, amend and replicate their digital records and provides brief examples of different types of DLT – showing how blockchain, although the best know example, is not the only one.

The evolution of ledgers

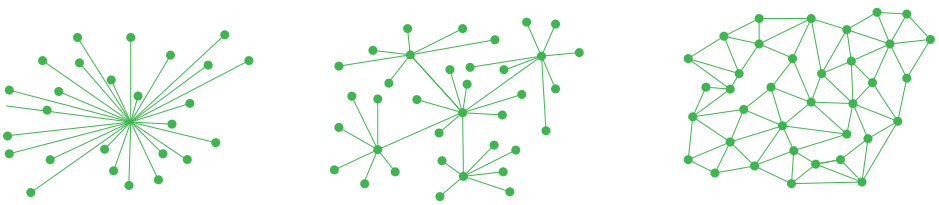
DLT refers to a group of technologies that use different techniques and structures to store, synchronise and maintain a shared ledger of digital records across a network of computing centres.

The idea of maintaining a ledger is not a new one. The earliest ledgers date back to c.4,000BC in Mesopotamia. These ledgers were kept on clay scripts or carved into stone, and were used to record and demonstrate definitive ownership, and the transfer of ownership, of crops in storage. Recording the ownership and movement of value has been a central tenet of human civilisation ever since. The form and structure of these ledgers however has evolved (and continues to evolve) with time.

The Mesopotamian example describes what we now call a centralised ledger (see Fig 1 below), in which a single definitive ledger exists within an ecosystem. In many circumstances, such centralised ledgers are effective, and many remain in use today. Centralised ledgers do however have some drawbacks, notably that they have a single point of failure (i.e. the single ledger). If the ledger is lost, stolen or attacked (i.e. tampered with by a third party), the ecosystem and its participants (those placing reliance on the definitive nature of the ledger's record keeping) will fail. As an ecosystem becomes more complex and its value rises, the use of a centralised ledger becomes less appropriate.

As civilisation has developed, so too have decentralised ledgers become more prevalent (see Fig 1). In modern society, we often rely on trusted intermediaries to keep and maintain common ledgers. These intermediaries may for example be financial institutions, which keep and maintain ledgers relating to our finances. Decentralised ledgers, just like their centralised cousins, are widely used today but also have their own drawbacks. They too have points of failure which can have widespread impact on the wider ecosystem – see for example the damage caused when a financial service provider's IT infrastructure suffers an outage. They also rely heavily on the trustworthiness and integrity of the intermediary maintaining the decentralised ledger – if this intermediary causes loss to its stakeholders through negligence or fraud, those stakeholders often have limited recourse.

Distributed ledgers seek to avoid the drawbacks associated with centralised and decentralised ledgers by, amongst other things, removing points of failure (see Fig 1). Distributed ledgers see the ledger (or parts of the ledger) replicated and stored across a network of computing centres. This network of computing centres, known as nodes, work to update the ledger as new updates (i.e. transactions) arise, and propagate the updated ledger to the network. Distributed ledgers are, theoretically, infinitely scalable, and by distributing their control and maintenance, seek to mitigate against the risk of attack.



Centralised Decentralised Distributed

Fig 1 – Centralised, decentralised, and distributed ledgers. Note that the structures of these ledgers, in particular the distributed ledger, have been simplified for illustrative purposes.

In this guidance we use the term **cryptoassets** loosely to mean an asset of whatever kind that is represented digitally on a DLT platform. Such assets might exist purely digitally, for example a so-called cryptocurrency such as Bitcoin (BTC), or physically, for example a piece of real estate that is represented by way of tokenisation. In line with terminology used by the Financial Action Task Force (FATF), **cryptoassets** are in this guidance occasionally also referred to as 'virtual assets'. This guidance distinguishes between **cryptoassets** which, in line with the UKJT Legal Statement, we hold to be capable of constituting property as a matter of English private law, and records, which we typically consider to be pure data and therefore not capable of constituting property as a matter of English private law.

