International Telecommunication Union

# ITU-T F.751.0

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

(08/2020)

SERIES F: NON-TELEPHONE TELECOMMUNICATION SERVICES

Multimedia services

# Requirements for distributed ledger systems

Recommendation ITU-T F.751.0

International Telecommunication Union

ITU-T F-SERIES RECOMMENDATIONS

**NON-TELEPHONE TELECOMMUNICATION SERVICES**

| | |
|---|---|
| TELEGRAPH SERVICE | |
| Operating methods for the international public telegram service | F.1–F.19 |
| The gentex network | F.20–F.29 |
| Message switching | F.30–F.39 |
| The international telemessage service | F.40–F.58 |
| The international telex service | F.59–F.89 |
| Statistics and publications on international telegraph services | F.90–F.99 |
| Scheduled and leased communication services | F.100–F.104 |
| Phototelegraph service | F.105–F.109 |
| MOBILE SERVICE | |
| Mobile services and multidestination satellite services | F.110–F.159 |
| TELEMATIC SERVICES | |
| Public facsimile service | F.160–F.199 |
| Teletex service | F.200–F.299 |
| Videotex service | F.300–F.349 |
| General provisions for telematic services | F.350–F.399 |
| MESSAGE HANDLING SERVICES | F.400–F.499 |
| DIRECTORY SERVICES | F.500–F.549 |
| DOCUMENT COMMUNICATION | |
| Document communication | F.550–F.579 |
| Programming communication interfaces | F.580–F.599 |
| DATA TRANSMISSION SERVICES | F.600–F.699 |
| **MULTIMEDIA SERVICES** | **F.700–F.799** |
| ISDN SERVICES | F.800–F.849 |
| UNIVERSAL PERSONAL TELECOMMUNICATION | F.850–F.899 |
| ACCESSIBILITY AND HUMAN FACTORS | F.900–F.999 |

*For further details, please refer to the list of ITU-T Recommendations.*

# Recommendation ITU-T F.751.0

## Requirements for distributed ledger systems


### Summary

Recommendation ITU-T F.751.0 establishes basic and advanced requirements for distributed ledger systems.



### History

| Edition | Recommendation | Approval | Study Group | Unique ID* |
|---|---|---|---|---|
| 1.0 | ITU-T F.751.0 | 2020-08-13 | 16 | 11.1002/1000/14332 |

### Keywords

Blockchain, consensus, decentralized ledger technology, non-repudiation, tamper-resistant.


* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

## FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

## NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

## INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

## Table of Contents

# Recommendation ITU-T F.751.0

## Requirements for distributed ledger systems

### 1 Scope

This Recommendation establishes requirements for distributed ledger systems, covering:

– a general overview;

– basic requirements;

– advanced requirements.

### 2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

None.

### 3 Definitions

#### 3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

**3.1.1 digital signature** [b-ITU-T X.800]: Data appended to, or a cryptographic transformation of a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery, e.g., by the recipient.

**3.1.2 Internet of things (IoT)** [b-ITU-T Y.2060]: A global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies.

**3.1.3 private distributed ledger technology system** [b-ISO 22739]: DLT system that is accessible for use only to a limited group of DLT users.

**3.1.4 public DLT; public distributed ledger system; public distributed ledger technology system** [b-ISO 22739]: DLT system which is accessible to the public for use.

#### 3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

**3.2.1 asset**: A representation of value. It can be a diamond, a unit of currency, items inside a shipping container, etc. An asset can be physical or virtual.

**3.2.2 blockchain**: A type of distributed ledger that is composed of digitally recorded data arranged as a successively growing chain of blocks with each block cryptographically linked and hardened against tampering and revision.

NOTE – Adapted from [b-ITU-T TS FG DLT D1.1].

**3.2.3 consensus**: Agreement that a set of transactions is valid.

NOTE – Originally published in [b-ITU-T TS FG DLT D1.1].

**3.2.4 consensus algorithm**: A set of rules that precisely establishes a sequence of operations to reach consensus.

**3.2.5 distributed ledger**: A type of ledger that is shared, replicated, and synchronized in a distributed and decentralized manner.

NOTE – Originally published in [b-ITU-T TS FG DLT D1.1].

**3.2.6 distributed ledger system**: A system that implements a distributed ledger.

**3.2.7 distributed ledger technology application; ledger application**: A client to a distributed ledger technology (DLT) system that is designed to help people perform an activity.

**3.2.8 entity**: Anything that has a separately identifiable existence (e.g., organization, person, group, smart contract or device). An entity uses distributed ledger technology to solve the problem of its business or information systems.

NOTE – Adapted from [b-ITU-T X.1215].

**3.2.9 entity address**: Identifier for one or more entities performing transactions or other actions in a blockchain or distributed ledger network.

**3.2.10 ledger**: Information store that keeps final and definitive (immutable) records of transactions.

NOTE – Originally published in [b-ITU-T TS FG DLT D1.1].

**3.2.11 mining**: A reward-seeking activity in some consensus mechanisms.

**3.2.12 node**: Device or process that participates in a distributed ledger network.

NOTE 1 – Nodes can store a complete or partial replica of the distributed ledger.

NOTE 2 – Originally published in [b-ITU-T TS FG DLT D1.1].

**3.2.13 P2P**: An instantiation of network architectures where all peers have equivalent authority and responsibility, differing completely from that of server and client system.

NOTE – Paraphrased from clause 1 of [b-ITU-T X.1161].

**3.2.14 permissioned distributed ledger system**: Distributed ledger system in which permissions are required to maintain and operate a node.

NOTE – Originally published in [b-ITU-T TS FG DLT D1.1].

**3.2.15 permissionless distributed ledger system**: Distributed ledger system where permissions are not required to maintain and operate a node.

NOTE – Originally published in [b-ITU-T TS FG DLT D1.1].

