

International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Y.3530

(09/2020)

SERIES Y: GLOBAL INFORMATION
INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS,
NEXT-GENERATION NETWORKS, INTERNET OF
THINGS AND SMART CITIES

Cloud Computing

**Cloud computing – Functional requirements for
blockchain as a service**

Recommendation ITU-T Y.3530



ITU-T Y-SERIES RECOMMENDATIONS

GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS, NEXT-GENERATION NETWORKS, INTERNET OF THINGS AND SMART CITIES

GLOBAL INFORMATION INFRASTRUCTURE	
General	Y.100–Y.199
Services, applications and middleware	Y.200–Y.299
Network aspects	Y.300–Y.399
Interfaces and protocols	Y.400–Y.499
Numbering, addressing and naming	Y.500–Y.599
Operation, administration and maintenance	Y.600–Y.699
Security	Y.700–Y.799
Performances	Y.800–Y.899
INTERNET PROTOCOL ASPECTS	
General	Y.1000–Y.1099
Services and applications	Y.1100–Y.1199
Architecture, access, network capabilities and resource management	Y.1200–Y.1299
Transport	Y.1300–Y.1399
Interworking	Y.1400–Y.1499
Quality of service and network performance	Y.1500–Y.1599
Signalling	Y.1600–Y.1699
Operation, administration and maintenance	Y.1700–Y.1799
Charging	Y.1800–Y.1899
IPTV over NGN	Y.1900–Y.1999
NEXT GENERATION NETWORKS	
Frameworks and functional architecture models	Y.2000–Y.2099
Quality of Service and performance	Y.2100–Y.2199
Service aspects: Service capabilities and service architecture	Y.2200–Y.2249
Service aspects: Interoperability of services and networks in NGN	Y.2250–Y.2299
Enhancements to NGN	Y.2300–Y.2399
Network management	Y.2400–Y.2499
Network control architectures and protocols	Y.2500–Y.2599
Packet-based Networks	Y.2600–Y.2699
Security	Y.2700–Y.2799
Generalized mobility	Y.2800–Y.2899
Carrier grade open environment	Y.2900–Y.2999
FUTURE NETWORKS	Y.3000–Y.3499
CLOUD COMPUTING	Y.3500–Y.3599
BIG DATA	Y.3600–Y.3799
QUANTUM KEY DISTRIBUTION NETWORKS	Y.3800–Y.3999
INTERNET OF THINGS AND SMART CITIES AND COMMUNITIES	
General	Y.4000–Y.4049
Definitions and terminologies	Y.4050–Y.4099
Requirements and use cases	Y.4100–Y.4249
Infrastructure, connectivity and networks	Y.4250–Y.4399
Frameworks, architectures and protocols	Y.4400–Y.4549
Services, applications, computation and data processing	Y.4550–Y.4699
Management, control and performance	Y.4700–Y.4799
Identification and security	Y.4800–Y.4899
Evaluation and assessment	Y.4900–Y.4999

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

Recommendation ITU-T Y.3530

Cloud computing – Functional requirements for blockchain as a service

Summary

Blockchain as a service (BaaS) is a cloud service category in which the capabilities provided to the cloud service customer are the ability of setting up blockchain platform, and development decentralized application using blockchain technologies. In BaaS, an integrated developing environment (IDE) for cloud service customers (CSCs) is provided to create, deploy and operate decentralized applications. Recommendation ITU-T Y.3530 introduces blockchain and blockchain as a service. Recommendation ITU-T Y.3530 also provides functional requirements of blockchain as a service which are derived from use cases.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T Y.3530	2020-09-29	13	11.1002/1000/14404

Keywords

Blockchain, BaaS, blockchain as a service, cloud computing.

* 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 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/>.

© ITU 2020

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.

Table of Contents

	Page
1	Scope 1
2	References..... 1
3	Definitions 1
3.1	Terms defined elsewhere 1
3.2	Terms defined in this Recommendation..... 3
4	Abbreviations and acronyms 3
5	Conventions 4
6	Overview of blockchain..... 4
6.1	Introduction to blockchain..... 4
6.2	Block and hash function 5
6.3	Consensus of blockchain 6
6.4	Transaction confidentiality 6
6.5	Blockchain network..... 7
6.6	Decentralized application 7
6.7	Smart contract..... 7
7	Blockchain as a service..... 7
7.1	Introduction to BaaS..... 7
7.2	System context of BaaS..... 7
8	Functional requirements of BaaS..... 9
8.1	Node configuration requirements 9
8.2	Operation and monitoring of blockchain platform requirements 10
8.3	Decentralized application development support requirements..... 11
8.4	Security requirements 12
9	Security considerations 12
Appendix I – Use cases and scenarios of blockchain as a service..... 13	
I.1	Utilization of BaaS for enterprise users..... 13
I.2	Developing DApp using blockchain as a service 14
I.3	Adding new block on the blockchain 15
I.4	Digital signature and controlling transaction 16
I.5	Communications among the nodes in multiple platforms 18
I.6	Developing and deploying DApp in multiple platforms with IDE 20
Appendix II – Use cases and scenarios of applications for blockchain as a service..... 22	
II.1	Creating smart contract in CSP:BaaS provider 22
II.2	Deployment and operation of a smart contract..... 23
Bibliography..... 25	

Recommendation ITU-T Y.3530

Cloud computing – Functional requirements for blockchain as a service

1 Scope

This Recommendation introduces blockchain as well as blockchain as a service (BaaS). This Recommendation also provides the system context of BaaS including sub-roles and activities in terms of cloud computing as well as functional requirements for blockchain as a service in a cloud computing environment. The functional requirements in this Recommendation are derived from use cases.