We also refer to **wallets**. Again, we use this term broadly to mean the digital device used to store a user's **public and private keys**, which are used to manage and control the user's DLT-stored records and/or cryptoassets. Please see Fig 2 below for details regarding the purpose and functionality of public and private keys in the context of DLT systems.

DLT is a rapidly evolving area of computer science and the limitations of this section are acknowledged. It does not seek to provide an exhaustive and detailed explanation of DLT, rather, it seeks to:

1. set out the main features of DLT;

2. explain consensus protocols; and

3. give brief examples of DLT types.

1. Main features of DLT

A series of mechanisms and computer protocols dictate how distributed ledgers work – namely, how their network participants may create, amend and synchronise records held on them. These mechanisms and computer protocols typically seek to:

i. enable network participants to **exclusively** control 'their' records or cryptoassets;

ii. maintain a clear **chronology** of distributed ledger entries; and

iii. provide a mechanism by which network participants will reach a **consensus** as to new distributed ledger entries and the state of the distributed ledger from time to time, thereby ensuring a common, synchronised ledger.

These three components represent key features of DLT. Each of them is explored below in more detail.

i. Exclusivity
To enable network participants to exclusively control 'their' records or cryptoassets, most DLT implementations utilise public key cryptography.

Public key cryptography is a cryptographic system that uses two types of information (typically a fixed length string) known as keys:

a. public keys: these may be widely disseminated and known to some or all other network participants; and

b. private keys: these should be known only to the relevant network participant.

If a network participant wishes to send a message (or, in the case of cryptoassets, make a transaction), they would enter their message (or transaction details) together with the intended recipient's public key (or a hash of the intended recipient's public key, known as a wallet address).

The network participant who is sending the message (or transaction) then 'signs' the message (or transaction) using their private key. The recipient, and the wider network, is then able to verify that the message (or transaction) is genuine, by entering the public key of the network participant who sent the message (or transaction). When combined, the message (or transaction) will (provided the public key entered is indeed associated with the private key used to send the message or transaction) be decrypted.

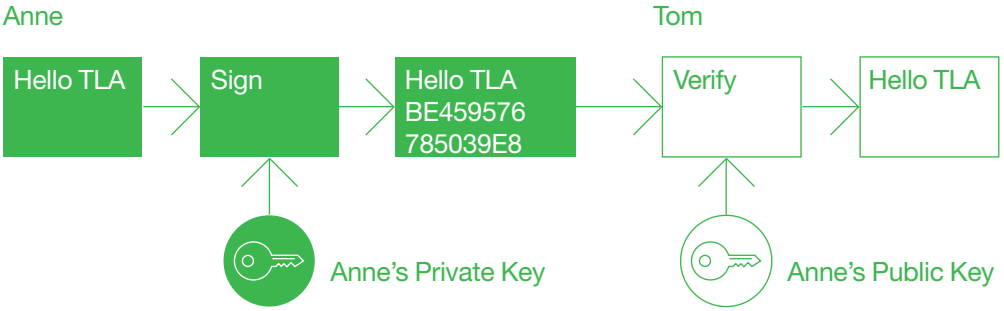


Fig 2 – Public key or asymmetrical cryptography-enabled messaging

Public key cryptography is also known as asymmetrical cryptography. This is because a message (or transaction) which was encrypted using the sender's private key, can be decrypted using the sender's public key, without revealing or compromising the security of the sender's private key.

An important conceptual point to grasp is that wallets do not contain records or cryptoassets. All that is contained in a wallet is a private key. Accordingly, when we make a new record or transaction on a distributed ledger, we do not 'send' records or cryptoassets per se, rather we send a message or transaction to the network's nodes, which then update their respective copies of the ledger accordingly.

DLTs therefore enable exclusive ownership of records and cryptoassets by ensuring that the right to send messages (or make transactions) on behalf of a public key relies on a private key, which is capable of being kept secret and known only to a single individual. In this way, an individual can be said to 'own' (albeit indirectly) certain cryptoassets.

ii. Chronology

One of the main challenges that faces a distributed ledger is how to establish a clear chronology of records or transactions. As the network becomes larger and more distributed across territories and time zones, so the so-called 'Distributed Ledger Problem' becomes more pronounced.