**3.2.16 smart contract**: Program written on the distributed ledger system that encodes the rules for specific types of distributed ledger system transactions in a way that can be validated, and triggered by specific conditions.

NOTE – Adapted from [b-ITU-T TS FG DLT D1.1].

**3.2.17 token**: A digital representation of value on a shared distributed ledger that is owned and secured using cryptography to ensure its authenticity and prevent modification or tampering without the owner's consent.

NOTE – Originally published in [b-ITU-T TS FG DLT D1.1].

**3.2.18 transaction**: An incident or an operation that leads to a change in the status of a ledger, such as adding a record or equivalent exchange based on currency.

## 4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

DDoS Distributed Denial of Service

DLT Distributed Ledger Technology

DoS Denial of Service

ID Identifier

IoT Internet of Things

P2P Peer to Peer

PoW Proof of Work

SPV Simplified Payment Verification

URL Uniform Resource Locator

## 5 Conventions

In this Recommendation:

The keywords "is required to" indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this Recommendation is to be claimed.

The keywords "is recommended" indicate a requirement which is recommended but which is not absolutely required. Thus, this requirement need not be present to claim conformance.

The keywords "can optionally" indicate an optional requirement which is permissible, without implying any sense of being recommended. This term is not intended to imply that the vendor's implementation must provide the option and the feature can be optionally enabled by the network operator or service provider. Rather, it means the vendor may optionally provide the feature and still claim conformance with this Recommendation.

## 6 Overview

Distributed ledger systems have no central administrator or centralized data storage. These systems form tamper-resistant, shared ledgers among distributed nodes [b-Zheng]. Besides the facilitation of the processing of transactions, distributed ledger systems enable the incorporation of data integrity and trust capabilities into applications.

DLT can be integrated into multiple areas. The World Bank thinks that DLT could fundamentally change the financial sector, making it more efficient, resilient and reliable [b-Natarajan]. DLT also has the potential to transform various other sectors, e.g., manufacturing, clean energy, health care and the IoT. More and more public or commercial services are piloting DLT.

DLT can be classified in two ways: public or private DLT; permissioned or permissionless DLT [b-Gramoli]. A public DLT has transaction records that are readable by anyone, while a private DLT limits the transaction records read access to some authorized groups. In a permissionless DLT, any computer can be a full functional node. In a permissioned DLT, only an authorized terminal can join the DLT network as a fully functional node.

## 7 Basic requirements for distributed ledger systems

This clause addresses basic requirements that apply to all distributed ledger systems, if not stated clearly otherwise.

### 7.1 Decentralization

Differing from many traditional systems, the operation of DLT relies on cooperation of multiple peer-to-peer (P2P) nodes. Each node maintains a partial or total local copy of the shared ledger, and validates the messages from others.

DEC-001: A distributed ledger system is required to maintain a shared ledger.

DEC-002: A distributed ledger system is required to identify each node on the network by a node identifier (ID).

DEC-003: A distributed ledger system is required to implement an addressing system. Each entity gets its address from the addressing system. The address is then being utilized by transactions to identify the senders and receivers.

DEC-004: A distributed ledger system node is required to be able to discover other nodes. Many technologies can be used for node discovery, such as bootstrapping, flooding or rumour mongering [b-Eugster].

DEC-005: A distributed ledger system is required to adopt a P2P network as its underlying infrastructure. Each node in the network exchanges messages directly with all others instead of via a central gateway. A particular P2P protocol is required to maintain the P2P network structure. The P2P network is required to support:

– data synchronization without connecting to a central server to get the ledger data;

– maintenance of node information and tracking the status of the nodes.

DEC-006: A distributed ledger system is required to send out broadcast messages. Broadcast messages are sent to all the other nodes in the DLT network.

DEC-007: A distributed ledger system is required to provide mechanisms to distribute the shared ledger among its nodes. This requires the P2P architecture to allow large amounts of data to be transferred between nodes. To distribute the shared ledger, a common practice is to broadcast the transactions, then have the consensus algorithm determine the final state of the shared ledger.

DEC-008: Any node of a distributed ledger system is required to be able to answer read-only transaction queries about the data they store. There is no need to forward a transaction to P2P network in this scenario.

DEC-009: A distributed ledger system is recommended to support different node types, e.g., light, full and full archiving.

DEC-010: A distributed ledger system can optionally support the execution of a transaction only if specific conditions are met. For example, a transaction must be executed within 24 h or a transaction dealing with more value than a defined threshold can be executed only after some days.

## 7.2 Proof of ownership

A distributed ledger is an information store that can be used to keep final and definitive (immutable) records of transactions. It keeps track of who owns an asset. In other words, a ledger records the transfer of assets from one owner to another. Assets may be physical or intangible. In DLT, any entity can be the owner of an asset. An entity can be a human, an organization, a smart contract, or a device. During their lifecycle, assets may be transferred multiple times and may be split or reorganized.

RWO-001: A distributed ledger system is required to adopt public key cryptology in its addressing system. Assets on the distributed ledger system are associated with private keys, and public key cryptology ensures data integrity, authentication and non-repudiation when assets are transferred.

RWO-002: A distributed ledger system is required to support transaction recording. Transactions are the fundamental activities that modify a shared ledger. Transactions may happen among multiple entities.

RWO-003: A distributed ledger system is required to record the addresses involved in a transaction and the assets influenced. The basic transaction is from one sender to one receiver, i.e., one to one. For example, a transaction records that address A sends *k* tokens to address B.

RWO-004: A distributed ledger system is required to have all transactions timestamped.

RWO-005: Because transaction recording is distributed, to make the timestamp consistent and valid, a distributed ledger system is required to have all nodes with transaction recording rights time synchronized.

RWO-006: A distributed ledger system is recommended to enable its transaction records to support supplementary information. For example, for a transaction in which A buys a house from B, the record contains not only the amount of tokens paid, but also the certificate of the house or the hash value of the certificate with the uniform resource locator (URL) of the certificate file.