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.

[ITU-T Y.3500] Recommendation ITU-T Y.3500 (2014), *Information technology – Cloud computing – Overview and vocabulary*.

[ITU-T Y.3501] Recommendation ITU-T Y.3501 (2016), *Cloud computing – Framework and high-level requirements*.

[ITU-T Y.3502] Recommendation ITU-T Y.3502 (2014), *Information technology – Cloud computing – Reference architecture*.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 account [b-FG-DLT Terms]: Representation of an entity whose data is recorded on a distributed ledger.

3.1.2 activity [ITU Y.3502]: A specified pursuit or set of tasks.

3.1.3 block [b-FG-DLT Terms]: Party which is in a business relationship for the purpose of using cloud services.

3.1.4 blockchain [b-FG-DLT Terms]: A type of distributed ledger which 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.

3.1.5 cloud service customer [ITU-T Y.3500]: Party which is in a business relationship for the purpose of using cloud services.

3.1.6 cloud service provider [ITU-T Y.3500]: Party which makes cloud services available.

3.1.7 cloud service partner [ITU-T Y.3500]: Party which is engaged in support of, or auxiliary to, activities of either the cloud service provider or the cloud service customer, or both.

3.1.8 consensus [b-FG-DLT Terms]: Agreement that a set of transactions is valid.

3.1.9 consensus mechanism [b-FG-DLT Terms]: Rules and procedures by which consensus is reached.

3.1.10 cryptography [b-ITU-T X.800]: The discipline which embodies principles, means, and methods for the transformation of data in order to hide its information content, prevent its undetected modification and/or prevent its unauthorized use.

NOTE – Cryptography determines the methods used in encipherment and decipherment. An attack on a cryptographic principle, means, or method is cryptanalysis.

3.1.11 decentralized application [b-FG-DLT Terms]: Application that runs in a distributed and decentralized computing environment.

3.1.12 digital signature [b-FG-DLT Terms]: Data appended to, or a cryptographic transformation (see cryptography) 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.

NOTE – Consider the definition 'digital signature' as "Data appended to data units, or cryptographic changes made to data units, which allows the recipient of the data unit to confirm the origin and integrity of the data and protect the data from being forged."

3.1.13 distributed ledger [b-FG-DLT Terms]: A type of ledger that is shared, replicated, and synchronized in a distributed and decentralized manner.

3.1.14 fork [b-FG-DLT Terms]: Creation of two or more different versions of a distributed ledger.

NOTE – There are two types of forks. See clause 3.1.15 (hard fork) and clause 3.1.26 (soft fork).

3.1.15 hard fork [b-FG-DLT Terms]: Change to the protocol or rules that result in a fork that is not backward compatible.

3.1.16 hash function [b-FG-DLT Terms]: A function that maps a bit string of arbitrary length to a fixed-length bit string.

3.1.17 ledger [b-FG-DLT Terms]: Information store that keeps final and definitive (immutable) records of transactions.

3.1.18 node [b-FG-DLT Terms]: Device or process that participates in a distributed ledger network.

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

3.1.19 peer [b-ITU-T X.1161]: Communication node on P2P network that functions simultaneously as both "client" and "server" to the other nodes on the network.

3.1.20 peer-to-peer [b-ISO CD 22739]: Relating to, using, or being a network of peers that directly share information and resources with each other without relying on a central entity.

NOTE – In the context of a distributed ledger system, peers are nodes.

3.1.21 proof of work [b-FG-DLT Terms]: Consensus process to solve a difficult (costly, time-consuming) problem that produces a result that is easy for others to correctly verify.

NOTE – Producing a proof of work can be a random process with low probability so that a lot of trial and error is required on average before a valid proof of work is generated.

3.1.22 proof of stake [b-FG-DLT Terms]: Consensus process, where an existing stake in the distributed ledger system (e.g., the amount of that currency that you hold) is used to reach consensus.

3.1.23 public key cryptography [b-FG-DLT Terms]: Cryptography in which a public key and a corresponding private key are used for encryption and decryption, where public key is disseminated, and private key is known only to the key owner.

NOTE – Users can digitally sign data with their private key, and the resulting signature can be verified by anyone using the corresponding public key.

3.1.24 role [ITU Y.3502]: A set of activities that serves a common purpose.

3.1.25 smart contract [b-FG-DLT Terms]: 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.

3.1.26 soft fork [b-FG-DLT Terms]: Change to the protocol or rules that result in a fork that is backward compatible.

