|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ITU logo | INTERNATIONAL TELECOMMUNICATION UNION  **TELECOMMUNICATION STANDARDIZATION SECTOR**  STUDY PERIOD 2017-2020 | | | FG-AI4H-F-025-A05 | |
| **ITU-T Focus Group on AI for Health** | |
| **Original: English** | |
| **WG(s):** | | | Plenary |  | |
| **DOCUMENT** | | | | | |
| **Source:** | | | FG-DLT | | |
| **Title:** | | | Updated text of D3.3 - Assessment criteria for DLT platforms | | |
| **Purpose:** | | | Discussion | | |
| **Contact:** | | Qing Sude CAICT  China | | | WeChat ID: qingsd Tel: +86188 0002 7056 E-mail:qingsude@caict.ac.cn |
| **Contact:** | | Wei Kai CAICT China | | | WeChat ID: winkee\_wk Tel: +8601062300249 E-mail: weikai@caict.ac.cn |
| **Contact:** | | Sun Hao  Institute of Digital Money The People's Bank of China | | | Tel: +86 10 59051628 Email：szhbyw@pbc.gov.cn |

|  |  |
| --- | --- |
| **Keywords:** | Assessment criteria; DLT platforms |
| **Abstract:** | This document contains the updated text of ITU-T FG DLT deliverable D3.3 “Assessment criteria for DLT platforms” reflecting the discussion at the Madrid meeting. |

**ITU-T FG DLT Deliverable D3.3**

**Assessment criteria for DLT platforms**

**Table of Contents**

Page

[1 Scope 6](#_Toc24387)

[2 Reference 6](#_Toc13440)

[3 Definition 6](#_Toc12847)

[4 Abbreviation and acronym 6](#_Toc3361)

[5 Overview 7](#_Toc23483)

[6 Criteria for DLT core functions 8](#_Toc32590)

[6.1 Account creation 8](#_Toc7522)

[6.2 Transaction processing 8](#_Toc17390)

[6.3 Query 8](#_Toc15457)

[6.3.1 Balance query 8](#_Toc886)

[6.3.2 Conditional query 8](#_Toc32471)

[6.4 Consensus mechanism effectiveness 8](#_Toc13570)

[6.4.1 Data consistency 8](#_Toc2826)

[6.4.2 BFT (Byzantine Fault Tolerance) / CFT (Crash Fault Tolerance) 9](#_Toc2827)

[6.5 Private key management 9](#_Toc424)

[6.5.1 Software-based method 9](#_Toc800)

[6.5.2 Hardware-based method 9](#_Toc13118)

[6.6 Smart contract validity 9](#_Toc4668)

[6.6.1 Monitorability of participants’ status 9](#_Toc21027)

[6.6.2 Lifecycle management of smart contract 9](#_Toc20028)

[6.6.3 Security of smart contract 9](#_Toc4150)

[6.6.4 Smart contract data access control 9](#_Toc10858)

[6.6.5 Usability of smart contract 10](#_Toc29331)

[6.7 Security of cryptography 10](#_Toc28733)

[6.7.1 Encryption declaration 10](#_Toc25984)

[6.7.2 Pluggable encryption algorithm. 10](#_Toc10465)

[6.7.3 Efficiency of encryption algorithm. 10](#_Toc27510)

[6.7.4 Strength of encryption 10](#_Toc23293)

[7 Criteria for DLT application functions 10](#_Toc3652)

[7.1 User authentication 10](#_Toc14279)

[7.1.1 User account verification 10](#_Toc22120)

[7.1.2 Login state management 10](#_Toc14712)

[7.1.3 User classification and user management 10](#_Toc25117)

[7.1.4 Authorization 10](#_Toc1696)

[7.1.5 Smart contract data access control 11](#_Toc16878)

[7.2 System stability 11](#_Toc18673)

[7.2.1 Stability for manage nodes 11](#_Toc11983)

[7.2.2 Stability for cross-chain operation 11](#_Toc23410)

[7.2.3 Network latency 11](#_Toc3592)

[7.2.4 Memory utilization 11](#_Toc4921)

[7.2.5 CPU utilization 11](#_Toc6522)

[7.2.6 Stability for concurrency 11](#_Toc16166)

[7.3 Economic mechanism design 11](#_Toc30469)

[7.3.1 Incentive mechanisms 11](#_Toc7726)