RWO-007: A distributed ledger system is required to have each transaction digitally signed by its authorized entity using its private key.

RWO-008: A distributed ledger system is recommended to support multi-signature. If a transaction requires more than one entity to authorize, the usage of multi-signature is more compact than a collection of distinct signatures from all authorizers. Multi-signature also allows that *m* members of a group to authorize a transaction, although there are *n* members in total for the group, i.e., *m*/*n* authorization.

RWO-009: A distributed ledger system is required to support the storage of transactions. Multiple copies should be stored by at least all entities involved in a transaction.

RWO-010: A distributed ledger system is required to support distributed ledger storage. Multiple copies are required to be stored.

RWO-011: A distributed ledger system can optionally support functionalities that help users if they lose their private keys. In some use cases, such functionality should only be controlled by the user.

RWO-012: A distributed ledger system can optionally support the specification of additional names for its addresses.

RWO-013: A distributed ledger system can optionally support the possibility of segregating those who sign a transaction from those who publish it on the network.

### 7.3 Data validity

Each node operates to supervise all others in the DLT network because there are no trustable centralized nodes. Messages received from others require verification to confirm the validity of the information.

DV-001: A distributed ledger system is required to provide mechanisms to validate transactions received from others.

DV-002: A distributed ledger system is required to support:

– examination of the transaction record to validate that it has not been tampered with;
– examination of the address and signature of the sender to validate whether the sender is valid and authentic;
– if a transaction involves token transfer, examination of the balance of the sender to validate that it has enough tokens to fulfil the transaction;
– if the transaction involves the execution of a smart contract, execution of the specific function.

DV-003: A distributed ledger system is required to provide mechanisms to validate the shared ledger, so that only valid transactions are recorded in it.

DV-004: A distributed ledger system is required to support examination of the shared ledger:

– to validate that it has not been tampered with;
– to confirm that there are no suspicious transactions or accounts.

DV-005: A distributed ledger system is recommended to adopt mechanisms to facilitate the validation of a shared ledger on storage-limited devices, such as the mobile phones, and to provide mechanisms to accelerate the validation. Simplified payment verification (SPV) is one such mechanism. SPV nodes do not have all transactions and do not download the full shared ledger. To verify that a transaction is included in a blockchain ledger, without having to download all the transactions in the block, SPV nodes use an authentication path, such as Merkle path.

DV-006: A distributed ledger system is required to protect the system from fraud, such as:

– transaction fraud, e.g., the entity sends a transaction that exceeds its payment capability;
– erroneous response – a node returns an opposite value, e.g., the value of the execution result is 0, but the node returns 1.
– fraud broadcast – a node without a ledger-recording privilege records and broadcasts it to other nodes.
– fraudulent record – a node records and broadcasts a fraudulent transaction.

## 7.4 Tamper-resistance

Tamper-resistance is an important feature of the DLT that is favoured by the industry. Once data is loaded into a shared ledger, it requires extensive computational resources or massive collusion among nodes to modify it unnoticed by others, hence rendering it practically immutable. There is no difference whether the change is intentional or accidental, malicious or benign.

TAM-001: A distributed ledger system is required to be tamper-resistant. Once the transaction record enters the shared ledger, it is hard to change it. If it does change, the process of implementing the change will inevitably be recorded and easily found, unless all nodes agree.

TAM-002: A distributed ledger system is required to implement mechanisms to authenticate the addresses of a transaction.

TAM-003: A distributed ledger system is required to have mechanisms to authenticate each node in the network.

TAM-004: A distributed ledger system is required to implement algorithms to protect the integrity of the transactions and the ledger distribution messages.

TAM-005: A distributed ledger system is required to implement mechanisms to ensure that data stored in the shared ledger cannot be repudiated.

## 7.5 Auditability

By sharing the data in the shared ledger, DLT facilitates audit processing. However, the supports of data privacy in public DLT and private DLT are different. A public DLT allows all data in a shared ledger to be audited by anyone, whereas a private DLT only allows those who are authorized to access a transaction. All transaction-related entities are authorized by default. To conform to regulations on data privacy makes audit processing more complicated, a public DLT has to either save information off-chain or save information in the shared ledger, but encrypted.

AUD-001: A distributed ledger system is required to allow all transaction-related entities to access their transaction information stored in a shared ledger.

AUD-002: A distributed ledger system is required to have all transaction records in the ledger signed by the entity or entities that initiate(s) the transaction.

AUD-003: A distributed ledger system is required to store information that can help audit processing in the log managed by nodes and the enable the provision of such a log to the auditor when it is needed.

## 7.6 Fairness

Fairness between nodes is crucial for achieving effective multi-party supervision and ensuring that data cannot be falsified. The roles of nodes in the DLT network may vary due to differences in bandwidth, storage, computing power or shares possessed in the DLT network. For example, some nodes may only have verification capabilities, and others may record ledgers. A distributed ledger system is required to guarantee that the rules to assign rights are consistent and the possibility for a node to get a ledger-recording privilege is random. In other words, nodes of the same ability should have a statistically identical opportunity in the network. A distributed ledger system must avoid giving too many ledger-recording rights to a node or a group of nodes. Such unfair treatment can easily lead to security problems.

FAI-001: A distributed ledger system is required to fairly assign rights to nodes, i.e., nodes with similar capabilities are given similar rights. Rights in the DLT network include governance participation, ledger recording and transaction verification.

FAI-002: A distributed ledger system is recommended to choose the ledger recording node, e.g., randomly or in turns, from all nodes with a ledger-recording privilege.

FAI-003: A distributed ledger system is required to support node management. Node management determines:

– which server can join the DLT network as a node;

– the rights of the node in the network.

FAI-004: A distributed ledger system is required to support either permissioned or permissionless DLT deployment.