The Distributed Ledger Problem

Records and transactions are passed from node to node within the network, and therefore the order in which transactions reach each node can differ.

For example, say an attacker has a wallet holding 1 TLA Coin (a fictional cryptoasset used for illustrative purposes only). Exploiting the Distributed Ledger Problem, the attacker may make a purchase from a supplier of goods and send 1 TLA Coin to the supplier as payment. The attacker would then wait for confirmation that the supplier has shipped the goods. Once the attacker has received the confirmation, he or she would then send a transaction to another of his wallets for 1 TLA Coin. Due to the Distributed Ledger Problem, some nodes might receive the second transaction before the first. Those nodes would then consider the initial transaction invalid, as the transaction inputs would be marked as already spent. If sufficient nodes to satisfy the distributed ledger's consensus protocol believed the second transaction to be the 'true' transaction, the transfer of TLA Coin to the supplier would be rejected and the supplier, having already shipped the goods, would be out of pocket.

The way in which DLTs establish a clear chronology of records and transactions is typically determined by the manner in which their ledger dataset is structured. This varies from DLT to DLT – see (3) below for some high-level examples of different forms of DLT.

iii. Consensus
Each DLT node has its own view of the state of the distributed ledger at a given time. The result of this, exacerbated by the Distributed Ledger Problem set out above, is that, at any one time, there may be as many views of the present state of the ledger as there are nodes in the network.

Distributed ledgers implement clear rules to enable their constituent nodes to reconcile differences and record messages and transactions in a harmonious fashion. These rules are known as consensus protocols. There are a number of 'flavours' of consensus protocols, each with their own trade-offs that in turn impact on the distributed ledger's performance and functionality. See below for some high-level examples of consensus protocols.

2. Consensus protocols

A range of different consensus protocols might be adopted by DLTs. The following is a high-level overview of two well-known examples: proof of work and proof of stake.

i. Proof of work
Proof of work requires participating nodes (known as 'miners') to prove that computational resource has been committed before a record of transactions can be accepted as part of the distributed ledger. Proof of work is perhaps the best-known example of a consensus protocol and is used by the Bitcoin (BTC) blockchain.

In order to prove their commitment of computational resource, miners 'race' to solve a computational puzzle which is designed to require a large number of computational steps without shortcuts. Once solved, the successful miner can broadcast the answer to the puzzle to the DLT's node network, which can then easily and quickly verify the answer as being correct and thus accept the new entry to the ledger. Most DLTs require a majority of nodes to verify the puzzle answer in order to accept the entry of the new records or transactions to the ledger. Typically, in DLTs that use proof of work, mechanisms are built in to reward and incentivise miner activity.

Proof of work's advantages include that it is secure (subject to a well distributed network of computing power), it deters spam (by requiring miners to expend effort in order to successfully enter new ledger entries), and it is democratic (as the same puzzle is posed to all miners). It has however been criticised for being, amongst other things, relatively slow, expensive (owing to the hardware required to give miners a reasonable prospect of success, which undermines its democratic credentials), and environmentally unfriendly (owing to the energy consumption associated with mining activity).

ii. Proof of stake
Proof of stake requires each node that seeks to update the ledger to prove that it has a 'stake' in the system. In 2022 we saw the Ethereum Foundation complete The Merge, leading to the adoption of proof of stake by the Ethereum blockchain network. Other well-known implementations of proof of stake include Stellar, DASH and NEO.

To establish a new ledger entry, competing nodes (known as 'validators') construct a particular type of transaction that 'locks-up' their funds in a form of deposit. Validators then take turns proposing and voting on the next ledger entry. The weight of each validator's vote is proportionate to the size of its lock-up. If a majority of validators reject a proposing validator's ledger entry, the proposing validator loses its lock-up.