3.1.27 sub-role [ITU Y.3502]: A subset of the activities of a given role.

3.1.28 transaction [b-FG-DLT Terms]: Whole of the exchange of information between nodes. A transaction is uniquely identified by a transaction identifier.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 blockchain as a service (BaaS): A cloud service category in which the capabilities provided to the cloud service customer are the ability of setting up blockchain platforms, and developing decentralized applications using blockchain technologies.

NOTE – Blockchain technology includes consensus algorithms, smart contracts, cryptography, etc.

3.2.2 blockchain network: A network to implement a blockchain platform among nodes.

3.2.3 blockchain platform: Implementation of a blockchain to process data in a node.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

BaaS	Blockchain as a Service
CPU	Central Processing Unit
CSC	Cloud Service Customer
CSN	Cloud Service partner
CSP	Cloud Service Provider
DApp	Decentralized Application
DLT	Distributed Ledger Technology
GPU	Graphic Processor Unit
IDE	Integrated Developing Environment
PII	Personally Identifiable Information
P2P	Peer-to-Peer
PoS	Proof of Stake
PoW	Proof of Work
SDK	Software Development Kit
TCP	Transmission Control Protocol
TPS	Transactions per second
UDP	User Datagram Protocol
URI	Uniform Resource Identifier

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 document is to be claimed.

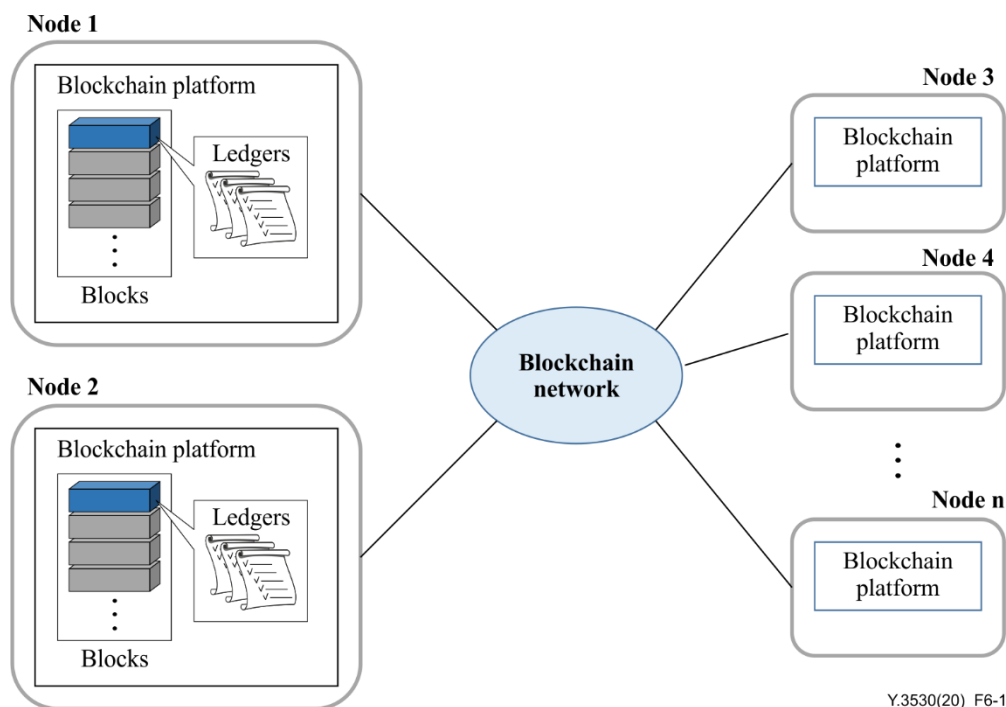
The keywords "is recommended" indicate a requirement which is recommended but which is not absolutely required. Thus, this requirement need not be present to claim conformance.

6 Overview of blockchain

6.1 Introduction to blockchain

Blockchain is a type of distributed ledger technology (DLT) which 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. A blockchain platform in a node creates a block containing ledgers which has final and definitive (immutable) records of transactions. The block is also shared, replicated, and synchronized in a distributed and decentralized manner among nodes.