FAI-005: A distributed ledger system is recommended to support nodes that cannot cooperate with the consensus, but can store the shared ledger and verify transactions.

FAI-006: A distributed ledger system is recommended to support the configuration of permissioned nodes to participate in the consensus algorithm and permissionless nodes to receive and verify transactions without participating in the consensus algorithm.

FAI-007: A distributed ledger system can optionally assign special rights only to nodes that have certain properties, e.g., a minimum number of tokens.

## 7.7 Stability

The distributed nature of the DLT makes it face stability challenges. Each node in a DLT may have the capability to record transactions, but which node is eventually selected is a result of the consensus algorithm. There are many factors that affect the stability of a DLT network, e.g., malicious nodes, network attacks, network failures, multipaths in networks, system upgrades or patches and governance disputes. The design of a distributed ledger system is required to take these factors into consideration and enables the distributed ledger to regain stable service through adaptive processing of the system.

However, special interventional procedures are sometimes required, which often result in soft or hard forks. In some cases, it can be solved by a soft fork, i.e., a gradual upgrade of all nodes. At this point, an upgraded node can be compatible with a non-upgraded node on the same ledger.

A hard fork is more serious situation, all nodes need to be upgraded synchronously to make the system resume stable operation. At this point, a node that has not been upgraded can no longer process the ledger information sent by an upgraded node. Note that due to the decentralization of nodes, it is almost impossible to upgrade them all simultaneously after the distributed ledger system has formed a certain scale. After the agreed upgrade time, there will be two parallel shared ledgers in the DLT network, one maintained by the upgraded nodes and another maintained by the un-upgraded nodes, which means that the shared ledger is forked.

STA-001: A distributed ledger system is required to adopt a consensus algorithm. In this Recommendation, consensus is attained when all nodes in the network make unanimous agreement on a transaction, which records who sends assets to whom.

STA-002: A distributed ledger system is required to run stably on heterogeneous networks.

STA-003: A distributed ledger system is required to reach a correct consensus with high probability in limited time.

STA-004: A distributed ledger system is required to reach a correct consensus even if there are malfunctioning or malicious nodes, providing that the number of malfunctioning nodes does not exceed the consensus threshold, which is the limitation on the number of malfunctioning or malicious nodes that the consensus algorithm can tolerate. The consensus threshold is determined mainly by the consensus algorithm. For example, the consensus threshold of proof of work (PoW) is 50% in many implementations. The consensus threshold is an important performance index of a distributed ledger system.

STA-005: A distributed ledger system is required to be able to resolve a fork created due to the propagation of different transactions in the network. As more than one node may produce valid proofs almost simultaneously, and nodes producing the proofs may have different transaction records, multiple versions of ledgers will be generated in the distributed ledger system. Resolution of such an inconsistency is recommended by accepting one of the records and eliminating others according to predefined rules (e.g., the longest fork rule).

STA-006: A distributed ledger system is required to defend against double spending, which is a failure mode of the DLT when it is possible to spend a single digital token twice. For example, an attacker who owns 10 tokens wants to pay the same 10 tokens to victims A and B. First, the attacker pays 10 tokens to A, which is recorded in the ledger and admitted by other nodes. Second, the attacker makes a transaction to B of the same 10 tokens that had been paid to A. Unlike physical token money, such as coins, electronic files can be duplicated, and hence the act of spending a digital coin does not remove its data from the ownership of the original holder. For this double spending, the network would have the ability to reject the second transaction and the second transaction is not recorded into the ledger.

STA-007: A distributed ledger system is required to prevent Sybil attacks and dismiss masquerading hostile entities. A Sybil attack in DLT is one in which consensus processing is subverted by forging identities in P2P networks, such as an attacker disguised as:

– a legitimate node to send a dishonest transaction;
– multiple nodes to break consensus in the distributed ledger system.

STA-008: For a permissionless distributed ledger system, the system is required to deploy an incentive mechanism, which allows a selected node to record that a shared ledger has been rewarded by an amount of tokens. The incentive encourages nodes to take active part in the algorithm and stabilizes the networks. Transaction fees are another type of incentive mechanism. For each transaction, a small amount of transaction fees is given to the network and shared by the rest of nodes.

STA-009: A distributed ledger system is required to automatically recover from a malicious node attack, network attack, node failure or network failure to smooth operation.

STA-010: A distributed ledger system is required to have the ability to initiate soft forks or hard forks. The system is required to have the ability to upgrade all nodes to the new system at the agreed time to initiate hard forks.

NOTE – An upgrade of a distributed ledger node can only be done by its administrator. The centralized upgrade service of the traditional centralized system is not suitable for the distributed ledger system.

### 7.8 Governance

Given that a distributed ledger system is inherently distributed, with multiple nodes typically owned and operated by different owners, the requirements of usual governance are mainly about how to govern the system as a whole and keep the system executing the tasks.

GOV-001: A distributed ledger system is recommended to adopt appropriate governance methods to ensure the continuity of its lifecycle.

GOV-002: A distributed ledger system is required to adopt appropriate governance methods to ensure that the system is able to resolve disputes in an appropriate manner.

GOV-003: A distributed ledger system is required to have its rules of network governance be open and clear. If it is necessary to amend the governance rules, their revision process should be ensured to be in accordance with the original agreement.

GOV-004: A distributed ledger system is required to adopt appropriate governance practices to ensure that governance supports decentralization, tamper resistance, fairness and stability.

GOV-005: A distributed ledger system is recommended to adopt appropriate governance methods to support changes in the platform.

## 8 Advanced requirements for distributed ledger systems

This clause defines advanced requirements for distributed ledger systems. These advanced functional requirements often make the design of the application more convenient and secure. However, because of the tighter coupling with the application, it is possible that these advanced requirements are not supported by all systems.

### 8.1 Access control

Access control is a requirement for DLT in most commercial enterprise services and in financial scenarios. Access control involves the rights to read the data in the shared ledger and to create new data.