[7.3.2 Token management 11](#_Toc8212)

[7.3.3 Token transfer 11](#_Toc18738)

[7.4 Data privacy 12](#_Toc17051)

[7.4.1 Secure transmission 12](#_Toc13442)

[7.4.2 Restricted data 12](#_Toc20245)

[7.4.3 Privacy protection 12](#_Toc20085)

[7.5 Application support functions 12](#_Toc17207)

[7.5.1 User interface for query 12](#_Toc11198)

[7.5.2 User interface for smart contract 12](#_Toc32628)

[7.5.3 Multi-languages SDK 12](#_Toc30562)

[8 Criteria for DLT operation functions 12](#_Toc9618)

[8.1 Network management 12](#_Toc19334)

[8.1.1 Node status monitoring. 12](#_Toc12499)

[8.1.2 Multi type nodes 12](#_Toc353)

[8.1.3 Node configuration modify 13](#_Toc15493)

[8.2 Risk management and mitigation 13](#_Toc14878)

[8.2.1 Recovery mechanisms 13](#_Toc21495)

[8.2.2 Trouble shooting 13](#_Toc12146)

[8.3 Data Archiving 13](#_Toc16158)

[8.3.1 Data archiving 13](#_Toc4653)

[8.3.2 Data query 13](#_Toc11796)

[8.3.3 Data recovery 13](#_Toc11507)

[9 Performance 13](#_Toc1473)

[9.1 Calculation Method 13](#_Toc6340)

[9.2 Preconditions for test 14](#_Toc28261)

[9.2.1 Test environment 14](#_Toc19638)

[9.2.2 Topology of the network 14](#_Toc12948)

[9.2.3 Test system deployment 14](#_Toc25399)

[9.3 Transaction 14](#_Toc15498)

[9.4 Test Tools 14](#_Toc30986)

[9.5 Maximum transaction time 14](#_Toc2958)

[9.6 Average transaction time 14](#_Toc22442)

[10 Criteria for DLT ecosystem 14](#_Toc5457)

[10.1 Platform maturity 14](#_Toc12181)

[10.2 Availability of professionals 14](#_Toc31626)

[10.3 Running cost 15](#_Toc18969)

[Bibliography 16](#_Toc32533)

**ITU-T FG DLT** **deliverable D3.3**

**Assessment criteria for DLT platforms**

Summary

This document is a deliverable of the ITU-T Focus Group on Application of Distributed Ledger Technologies (FG DLT). It defines an assessment framework for a distributed ledger technology (DLT) platform, which includes a set of criteria for function, performance and other aspects. The framework can be used as a guideline for DLT platform assessment as well as information disclosure of a certain DLT platform product.

Keywords

Distributed Ledger Technologies, blockchain, assessment criteria

Introduction

# Scope

This document defines an assessment framework for a distributed ledger technology (DLT) platform, which includes a set of criteria for function, performance and other aspects. The framework can be used as a guideline for DLT platform assessment as well as information disclosure of a certain DLT platform product.

# References

The following ITU-T Recommendations are referred to the text in such a way that some or all of their content constitutes provisions of this document.

*[to be added]*

ITU-T FG DLT deliverable D1.1

ITU-T FG DLT deliverable D3.1

ISO/IEC 25000-25099

# Definitions

This document applies the terms and definitions in ITU-T FG DLT deliverable D1.1.

Client

Dapp

Wallet

Address

Node

# Abbreviation and acronym

This Document uses the following abbreviations:

API Application Programming Interface

CPU Central Processing Unit

DDoS Distributed Deny of Service

DLT Distributed Ledger Technology

KYC Know Your Customer

NoSQL Not Only SQL

SDK Software Development Kit

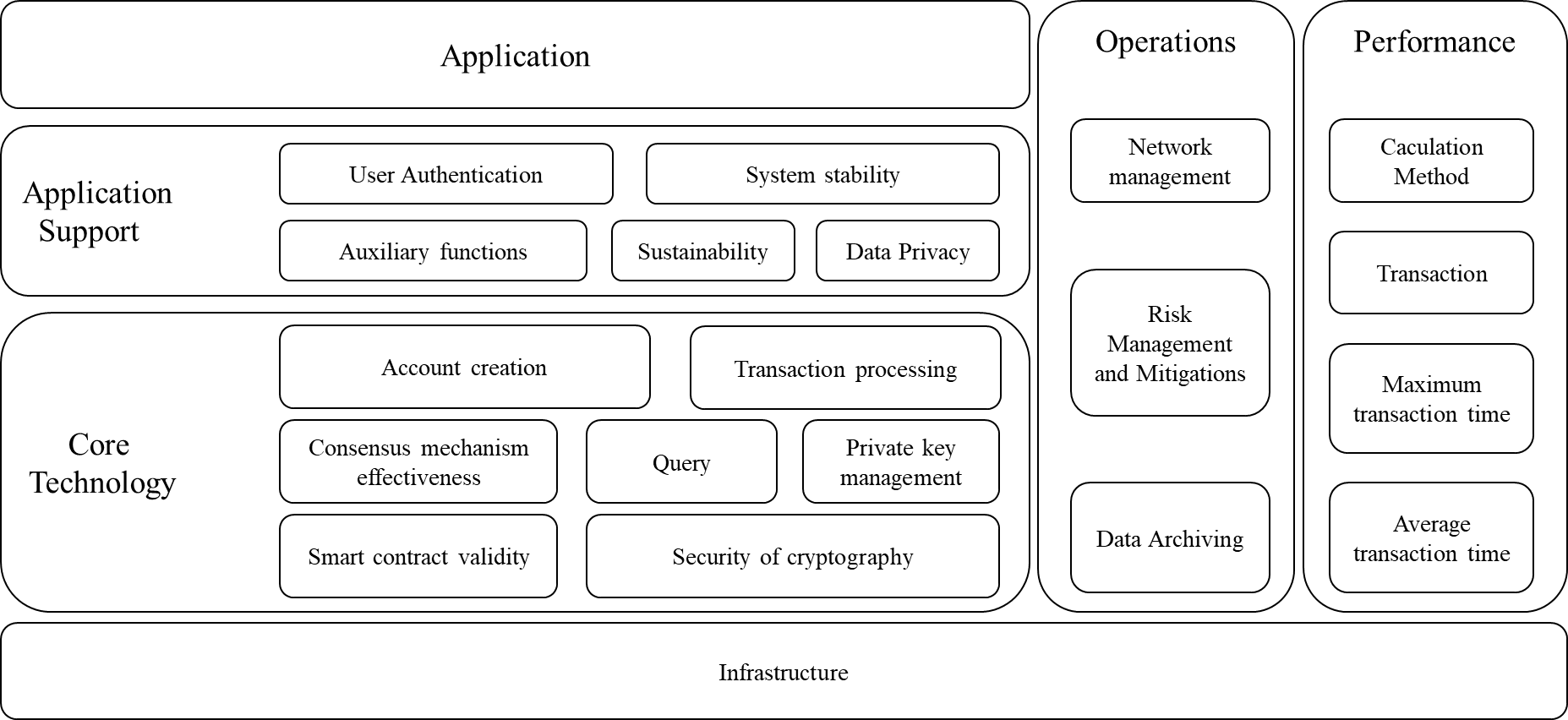
SQL Structured Query Language

SPV Simplified Payment Vеrifiсаtiоn

TPS Transaction Per Second

# Overview

The assessment criteria framework defined in this document consists of 15 assessment items, which can be classified into 3 domains including core functions, application support functions and operation functions of a distributed ledger technology (DLT) platform, this framework also has a performance domain, see Figure 1.



**Figure 1 Overview of the DLT assessment framework**

The framework corresponds to the high-level architecture of distributed ledger technology, see Figure 2.



**Figure 2 High Level Conceptual Architecture of DLT**

The framework corresponds to the architecture reference of a distributed ledger technology (DLT) platform, see Figure 3.



**Figure 3 DLT architecture diagram**

The vendors are required to reveal details about these metrics, which can be verified through document review or functional testing.

# Criteria for DLT core functions

## Account creation

This item stands for the ability to create user accounts. Account contains public and private key pairs. Create action can be launched by client or smart contract, private key shall be generated by client only. If an account name could be customized by user, it needs to make sure that the account name is unique in the system.

## Transaction processing

This item stands for the ability to process transaction(s). There are two types of transactions,

* 1. Asset transfer transaction which refers to the transfer of certain amount of asset among accounts, ensuring the asset in ledger is balancing and
  2. Non-asset transfer transaction, such as changing configuration parameter of an account, modifying status of a smart contract and other status modification operation within an account, without any asset being transferred.