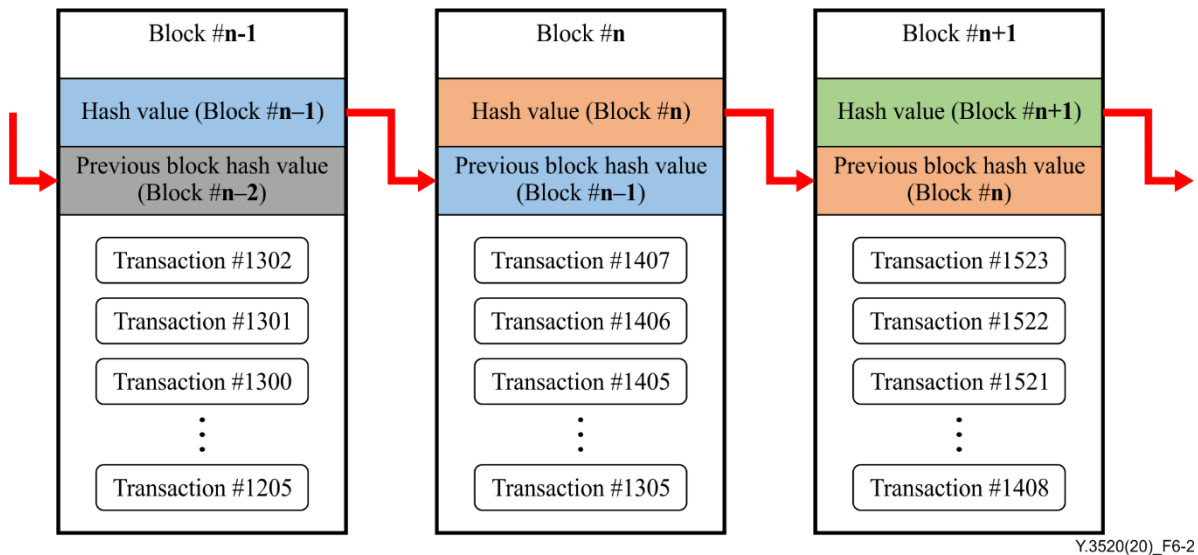
Figure 6-1 shows the concept of blockchain. The blockchain platform within a node is the implementation of the blockchain technology necessary for the operation and management of the blockchain. The blockchain platform supports verifying transactions, creating and distributing blocks, consensus algorithms, smart contracts, etc. The blockchain platform also provides cryptographic algorithms such as hash functions, and digital signatures.



Y.3530(20)_F6-1

Figure 6-1 – Concept of blockchain

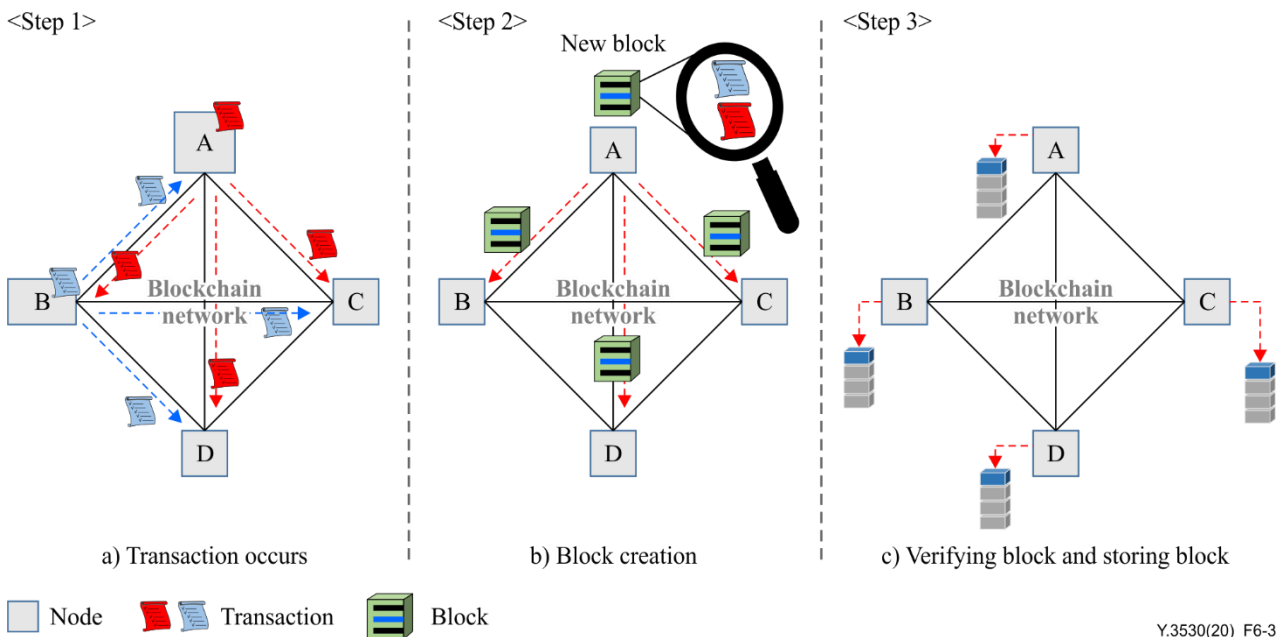
In blockchain, all blocks are chained through the hash value. As an example, Block #n contains the hash value of the previous Block #n-1 as shown in Figure 6-2. By linking the previous hash value to the block creation, it prevents changing or inserting a block between two existing blocks (Block #n-1 and Block #n, or Block #n and Block #n+1 in Figure 6-2).



Y.3520(20)_F6-2

Figure 6-2 – Chaining blocks with hash value

Figure 6-3 shows a process of creating and storing a block in blockchain network in detail. When a transaction occurs in any node, a blockchain platform in a node transmits the transaction to all other nodes. The blockchain platform in each node tries to create the block including the transactions. If a block is generated, the block is transmitted to all other nodes in the blockchain network, and the blockchain platform verifies the block. If there is no problem, the block is stored in each node's storage.