Implementing access control on a public DLT by encryption is not recommended. Storing private information off-chain and storing only hashes on-chain can be a way to appropriately handle private data on a public DLT. A private DLT is able to carry a more comprehensive access control by utilizing capabilities such as entity ID, blacklist or whitelist.

ACC-001: A distributed ledger system can optionally protect the accessing of records in a shared ledger in some use cases.

ACC-002: A distributed ledger system is required to use privacy enhancing technologies for the handling of private data. For example, if a social ID is stored in a shared ledger, storage of the ID as plaintext should be avoided.

ACC-003: A distributed ledger system can optionally store only the integrity checksum (of the data) in the shared ledger and implement access control off the shared ledger.

ACC-004: A distributed ledger system can optionally support the limitation of data access from the time perspective.

ACC-005: If access control is implemented in the system, the system can optionally record all successful or failed data accesses.

ACC-006: A distributed ledger system is recommended to be able to control who can create data in the shared ledger.

## 8.2 Consensus

A consensus algorithm is a set of rules that precisely defines a sequence of operations to reach consensus. Lots of functional or performance parameters are determined by consensus algorithms, including the fault tolerance threshold, the maximum number of transactions processed per second, the ledger recording period, and the transaction confirmation delay.

CON-001: The distributed ledger system is required to determine a threshold of the number of nodes that need to agree in order to consider a transaction valid in the whole network. After a transaction is considered valid, all remaining nodes need to agree that the transaction is valid in order to continue with the same view of the shared ledger.

CON-002: A distributed ledger system is required to support at least one type of crash fault tolerant consensus algorithm.

CON-003: A distributed ledger system is recommended to support at least one type of Byzantine consensus algorithm.

CON-004: A distributed ledger system is recommended to support the user selection of the consensus algorithm.

CON-005: A distributed ledger system is recommended to have its consensus algorithm able to defend against denial of service (DoS) or distributed denial of service (DDoS) attacks.

CON-006: A distributed ledger system is recommended to have its consensus algorithm able to defend against eclipse attacks, wherein an adversary controls a sufficient number of IP addresses to monopolize all connections between victim nodes and the DLT network, and exploits the victim nodes to launch other attacks, such as double spending.

CON-007: If mining is supported, the distributed ledger system is recommended to have its consensus algorithm able to defend against selfish mining, wherein a group of colluding nodes hide newly generated records instead of broadcasting them instantly, and publish the records strategically. Selfish mining enables nodes to win more rewards than deserved, so rational nodes will prefer to join the colluding group. In this way, the colluding group will increase in size until it reaches the fault tolerance threshold.

CON-008: A distributed ledger system is recommended to support changes in the consensus algorithm configuration. For example, how difficult it is to solve a puzzle in a PoW consensus algorithm.

CON-009: A distributed ledger system is recommended to offer a quantum-resistant consensus algorithm.

## 8.3 Data storage

Because of the tamper resistant nature of DLT, many applications store its data on ledger. The data may be in various formats, such as images, audio, video or text. On-ledger data occupies *N* times the storage than the off-ledger case, where *N* is the number of copies of the data on the ledger.

A data digest can be used to save ledger storage space while keep the data tamper resistant. When using a data digest, the hash digest of the data is stored on ledger, where the copy of the data is stored off ledger. One-way IDs can be used to point to data stored off ledger, to facilitate data access.

DS-001: A distributed ledger system is recommended to support the storage of multiple data types, such as a video or audio clips, the name of the entity or the audit log for the transaction.

DS-002: A distributed ledger system is recommended to store the hash digest of a file, and to store the file off ledger.

DS-003: To avoid leakage of information, if one-way IDs are used to access off ledger files, the distributed ledger system is required to make the one-way ID independent of the data content.

DS-004: A distributed ledger system can optionally store the transactions in blocks. Each block may contain a header with metadata and a body with a set of transactions.

## 8.4 Identity

A distributed ledger system promotes the protection and utilization of identity, and achieves the protection of identity information in terms of confidentiality, integrity and availability. Achieving these protections requires a combination of the following factors:

– a combination of technical and non-technical means;

– data processing across DLT and non-DLT systems;

– security and encryption processing requirements;

– access management through API and storage architecture.

A distributed ledger system is the basis for decentralized identification. Trust is no longer rooted in any single entity, but is based on information sharing and mutual supervision in the network. This trust model allows non-trusted stakeholders to conclude transactions in the system, and the use of encryption and signatures helps to verify the provenance and hence facilitates audits. Decentralized identification can be applied to natural people, legal entities, things and processes.

IDE-001: A distributed ledger system can optionally ensure that any certification information is recorded and stored in a shared ledger. The certification information includes, but is not limited to, copyright information, insurance information and enterprise qualification.

IDE-002: A distributed ledger system can optionally provide a mechanism to validate the authenticity of certification information. Any certification information must be validated as authentic before being recorded and stored in a shared ledger.

IDE-003: A distributed ledger system can optionally provide an access mechanism to ensure that any person or organization that wants to query the certification is authorized by the information owner.

## 8.5 Interoperability

Multiple DLT systems may run in parallel in the future, although the original DLT designer may wish to serve all the requirements by a single DLT system. If multiple DLTs are inevitable, how do these DLT systems interoperate?

Interoperable DLT opens up a comprehensive service that moves assets from one platform to another, supporting atomic exchange across DLTs, making it easy to access information in other DLT networks (e.g., a link of "identity DLT" with a payment DLT network). Interoperability can also be achieved by a third party without any additional effort by the original DLT protocol.

Interoperable DLT can be used in several use cases. Examples follow.

– Transfer of assets across distributed ledger systems. For example, asset A is safely transferred from the parent ledger X system to the Y system, which can go through a series of transactions in the Y system, e.g., using the asset as collateral. The entire process should ensure that the relevant assets in the Y system are supported for back transfer to the mother ledger X system.