Users can query the correct result from any of the nodes in DLT system after a transaction is successfully committed.

## Query

User(s) can get result(s) by information request(s).

### 6.3.1 Balance query

User can acquire its account balance with a searching condition.

### 6.3.2 Conditional query

User can search its historical information in the DLT platform with a searching condition such as a time period specification or specified user account.

## Consensus mechanism effectiveness

A consensus mechanism is a fault-tolerant mechanism that is used in DLT platform to achieve the necessary agreement with a single data value or a single state of the network among distributed processes or multi-agent systems. An insecure consensus mechanism may lead to a disastrous result so that it should be checked whether there are enough nodes to ensure consensus mechanism effectiveness and that data stored in the underlying database is accuracy and consistency.

### 6.4.1 Data consistency

The data synchronization module ensures that the distributed ledger has a consistent ledger. The data synchronization module transmits a new kind of ledger between different nodes. The synchronization module also validates the synchronized data to ensure the correctness and consistency of the synchronized data in an eventually consistent or strong consistent way, depending on the specified consensus mechanism.

### 6.4.2 BFT (Byzantine Fault Tolerance) / CFT (Crash Fault Tolerance)

DLT system should work as well as an indefinite number of nodes that take malicious actions or crash. The amount of malicious/crashed node is determined by the consensus mechanism selected.

## Private key management

Private key management is an important function for user experience. It provides a reliable and safe way to keep users’ private keys. There are two common methods to manage private keys: software-based method and hardware-based method.

### 6.5.1 Software-based method

It could be a digital app or online service to store the private key and could be used to track ownership. In a deterministic wallet, it should be checked that mnemonic sentence or word seed could be generated and the single root key is safe enough. In a non-deterministic wallet, each key can be randomly generated on its own accord. Therefore, any backups of the wallet must store each and every single private key used as an address.

### 6.5.2 Hardware-based method

A hardware-based wallet should be a device or physical medium which stores the private key. It should demonstrate basic operations such as receiving and sending cryptocurrencies.

## Smart contract validity

With the development of blockchain technology, blockchain has been extended from simple accounting transactions to supporting complex transactions which are stored in blockchain network in the form of bytecode programs, as an extension of distributed ledger module. According to the blockchain terminology, the bytecode programs are called smart contracts.

To support smart contract, smart contract mechanism carries specific services on the blockchain, including the language definition, compilation, and execution of the code. Smart contracts for different block chains can be implemented using simple interpreted scripts or fully functional virtual machines.

### 6.6.1 Monitorability of participants’ status

This item stands for all status of smart contract participants should be monitored.

### 6.6.2 Lifecycle management of smart contract

This item stands for contract should have state ID for its lifecycle, such as create, deploy, active, suspend and destroy.

### 6.6.3 Security of smart contract

This item stands for high quality-less bug by using special programming language, providing contract templates.

### 6.6.4 Smart contract data access control

The DLT system should disclose how developers specify the authorization and confidentiality of their smart contract in a technical view.

### 6.6.5 Usability of smart contract

Users or potential users of the smart contracts can have an easy way to understand the smart contract governance model, especially who owns what (e.g. user data, cryptocurrency assets) and who has access to what, for example, via visual interfaces to represent smart contracts and query them.

## Security of cryptography

The DLT system should use secure, strong enough encryption.

### 6.7.1 Encryption declaration

The system should specify where the encryption derives from, open-source encryption or regulatory compliance encryption.

### 6.7.2 Pluggable encryption algorithm.

The system can use pluggable modular encryption and switch to a specified encryption algorithm online or offline as required.

### 6.7.3 Efficiency of encryption algorithm.

It should run several encryption settings to process different sizes of data blocks to evaluate the algorithm's encryption/decryption and signature/verification speed.

### 6.7.4 Strength of encryption

It should declare that how difficult the encryption can be broken. Category and cipher strength of the encryption could be taken as metrics. Besides, anti-quantum encryption algorithm could be taken into account.

# Criteria for DLT application functions

## User authentication

DLT system should have user authentication modules and user access control management modules. Electronic signatures are an effective way to user authentication.

### 7.1.1 User account verification

This is the validation of information, such as keystore and password, or 2-step verification.

### 7.1.2 Login state management

The system should update the user login-state after user login.