Y.3530(20)_F6-3

Figure 6-3 – Process of creating and storing a block in a blockchain network

6.2 Block and hash function

The block consists of block number, block hash, block header, transactions, other values (block size, version, etc.) and a hash tree as shown in Figure 6-4. In the blockchain, the root hash is included in the block header. In order to get the root hash, first, a hash value corresponding to the transaction is generated, and then two hashes are combined to get one upper hash, and two upper hashes are also combined to get a higher hash value again. If any transaction data is tampered with, the corresponding hash value change and all the hash values generated thereafter are also changed.

Therefore, it is possible to immediately check whether the data has been tampered with by comparing only the root hash of a block.

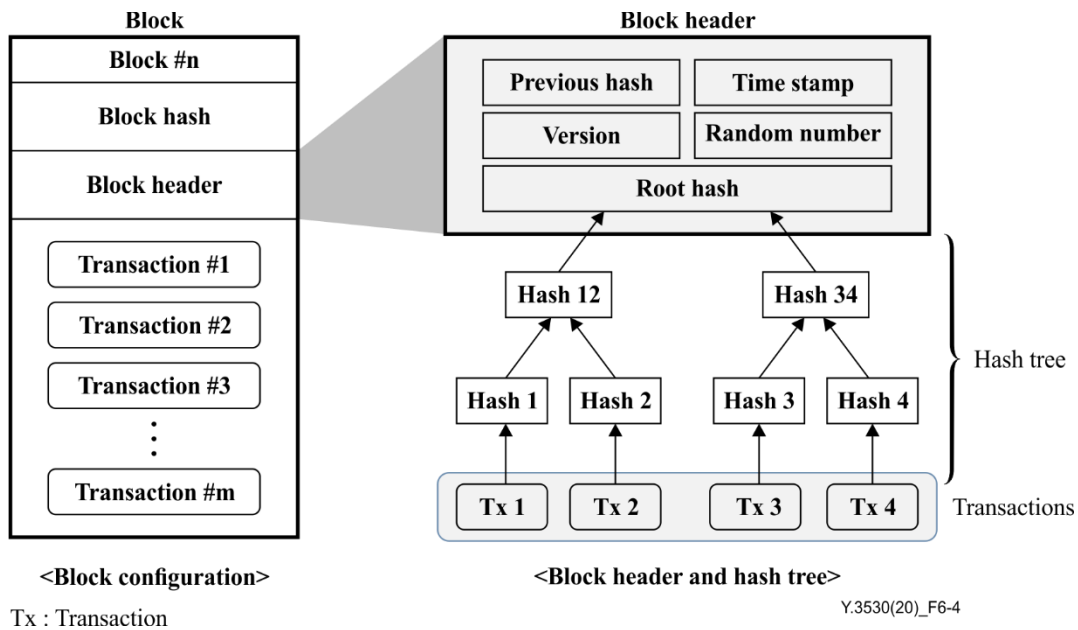


Figure 6-4 – Block diagram

NOTE 1 – Previous hash is the hash value of the previous block.

NOTE 2 – Root hash is a hash value of the root of the hash tree.

NOTE 3 – Timestamp is creation time of the block.

NOTE 4 – Version is number to track software of the blockchain platform.

NOTE 5 – Random number is a counter used for the proof of work algorithm.

NOTE 6 – The block header in Figure 6-4 is an example and can be changed according to consensus algorithms.

6.3 Consensus of blockchain

Consensus is an operation that verifies transactions and blocks. It ensures data consistency within the blockchain without being controlled by a specific central point. A consensus is achievable through the consensus algorithm in the blockchain. The typical consensus algorithm is proof of work (PoW), proof of stake (PoS), etc.

NOTE 1 – PoW is a consensus algorithm which is used to confirm transactions and create new blocks in a blockchain. Miners (nodes) compete against each other to complete transactions on the blockchain network.

NOTE 2 – PoS is a consensus algorithm that performs verification procedures. Each node generates blocks in proportion to the amount of stake held.

6.4 Transaction confidentiality

The blockchain platform ensures protection of transactions using cryptography algorithms such as symmetric encryption, asymmetric encryption and so on. In general, symmetric encryption is mainly applied to private blockchain networks and asymmetric encryption is mainly applied to public blockchain networks.

NOTE 1 – Symmetric encryption is a single key encryption algorithm. The same key can be used to encrypt and decrypt information at the same time. Symmetric encryption is mainly applied to private blockchain networks.

NOTE 2 – Asymmetric encryption is an encryption algorithm that requires a pair of different encryption keys, including a public key and a private key. Public keys are widely distributed, while private keys are kept secret. Using a private key, users can digitally sign transactions with their private key, and the resulting signature can be verified by anyone using the corresponding public key.

6.5 Blockchain network

In the blockchain, nodes communicate with each other and deliver blocks and transactions in the blockchain network. Data transfer between nodes (e.g., transactions and blocks broadcasting for data synchronization) is based on peer-to-peer connections.

The blockchain networks are categorized into two types which include public blockchain networks and private blockchain networks. In the public blockchain network, a blockchain platform is used without restrictions on network access and nodes that want to participate in the network can freely participate in the blockchain network and access data.