– Atomic swap across distributed ledger system. Atomic swap consists of two asset exchanges and needs to ensure that they either succeed or fail. The atomic swap across the distributed ledger system allows user X to transfer digital asset A to user Y in exchange for Y to transfer digital asset B to user X. (A and B are maintained by different DLT systems. Both X and Y have accounts in A and B.) Cross-chain atomic exchange provides secure transaction processing for mutually untrusted parties.

– Oracle across distributed ledger system. For example, a smart contract running on an X system will only perform certain operations if it receives valid evidence from the oracle address on another Y system.
– Cross-DLT asset collateral: locks asset A on the X system, while the lock condition depends on the activity on the Y system. Many scenarios in a multi-DLT environment will require such cross-chain processing capabilities, such as liens, collateral for financial derivatives, bankruptcy recovery, court orders, and various use cases involving margin.
– Smart contracts across distributed ledger system. For example, paying dividends to holders of assets registered on the Y system in the X system.

INT-001: A distributed ledger system can optionally support interoperability with other distributed ledger systems.

INT-002: A distributed ledger system can optionally support the transfer of assets to another distributed ledger system.

INT-003: A distributed ledger system can optionally support atomic exchanges across distributed ledger systems.

INT-004: A distributed ledger system can optionally support oracle across distributed ledger systems.

INT-005: A distributed ledger system can optionally support cross-DLT asset collateral.

INT-006: A distributed ledger system can optionally support smart contracts across DLT.

## 8.6 Membership

A distributed ledger system is a distributed network, and the public DLT treats the separation of the private key from the identity as a method to protect user privacy. Therefore, the public chain seldom provides user management, i.e., does not support the query of a user's address, and cannot verify the public key a user provided. However, an application exists that supports the disclosure the owner of some addresses. Such information is mainly collected from public information, e.g., the holder proactively claiming to hold certain addresses, or some financing activities or transactions reveal the ownership of some addresses.

However, membership management is a requirement for most private or permissioned distributed ledger systems. The membership service can be implemented either distributed or centralized, i.e., on or off ledger, respectively. No matter how the membership service is implemented, the membership service means that important identification information is stored in the system; hence security and privacy protection is required.

MEM-001: A public distributed ledger system can optionally have applications to serve the query of user/node information. The applications have to collect user or node information using public methods.

MEM-002: A private or permissioned distributed ledger system is required to build a membership management service, to manage either the user information or the node information.

MEM-003: If membership management is supported, the distributed ledger system is required to adopt appropriate methods to protect the security and privacy of the member's identification data.

## 8.7 Openness

Early distributed ledger systems were public DLTs, which benefit data disclosure and audit processing with open data access. In most enterprise services, government services, and financial services, the private DLTs are used. A viewing account needed to be authorized for audit by others.

OPE-001: A public distributed ledger system is required to provide open access to the transaction records to anyone.

OPE-002: A private distributed ledger system is required to limit the transaction records read access to the (predefined) particular group.

OPE-003: To improve the stability and security, a permissioned distributed ledger system is recommended to deploy a node monitoring mechanism to monitor the bandwidth, CPU, storage, etc.

OPE-004: A permissioned distributed ledger system can optionally implement censoring mechanisms, such as:

– removal of a specific node from the distributed ledger system, i.e., so that the system neither processes messages received from nor sends messages to the node;

– locking of a specific address so that it cannot make any new transactions;

– monitoring of all transactions of a specific address by dedicated censors.

OPE-005: A permissionless distributed ledger system can optionally implement some censoring mechanisms. One example is a decentralized decision about a forward transaction coming from specific nodes to other P2P nodes.

### 8.8 Privacy protection

For the public DLT and the private DLT, privacy protection has different meanings and implementations.

For public DLT, anyone can access the transaction record in the ledger, although the transaction record itself is a very important private message. In order to protect privacy, most public distributed ledger systems anonymously process transaction records, i.e., the transaction record contains only signatures and addresses, and the address is not bound to a person's identity. Research has proved that this privacy protection method is not reliable, and a tracker can gradually confirm the holder of the transaction address through various means, such as address tracing and network tracing. To address this problem, many privacy transaction techniques have been introduced into public DLT, such as zero-knowledge proof, homomorphic encryption, ring signature and CoinJoin.

For private DLT, privacy protection is mainly to prevent unauthorized access. Although the privacy protection methods in the public distributed ledger systems in the previous paragraph can still be used, having access control to protect privacy is a more direct and reliable solution. Besides the transaction records, there may have been user identification information in the ledger, the distributed ledger system is required to provide effective protection for this identification information.

PRI-001: A distributed ledger system is recommended to protect the accessing of sensitive information on the shared ledger. Only those allowed by the owner of the data can access it.

PRI-002: A distributed ledger system can optionally adopt change addresses schemes, to protect the privacy of an entity in transaction. With changed addresses, a new key pair (and thus a new address) can be used for each transaction.

PRI-003: A distributed ledger system can optionally implement zero-knowledge proof mechanisms to protect privacy. Zero-knowledge proof is a method by which one party (the prover) can demonstrate to another party (the verifier) that a given statement is true, without conveying any information apart from the fact that the statement is indeed true.

PRI-004: A distributed ledger system can optionally implement ring signature mechanisms to protect privacy. A ring signature is a type of digital signature that can be performed by any member of a group of users that have keys.

PRI-005: A distributed ledger system can optionally implement CoinJoin mechanisms [b-Zheng] to protect privacy. CoinJoin makes a joint transaction instead of sending out several single transactions, i.e., when one wants to make a payment, find someone else who also wants to make a payment and make a joint transaction together.

PRI-006: A distributed ledger system can optionally implement homomorphic encryption mechanisms to protect privacy. Homomorphic encryption is a form of encryption that allows computations to be carried out on cipher text, thus generating an encrypted result which, when decrypted, matches the result of operations performed on the plaintext.