### 7.1.3 User classification and user management

This is to assign users into one of several types and manage their permissions.

### 7.1.4 Authorization

Users can grant authority to others to access or modify their private data.

### 7.1.5 Smart contract data access control

Smart contract data should be shared among participants of the smart contract, all participants should have access to get this information, for example, interfaces to represent smart contracts and query them.

## System stability

System stability is the measurement of overall system interoperability, accessibility and usability. The platform is required to satisfied at least, but not limited to the following requirements.

### 7.2.1 Stability for manage nodes

The system should be used normally when some nodes join, leave or upgrade.

### 7.2.2 Stability for cross-chain operation

The system should be used normally when cooperated with other DLT system or cloud System.

### 7.2.3 Network latency

The system should remain stable after running 7x24 hours with latency of network. The tolerable extent of latency based on design of system.

### 7.2.4 Memory utilization

The system should remain stable after running 7x24 hours without memory exceptions.

### 7.2.5 CPU utilization

The system should remain stable after running 7x24 hours without CPU exceptions.

### 7.2.6 Stability for concurrency

The system should remain stable with bursts of concurrent transactions.

## Economic mechanism design

Economic mechanism is to motivate user participation and how to manage token.

### 7.3.1 Incentive mechanisms

The rewarding in blockchain system, especially permission-less blockchain. System relies on nodes owned by different parties and an incentive should be in place to motivate them to continue serving and using the system. For example, Bitcoin Proof-of-Work (PoW) uses a lot of electricity and miners are rewarded through Bitcoins. If an incentive mechanism is in place, it should be detailed, e.g.

### 7.3.2 Token management

The duty of using a token. If tokens are used, a tokenomics report explaining the tokens economics should be publicly released. Any potential environmental issues should be highlighted in the report, e.g., the high electricity consumption of Bitcoin PoW.

### 7.3.3 Token transfer

System should use a standard protocol where possible for increased token transfer. For example, regarding its potential tokens, they should be based on de facto standards such as ERC-20[b\_erc20] or NEP-5[b\_nep5]. System should have functionalities to facilitate cross-chain or cross-DLT system operations.

## Data privacy

Data privacy is an assignable feature for DLT platform, especially in financial industry. It stands for data generation, storage and transmission in a DLT system should be confidential and user private information cannot be leaked.

### 7.4.1 Secure transmission

Confidential or proprietary information should be transferred over a secure channel. Many infrastructures, especially in financial industry, should rely on a specified secure transmission protocol to prevent a catastrophic breach of security.

### 7.4.2 Restricted data access

Any confidential or personal information that is protected by law or policy requires the appropriate level of differential access controls and security protections whether in storage or in transit.

### 7.4.3 Privacy protection

System may use privacy protection algorithm(s) such as zero knowledge proof, ring signature, secure multi-party computation and homomorphic encryption to avoid privacy disclosure.

## Application support functions

DLT system may implement application support functions to improve user-friendly.

### 7.5.1 User interface for query

DLT System may provide a browser to visualize a query result and show the status of ledger.

### 7.5.2 User interface for smart contract

DLT System may provide a browser to visualize deployment and invocation of smart contract, and query to smart contract data.

### 7.5.3 Multi-languages SDK

System may provide SDK for multi-language version.

# Criteria for DLT operation functions

## Network management

The management and monitoring nodes, including status, configuration, type and behaviour.

### 8.1.1 Node status monitoring.

System should monitor nodes status, such as number of nodes online/offline, synchronization status, client version and so on.

### 8.1.2 Multi type nodes

The system should support node classification. For instance, node can be classified into two categories, full node and lightweight node, according to whether a complete ledger copy is stored in the node. According to a lightweight node, SPV method should be applied to verify the correctness of the shared ledgers.

### 8.1.3 Node configuration modify

The system should support hot or cold modification of the node’s configuration parameter, such as the consensus algorithm type, block size, node type, generate speed of block and so on.

## Risk management and mitigation

System should have strategy(ies) to resist DDoS attack, Sybil attack or dishonest node(s). If failed under attack, DLT system should have strategies to recover.

### 8.2.1 Recovery mechanisms

The DLT system should be able to recover from failure by downgraded recovery, security service, etc. Recovery solution should be flexible and problem-oriented.

### 8.2.2 Trouble shooting

The DLT system should have a trouble alarm, rapid trouble shooting, automated failure notification.