In addition to deterring validators from proposing fraudulent new entries (for fear of losing their lock-up), proof of stake DLTs also ensure that the state of their ledger is dictated by those invested in them – those investors will wish to ensure the integrity of the ledger as, if doubt is cast upon it, the value of the DLT (and in turn the investor's investment) will diminish. Other advantages of proof of stake include that it is quicker and more energy efficient than some other consensus protocols (such as proof of work). Disadvantages of proof of stake include that is more difficult to secure and can be seen as undemocratic.

3. Examples of DLT

i. Blockchain
ii. Directed acyclic graph
iii. Hedera Hashgraph

i. Blockchain

The best-known example of a DLT is blockchain, which rose to prominence on the publication of the Bitcoin white paper in 2008 under the pseudonym Satoshi Nakamoto. Blockchains bundle digital records into data container structures known as 'blocks'. These blocks are appended to the end of a chain of blocks in chronological order, hence the name.

Typically, each block in a blockchain will contain a hash of the preceding block. This ensures that a clear, irrefutable chronology is established and maintained.

Journal ID	Date Stamp	From	To	Currency	Amount
1	26.01.2019 08:35	Barclays	HSBC	GBP	500.00
2	28.01.2019 10:50	Barclays	Santander	GBP	4,250.00
3	29.01.2019 12:00	Santander	Barclays	GBP	2,000.00
4	28.01.2019 10:50	HSBC	Santander	GBP	100.00

Block 1			Block 2			Block 3		
Data	Prev Hash	Current Hash	Data	Prev Hash	Current Hash	Data	Prev Hash	Current Hash
1	100	101	1	101	102	1	102	103

Fig 3 – Blockchain structure

ii. Directed acyclic graphs

Directed acyclic graphs (DAGs) are a well-established branch of graph theory and computer science. They are graphs that travel in one direction without cycles connecting the other edges. The graph uses topological sorting, wherein each node is in a certain order. In the context of DLT however, directed acyclic graphs present an exciting alternative to blockchain database structuring.

The one-directional nature of a directed acyclic graph ensures that a clear chronology can be maintained, while the impossibility of 'loops' mitigates against the risk of 'double-spend', which is often associated with distributed ledgers. The consensus protocols typically adopted by directed acyclic graph DLTs prevent against network participants validating their own transactions (save by chance) and can allow for multiple transactions to be simultaneously verified, thereby improving performance.

In graph theory, vertices or nodes represent entities in the network. In a distributed network, each computational centre is a node. Edges convey information about the relationship or link between nodes. In a distributed network, such relationships or links might include communications between computational centres.

Depending on the relationship between the nodes, several types of graphs emerge:

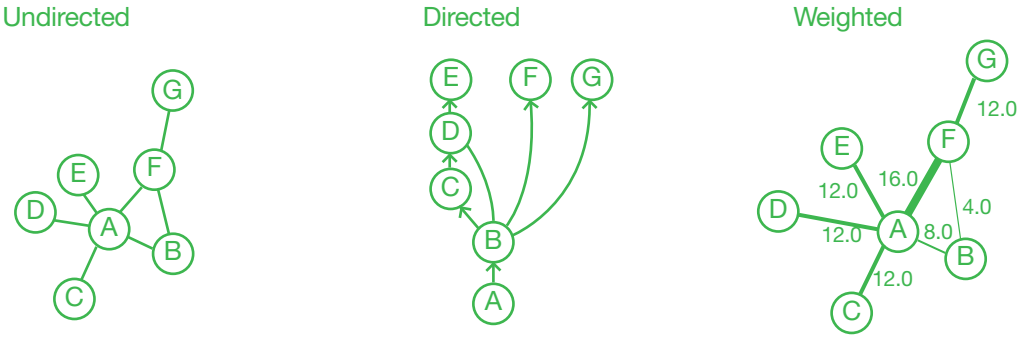


Fig 4 – Forms of acyclic graphs

— **Undirected:** An edge connects all nodes. The Facebook social media platform is an example of an undirected graph: when two users connect as Friends, both parties follow each other.

— **Directed:** The edge displays the directionality of the relationship from one node to another. The Twitter social media platform is an example of a directed graph: a user might connect with another user by Following them, without receiving a Follow back.