PRI-007: For a private distributed ledger system, the system is required to provide effective protection for any identification information stored on a ledger.

### 8.9 Provenance

Provenance is evidence of the origin and sequence of transactions about assets of interest. It is the ability to track the entire lifecycle of an asset from its creation. Different distributed ledger systems have different requirements for provenance. On the one hand, the privacy trading system keeps provenance confidential; on the other, the supply chain management system wants all transactions related to the asset to be maintained in the tamper resistant ledger, enabling it to be traced back to the source of any asset, and to prove the source of the asset.

PRO-001: A distributed ledger system can optionally support provenance, i.e., support track the entire lifecycle of an asset from its creation.

### 8.10 Smart contract

A smart contract is a software program written on the distributed ledger system which encodes the rules for specific types of distributed ledger system transactions in a way that can be validated, and triggered by specific conditions. Based on smart contracts, different decentralized applications have been built. While support for smart contracts is an important feature of the distributed ledger system, there are different conceptions and implementations of them:

– Global addressed. Each smart contract is identified by an address on the distributed ledger system. The application calls the function defined by the smart contract by sending a transaction to the smart contract address. The execution results are stored on the system. In this case, each smart contract address has a creator or owner. All nodes with ledger record or verification rights execute smart contracts. Since the ledger record itself requires a lot of computational processing, smart contracts need to avoid large storage requirements and avoid complex processing.

– Node addressed. The installation of programs on dedicated nodes that can be executed by addressing function calls of these programs using the node address. Since the execution of smart contracts occurs primarily at dedicated nodes, it can use almost any resource on the local node.

SC-001: A distributed ledger system can optionally support smart contracts and decentralized applications.

SC-002: If smart contracts are supported, the distributed ledger system is required to provide mechanisms to automate the lifecycle of smart contracts.

SC-003: If smart contracts are supported, the distributed ledger system is recommended to provide mechanisms to automate the definition of different roles to interact with the smart contracts.

SC-004: If smart contracts are supported, the distributed ledger system is required to provide mechanisms to protect against attacks that take advantage of its smart contract capability.

SC-005: If smart contracts are supported, the distributed ledger system can optionally support the running of a decentralized application over it.

SC-006: If smart contracts are supported, the distributed ledger system can optionally support the running of multiple applications over it concurrently.

SC-007: If smart contracts are supported, the distributed ledger system can optionally support the storage of data of an overlay application in the shared ledger.

SC-008: To avoid incorrect automatic processing, a distributed ledger system can optionally have the validation of a transaction to check whether the address of the receiver is valid.

SC-009: If global-addressed smart contracts are supported, the distributed ledger system is required to deploy mechanisms that protect the resources on ledger being abused. The resources may be the storage space on ledger or the computational power of the nodes, etc.

SC-010: If global-addressed smart contracts are supported, the distributed ledger system can optionally store the global state data.

SC-011: If global-addressed smart contracts are supported, the distributed ledger system is required to adopt mechanisms to limit the number and time of execution of smart contracts to guarantee the system stability.

SC-012: If global-addressed smart contracts are supported, the distributed ledger system is required to adopt mechanisms to direct data-intense applications to store and retrieve their data off-ledger to prevent a DoS attack.

SC-013: If smart contracts are supported, the distributed ledger system can optionally support the possibility of using multiple programming languages.

SC-014: If smart contracts are supported, the distributed ledger system can optionally support a Turing-complete programming language.

SC-015: If smart contracts are supported, the distributed ledger system can optionally support the possibility of emitting events to sign specific states.

SC-016: If smart contracts are supported, the distributed ledger system can optionally support a link between the programming language code and a natural human language.

SC-017: If smart contracts are supported, the distributed ledger system can optionally support a mechanism to help to upgrade the contracts used in an application.

### 8.11 Token

A token is a digital representation of value on a shared distributed ledger that is owned and secured using cryptography to ensure its authenticity and prevent modification or tampering without the owner's consent. Smart contracts provide an autonomous solution for supporting the issuance, trading, and destruction of tokens, although they can also be recorded directly in ledger transaction records. The technology to support tokens through smart contracts has experienced rapid improvements. Initially only able to issue fungible tokens, now extended to support tokens such as non-fungible and security. A non-fungible token corresponds to a unique item that is physically indivisible, such as a gem. The security token needs to meet related regulatory requirements for a securities register and trading in various countries. No matter how the tokens are implemented, on a distributed ledger system, tokens are associated with private keys. There are some terms with similar meanings to token in the DLT industry, such as crypto asset, cryptocurrency and virtual coin.

TOK-001: A distributed ledger system is recommended to support the use of tokens that are associated with private keys or smart contracts.

TOK-002: If the use of a token is supported, the distributed ledger system is required to support token issuance, token transfer, token withdrawal, token settlement and clearance, token balance enquiry.

TOK-003: If the use of a token is supported, the distributed ledger system is required to protect the security of token processing, such as token issuance, token transfer, token withdrawal, token settlement and clearance.

TOK-004: If the use of a token is supported, the distributed ledger system is recommended to ensure the appropriate level of privacy of token owners.

TOK-005: If the use of a token is supported, the distributed ledger system is recommended to support the function of the token and its value generation method.

## 8.12 Cipher suites

Cipher suites are used in many functions in a DLT system, including addressing generation, consensus algorithm and data storage. Cipher suites include at least asymmetric cryptography and a hash function.

CIP-001: A distributed ledger system is required to have the ability to provide cipher suites with appropriate security level, e.g., quantum-resistant, to ensure the functionalities and privacy of the platform.

CIP-002: A distributed ledger system is recommended to support the user selection of cipher suites.

CIP-003: If smart contracts are supported, a distributed ledger system is recommended to offer native support of asymmetric function and a hash function in the programming languages used to develop these smart contracts.