### 8.2.3 Avoid single point of failure

The DLT system shall not depend on any centralized system which might cause a single point of failure.

## Data Archiving

The DLT system is an append-only trusted data set. However, mass data stored in DLT system may degrade the query performance so that it is essential to move data that is no longer actively used to a separate storage device for long-term retention, especially for a permissioned DLT system.

### 8.3.1 Data archiving

Data no longer used or whose activity level is below a threshold may be transferred to independent storage. The DLT system should have a specified mechanism to migrate the active data from a node.

### 8.3.2 Data query

Data may be queried by tools such as SQL rules, or APIs after archiving.

### 8.3.3 Data recovery

Archived data should be recovered in some ways, and the ledger(s) should stay the same relative to the status before archiving.

# Performance

The throughput and resource usage of processing standard transactions. Environment and deployment reasons may affect performance, such as network topology and test environment (CPU, memory, disk, network). Transactions Per Second (TPS) is a standard performance indicator. When evaluating performance with TPS, topology deployment and test environment must be indicated.

## 9.1 Calculation Method

The data used to calculate TPS needs to be gathered when system is stable. Both system booting or shutting down and test environment instability may led to inaccuracy.

## 9.2 Preconditions for test

### 9.2.1 Test environment

When evaluating performance with TPS, the hardware such as CPU, memory, hard disk, network must be indicated.

### 9.2.2 Topology of the network

When evaluating performance with TPS, node topology must be indicated.

### 9.2.3 Test system deployment

When evaluating performance with TPS, the type of deployment must be indicated.

## 9.3 Transaction

When evaluating performance with TPS, the type of transaction must be indicated.

## 9.4 Test Tools

There are open source tools designed especially for blockchain system to test performance, such as Hyperleder Caliper[b\_caliper] or TrustedBench[b\_trustedb]. Script can also be used to make performance test.

## 9.5 Maximum transaction time

The longest waiting time from a transaction submission to confirmation. This item must give a certain number, if time is out and transaction is not finished, transaction must be destroyed and return failure.

## 9.6 Average transaction time

The average waiting time from a transaction submission to confirmation. This item is about performance and do not must give a certain number. When evaluating the DLT system with this indicator, test environment, deployment of system should be considered.

# Criteria for DLT ecosystem

## Platform maturity

Platform maturity of DLT includes many factors such as year of creation, launch of production version, deploying network (mainnet/testnet), numbers of production networks, numbers of applications. Whether permissioned network or permissionless network, DLT vendor should disclose this information to all consumers.

## Availability of professionals

Different DLT platform refers to diverse technical knowledge background, indirectly determining the number of professionals available in the market. Talent introduction, personnel training and team construction may generate a good talent market future and existing talent statistic depends on major factors, such as programming language, communication protocols, virtual machine, technical architecture, algorithm model and minor factors, such as IDE type, SDK, work location and environment and so on.

## Running cost

It is an essential factor for DLT ecosystem to calculate the running cost so as to tell whether applied one DLT system to share data and transfer value. It refers to transaction fee, transaction confirm time and cost of smart contract’s writing and reading. In future, with the importance of smart contract security, audit fee of smart contract and code review and many other intermediate fee should be taken into consideration.

## Avoid vendor lock-in

DLT system should have unified APIs, such as, service access, data dictionary, communication protocol, encryption algorithm and system testing, so that there can be multiple vendors that provide similar services for the customers to avoid vendor lock-in.

# Bibliography

[b-D3.1] ITU-T FG-DLT Base document D3.1 “Distributed Ledger Technology Architecture”

[b-D3.3] ITU-T FG-DLT Base document D3.3 “Assessment criteria for DLT platforms”

[b-D1.1] ITU-T FG-DLT Base document D1.1 “Distributed ledger technology terms and definitions”

[b- SQuaRE] ISO/IEC 25000:2014, “Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE”

[b-Y.2091] Recommendation ITU-T Y.2091:2011, “Terms and definitions for next generation networks”

[b\_caliper] <https://cn.hyperledger.org/projects/caliper>

[b\_trustedb] <https://github.com/TrustedBlockchain/TrustedBench>

[b\_erc20] <https://theethereum.wiki/w/index.php/ERC20_Token_Standard>

[b\_nep5] <https://neoguide.io/nep-5-tokens-2/>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_