In the private blockchain network, a blockchain platform can be used to obtain permission from the authorization group. Nodes that want to participate in private blockchain networks need to get permission to participate in a group with authority, participate in the blockchain network, and access data.

6.6 Decentralized application

A decentralized application (DApp) is a digital application or program that runs in distributed and decentralized computing environments instead of a single computer. Also there is no single authority to execute changes and controls. A blockchain platform provides development environments for a DApp developer such as a programming language, development tools and so on. The blockchain platform also provides the access interface for the DApp developer to use distributed ledgers.

6.7 Smart contract

The smart contract is a computer program written on the distributed ledger which encodes the rules for specific types of distributed ledger. The program contains all the rules, conditions, expiry dates and other relevant information needed for its fulfilment. A smart contract is deployed using cryptographically signed transactions on the distributed ledger. A smart contract deployment is completed by storing the signed transactions in all nodes. The smart contract is executed automatically as part of a transaction by the blockchain platform in nodes in the blockchain network. The results of the execution of smart contracts are validated by a consensus algorithm and recorded in the block.

7 Blockchain as a service

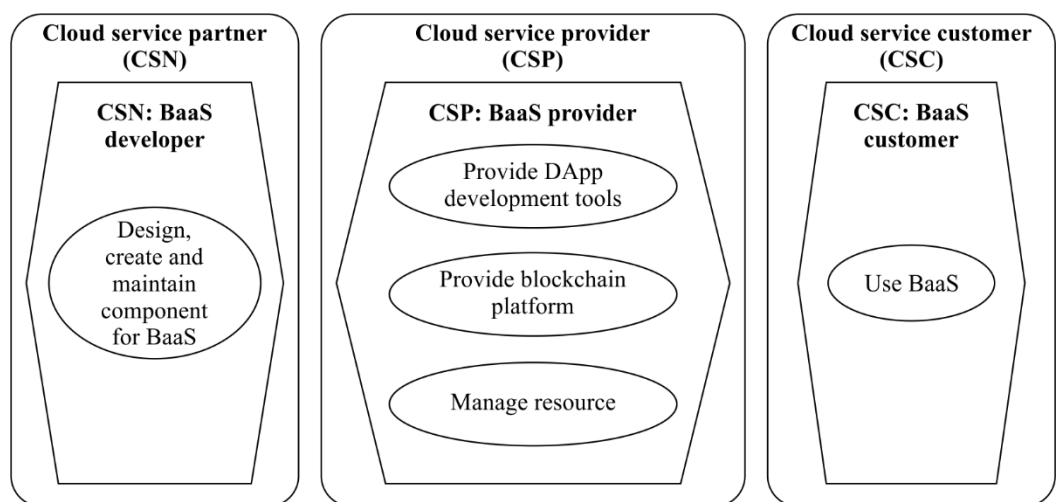
7.1 Introduction to BaaS

Blockchain as a service (BaaS) is a cloud service category in which the capabilities provided to the cloud service customer are the ability of setting up a blockchain platform, developing a decentralized application using blockchain technologies. A cloud service customer (CSC) creates and operates the blockchain network using BaaS, and the CSC develops DApps using development tools on the blockchain platform in BaaS.

7.2 System context of BaaS

The system context of BaaS provides additional sub-roles and activities based on the architectural user view defined in [ITU-T Y.3502]. Figure 7-1 illustrates the cloud computing sub-roles for blockchain. It identifies activities specific for blockchain and assigns them to cloud computing sub-roles. The system context of BaaS includes the following roles:

- CSN:BaaS developer;
- CSP:BaaS provider;
- CSC:BaaS customer.



Y.3530(20)_F7-1

Figure 7-1 – System context of blockchain as a service

7.2.1 CSN: BaaS developer

CSN:BaaS developer is the sub-role of a cloud service partner (CSN) which is engaged in support of, or auxiliary to, activities of a CSP:BaaS provider. The CSN:BaaS developer is responsible for developing blockchain as a service.

7.2.1.1 Design, create and maintain component for BaaS

This activity is responsible for developing the blockchain platform including blockchain technologies such as consensus algorithms, cryptographic capabilities in cloud computing environment, etc.

The design, create and maintain component for BaaS activity involves:

- designing and creating software components that are part of the implementation of a BaaS;
- creating the functionality which is offered to users of the BaaS, which also involves connecting the service components to the provider's operational support systems, so that the service implementation can be monitored and controlled;
- processing problem reports relating to the operation of a BaaS implementation;
- providing fixes to BaaS implementations;
- providing enhancements to BaaS implementations.

7.2.2 CSP:BaaS provider

CSP:BaaS provider is the sub-role of cloud service provider (CSP) providing the blockchain platform and development environment for decentralized applications. It also provides monitoring resources which are allocated in each node within the blockchain network.