## 8.13 Utilities

To make a distributed ledger system easy to use for system developers, maintainers and users, there are various utilities requirements, such as complex transaction support, blockchain browsers, wallets, network monitoring tools, node self-test tools, smart contract debugging tools and test networks.

EOU-001: A distributed ledger system is recommended to provide a shared ledger browser that allows users to query transaction details for a specified transaction or address, or to track the origin of a given asset.

EOU-002: If used for asset management, a distributed ledger system is recommended to provide a wallet application to facilitate user access to account balances, perform transfers, and track the progress of operations.

EOU-003: A distributed ledger system can optionally support one to multiple transactions. For example, a transaction records that entity A sends *k*, *l*, *m* tokens, respectively, to entities B, C, D.

EOU-004: A distributed ledger system can optionally support multiple to multiple transactions. For example, a transaction records that entity A and B spends *k* and *l* tokens respectively, and entity C, D, E earns *m*, *n*, *o* tokens, respectively. Note that in this example, $k + l$ must equal $m + n + o$.

EOU-005: A distributed ledger system can optionally provide an application interface (API) to third party developers.

EOU-006: If APIs are provided, the distributed ledger system is required to ensure the security and privacy of API processing.

EOU-007: A distributed ledger system can optionally monitor the performance of nodes. Abnormal nodes will be handled in time to maintain the stability and security of the system.

EOU-008: A distributed ledger system can optionally monitor network status, such as the number of pending transactions, the number of active nodes, node addresses, geographic distribution, bandwidth and processing capabilities. For DLT using the PoW consensus, the total hashing power of the network and the change in the hashing power can optionally be also monitored.

EOU-009: A distributed ledger system can optionally monitor usage, such as the total number of transactions, total transaction amount, and total data records, in a given time period.

EOU-010: A distributed ledger system can optionally monitor user volume such as the number of new users per day, the number of active users per day, the number of independent address visits per day, and changes in the foregoing information.

EOU-011: A distributed ledger system can optionally provide accessibility to system upgrades. Since the DLT is based on a distributed network, the upgrade of the system on each node must be performed by the node administrator. The DLT system can construct a message channel for all node administrators, through which the system update notification is pushed.

EOU-012: A distributed ledger system can optionally provide the test tools and test networks to facilitate application development based on the system. Although some of the systems are open source, the appropriate test tools and test networks can effectively reduce the developer's time and effort.

# Bibliography


[b-ITU-T X.800] Recommendation ITU-T X.800 (1991), *Security architecture for Open Systems Interconnection for CCITT applications*.

[b-ITU-T X.1161] Recommendation ITU-T X.1161 (2008), *Framework for secure peer-to-peer communications*.

[b-ITU-T X.1215] Recommendation ITU-T X.1161 (2019), *Use cases for structured threat information expression*.

[b-ITU-T Y.2060] Recommendation ITU-T Y.4000/Y.2060 (2012), *Overview of the Internet of things*.

[b-ITU-T TS FG DLT D1.1] Technical Specification ITU-T FG DLT D1.1 (2019), *Distributed ledger technology terms and definitions*.

[b-ISO 22739] ISO 22739:2020, *Blockchain and distributed ledger technologies – Vocabulary*.

[b-Eugster] Eugster, P. T., Guerraoui, R., Handurukande, S. B., Kouznetsov, P., Kermarrec, A. M. (2003). Lightweight probabilistic broadcast. ACM T. Comput. Syst., **21**(4), pp. 341-374. DOI: 10.1145/945506.945507

[b-Gramoli] Gramoli, V., Staples, M. (2018). Blockchain standard: Can we reach consensus? *IEEE Commun. Standards Mag.* **2**, pp. 16-21. DOI: 10.1109/MCOMSTD.2018.1800022.

[b-Natarajan] Natarajan, H., Krause, S., Gradstein, H. (2017). *Distributed ledger technology (DLT) and blockchain*, FinTech Note; No. 1. Washington, DC: World Bank. 60 pp. Available [viewed 2020-10-06] at: https://openknowledge.worldbank.org/bitstream/handle/10986/29053/WP-PUBLIC-Distributed-Ledger-Technology-and-Blockchain-Fintech-Notes.pdf?sequence=1&isAllowed=y

[b-Zheng] Zheng, Z., Xie, S., Dai, H., Chen, X., Wang, H. (2017). An overview of blockchain technology: Architecture, consensus, and future trends. In: *2017 IEEE International Congress on Big Data (BigData Congress)*, pp. 557-564. New York, NY: IEEE.

## SERIES OF ITU-T RECOMMENDATIONS

| | |
|---|---|
| Series A | Organization of the work of ITU-T |
| Series D | Tariff and accounting principles and international telecommunication/ICT economic and policy issues |
| Series E | Overall network operation, telephone service, service operation and human factors |
| **Series F** | **Non-telephone telecommunication services** |
| Series G | Transmission systems and media, digital systems and networks |
| Series H | Audiovisual and multimedia systems |
| Series I | Integrated services digital network |
| Series J | Cable networks and transmission of television, sound programme and other multimedia signals |
| Series K | Protection against interference |
| Series L | Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant |
| Series M | Telecommunication management, including TMN and network maintenance |
| Series N | Maintenance: international sound programme and television transmission circuits |
| Series O | Specifications of measuring equipment |
| Series P | Telephone transmission quality, telephone installations, local line networks |
| Series Q | Switching and signalling, and associated measurements and tests |
| Series R | Telegraph transmission |
| Series S | Telegraph services terminal equipment |
| Series T | Terminals for telematic services |
| Series U | Telegraph switching |
| Series V | Data communication over the telephone network |
| Series X | Data networks, open system communications and security |
| Series Y | Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities |
| Series Z | Languages and general software aspects for telecommunication systems |

Printed in Switzerland
Geneva, 2020