— **Weighted:** The edge sizes represent the strength of a relationship. Many corporate CRM tools are examples of weighted graphs, by making connections between users based on the strength of interpersonal relationships.

Specifically, DAGs are directed graphs because it is possible to infer the direction of how one node relates to another. In the case of DLT, DAGs' nodes or vertices represent or hold the information of transactions or events, while edges indicate the ordering of the transactions. The application of DAGs as a DLT presents the benefit of processing several transactions or events simultaneously while allowing the consensus to decide the proper order of the transactions.

iii. Hedera Hashgraph

Hedera Hashgraph is an alternative DLT and close cousin of the directed acyclic graph, developed by Leemon Baird in 2016.

Hashgraph is perhaps best known for its so-called ‘gossip protocol’, whereby every node spreads ‘gossip’ regarding its information (i.e. records or transactions, known in Hashgraph as ‘events’) and events it has heard (via the gossip protocol) from others, to two randomly chosen neighbourswhich in turn further propagate the gossip alongside their own events in an aggregated fashion). Chronologies are established using timestamped events.

The advantages of Hashgraph’s streamlined consensus mechanism include speed and fairness. A potential disadvantage is Hashgraph’s inherent assumption that fewer than a third of nodes are bad actors (i.e. those who forge, delay, replay and drop incoming and/or outgoing events): if this is not (or cannot be reliably be proved to be) the case, security concerns may arise.

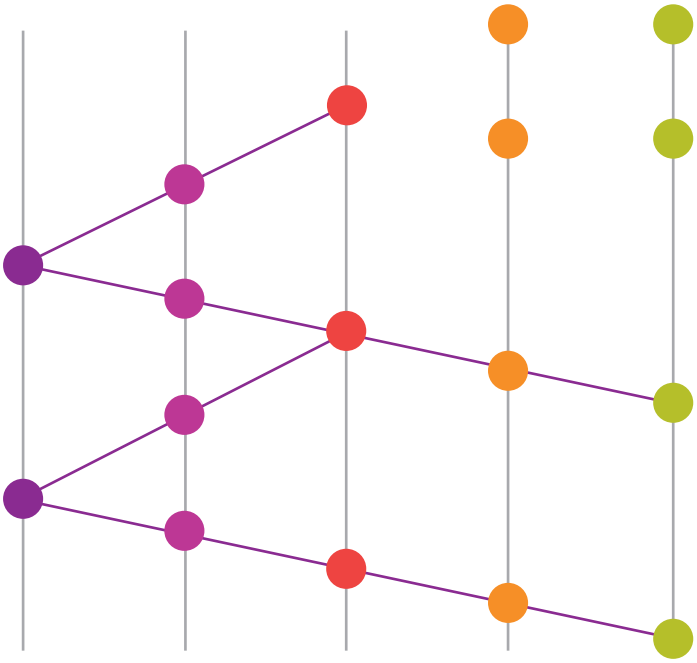


Fig 5 Hedera Hashgraph structure

4. Layer 2 protocols and beyond

In recent years so-called layer 2 protocols have emerged as a key feature of the DLT ecosystem. Layer 2 protocols are separate protocols which may or may not themselves be DLTs, which operate on top of underlying DLTs. Polygon is perhaps the most well known layer 2 protocol, which operates on top of the Ethereum blockchain.

Typically, a layer 2 protocol receives and processes user transactions, and periodically writes aggregated updates to the underlying DLT. In this way, layer 2 protocols are often seen as a scaling solution to DLT, enabling faster settlement times and lower transaction fees. They are not without their drawbacks, and thoughtful implementations should consider how best to obtain the security benefits associated with DLT while also availing themselves of the scalability afforded by layer 2 protocols.

In more recent times we are beginning to see the emergence of so-called layer 3 solutions. In these implementations we must trade-off between additional complexity and benefits.

To be clear, layer 2 or layer 3 procotols need not be themselves distributed ledgers, and careful thought should be given as to the most appropriate structure for a given implementations.