7.2.2.1 Provide blockchain platform

The provide blockchain platform activity is responsible for the configuring and operation of the blockchain as a service. This activity involves:

- supporting interconnection and cross-ledger connection with other blockchain platforms;
- verifying transactions;
- distributing transactions;
- storing transactions and blocks;
- performing consensus algorithms among blockchain platforms;

- providing multiple consensus algorithms to satisfy the needs of CSC, including proof of work (POW), proof of stock (POS), and so on;
- establishing, maintaining and releasing connectivity among nodes;
- providing authentication of nodes;
- providing cryptography capabilities such as hash function, digital signature for verifying transactions and blocks;
- supporting multiple encryption algorithms such as symmetric encryption, asymmetric encryption, and so on.

NOTE – Symmetric encryption is a single key encryption algorithm. The same key can be used to encrypt and decrypt information at the same time. Asymmetric encryption is an encryption algorithm that requires a pair of different encryption keys, including the public key and private key. If the public key is used to encrypt the data, only the corresponding private key can be used to decrypt it.

7.2.2.2 Provide DApp development tools

This activity is responsible for providing an integrated developing environment (IDE). The provide DApp development tools activities involve:

- providing programming language;
- supporting syntax check errors for programming language;
- providing test suite;
- providing compiling and debugging DApp.

7.2.2.3 Manage resource

The manage resource activity is responsible for monitoring and alarming the status of resource utilization to the CSC. The manage resource activity involves:

- monitoring and alarming the status of nodes and their associated resources;
- monitoring and alarming the status of the blockchain network.

7.2.3 CSC:BaaS customer

CSC:BaaS customer is the sub-role of the cloud service customer performed by end users in order to use the BaaS from the CSP:BaaS provider.

7.2.3.1 Use BaaS

Use BaaS activity is responsible for configuring the blockchain platform and developing DApps (i.e. design, create and maintain decentralized applications) using BasS provided by the CSP:BaaS provider. The use BaaS activity involves:

- configuring the blockchain platform;
- developing the decentralized application.

8 Functional requirements of BaaS

8.1 Node configuration requirements

- 1) It is required that CSP:BaaS provider provides the available list of blockchain platforms.
- 2) It is required that CSP:BaaS provider provides one or more consensus algorithms.
NOTE 1 – Consensus algorithm includes proof-of-work, proof-of-stake, etc.
- 3) It is required that CSP:BaaS provider provides the configurations of a blockchain platform.

NOTE 2 – A blockchain platform includes public and private blockchains. A public blockchain has transaction records that are readable by anyone, but a private blockchain limits the transaction records read access to authorized groups.

- 4) It is required that CSP:BaaS provider provides the configuration of the hardware specifications for nodes.

NOTE 3 – Hardware specifications include memory size, graphic processor unit (GPU), central processing unit (CPU), storage, etc.

- 5) It is required that CSP:BaaS provider provides the configuration of block size in block.
- 6) It is required that CSP:BaaS provider provides the configuration of the number of nodes.
- 7) It is required that CSP:BaaS provider provides the configuration of the time interval of the block.
- 8) It is required that CSP:BaaS provider provides the configuration of consensus mechanism.

NOTE 4 – Configuration includes choosing a consensus algorithm and setting parameters such as the hash function.

- 9) It is recommended that CSP:BaaS provider provides recommendations of the blockchain platform for deploying DApp.

NOTE 5 – The recommendation of blockchain platform is searching and discovering the best blockchain platform with analysis of DApp performance to meet the CSP:BaaS customer's purpose.

- 10) It is required that CSP:BaaS provider provides deploying nodes on the blockchain network.

NOTE 6 – Deploying node is the participation of new nodes in a blockchain network. The participation includes the connecting to the blockchain platform in other nodes and downloading blocks to a new node.

- 11) It is recommended that CSP:BaaS provider provides language translation to convert smart contract codes for multiple blockchain platforms.

NOTE 7 – The language translation is a functionality to transform an input source code to a modified output code for the target blockchain platform.

- 12) It is recommended that the CSP:BaaS provider provides customized blockchain platforms for specific purposes including interoperability, anonymity, etc.

NOTE 8 – The interoperability of a blockchain is the ability to share information across different blockchain platforms.

NOTE 9 – The blockchain platforms perform different consensus algorithms and provide different languages to meet the specific purpose.

NOTE 10 – The blockchain platforms for interoperability support the network protocols for inter-blockchain communication. The blockchain platforms for anonymity support the untraceable transaction for privacy.

8.2 Operation and monitoring of blockchain platform requirements

- 1) It is required that CSP:BaaS provider provides software upgrades of blockchain platforms.

NOTE 1 – Software upgrade means soft fork (3.1.26) and hard fork (3.1.15).

- 2) It is required that CSP:BaaS provider provides monitoring the status of nodes.

NOTE 2 – The status of node is on or off.

- 3) It is recommended that CSP:BaaS provider provides status of blockchain platforms.

NOTE 3 – The status of blockchain platforms includes information of consensus algorithms, cryptographic algorithms, hash functions and so on.

- 4) It is recommended that CSP:BaaS provider provides lists of nodes in blockchain networks.

- 5) It is recommended that CSP:BaaS provider provides block information of nodes.

NOTE 4 – Block information includes block size, lasted block number, account, node IP address, etc.

- 6) It is required that CSP:BaaS provider provides monitoring status of resources utilization on each node.
- 7) It is required that CSP:BaaS provider provides configuration of threshold of transactions per second (TPS).

NOTE 5 – Transactions per second (TPS) refers to the number of transactions that a blockchain network is capable of processing each second.

- 8) It is required that CSP:BaaS provider provides alerts for the exceeding of a TPS threshold to CSC:BaaS customer.

8.3 Decentralized application development support requirements

- 1) It is required that CSP:BaaS provider provides development tools for developing DApp.

NOTE 1 –The development tool includes syntax check for errors for programing language, a tool for compiling and debugging DApp, etc.

- 2) It is required that CSP:BaaS provider provides one or more programming languages for developing DApp.

- 3) It is required that CSP:BaaS provider provides the deployment of smart contract codes.

NOTE 2 – Deployment of the smart contract is storing the block in all nodes.

NOTE 3 – A smart contract code is the set of instructions which is executed automatically as part of a transaction by blockchain platform in nodes.

- 4) It is required that CSP:BaaS provider provides a validation of syntax for smart contract languages.

- 5) It is required that CSP:BaaS provider provides test suites of smart contract code.

NOTE 4 – A test suite is a collection of test cases that are intended to be used to test a software program to show that it has some specified set of behaviours.

NOTE 5 – CSP:BaaS provider can prevent abnormal termination or infinite loop that occurs when executing a smart contract code using test suites.

NOTE 6 – Abnormal termination is unsuccessful completion of execution in the execution of a computer program because of an undesirable event, such as an operator error.

- 6) It is recommended that CSP:BaaS provider provides a blockchain test network for validating a smart contract or DApp.

NOTE 7 – The blockchain test network is the environment to test DApp prior to deploying it in a blockchain network.

NOTE 8 – The blockchain test network imitates the blockchain network.

- 7) It is required that CSP:BaaS provider provides security audits to find security vulnerabilities in smart contract codes.

NOTE 9 – Vulnerabilities includes reentrancy, timestamp dependence, transaction-ordering dependence, etc.

- 8) It is required that CSP:BaaS provider provides links such as uniform resource identifiers (URIs) to access smart contract codes.

- 9) It is recommended that the CSP:BaaS provider provides the predefined smart contract codes for supporting inter-blockchain communication.

NOTE 10 – The predefined smart contract codes are the software library of a smart contract.

NOTE 11 – The example of the predefined smart contract codes for supporting inter-blockchain communication is locking of the transaction between different blockchain platforms until the transaction is validated in both platforms.

8.4 Security requirements

- 1) It is recommended that CSP:BaaS provider provides public key cryptography for transaction confidentiality.
- 2) It is required that CSP:BaaS provider verifies the digital signature on transactions in blockchain networks.
- 3) It is recommended that CSP:BaaS provider provides an access right of transaction.
NOTE 1 – According to the access right of transaction, the access to a transaction is operated differently to CSC.
- 4) It is required that CSP:BaaS provider provides security policy for transactions.
NOTE 2 – CSP:BaaS provider classifies data according to the security level which is defined according to security policy.
- 5) It is recommended that CSP:BaaS provider provides verifying of PII in transactions before creating a block.
- 6) It is recommended that CSP:BaaS provider provides a protection mechanism for a PII in transactions.
NOTE 3 – When a PII is included in a transaction, CSP:BaaS provider decides if there will be encrypting of the transaction or not based on the security policy.
- 7) It is required that CSP:BaaS provider provides multiple level transaction encrypting according to the security level.
NOTE 4 – If the security level of the transaction is different, the transaction performs multi-encryption according to the specified security level and stores it in a block.

9 Security considerations

Security aspects for consideration within the cloud computing environment, especially for BaaS, are addressed by security challenges for the CSPs, as described in [b-ITU-T X.1601]. In particular, [b-ITU-T X.1601] analyses security threats and challenges, and describes security capabilities that could mitigate these threats and meet the security challenges.

[b-ITU-T X.1631] provides guidelines supporting the implementation of information security controls for cloud service customers and cloud service providers. Many of the guidelines guide the cloud service providers to assist the cloud service customers in implementing the controls, and guide the cloud service customers to implement such controls. Selection of appropriate information security controls, and the application of the implementation guidance provided, will depend on a risk assessment as well as any legal, contractual, regulatory or other cloud-sector specific information security requirements.

[b-ITU-T X.1401] mentions limitations and security threats which vary according to deployment model and service in distributed ledger technology (DLT). In brief, security is not only one motivation for applying DLT but also should be considered as an enhancement. [b-ITU-T X.1401] provides security analysis for developing, operating or using DLT and to support a security evaluation for the platform or service system based on DLT. It researches security capabilities achieved by DLT itself and also lists security threats to DLT. [b-ITU-T X.1402] describes security capabilities that could mitigate the related security threats and specifies a security framework methodology to determine how to use these security capabilities to mitigate security threats for a specific DLT system.

Appendix I

Use cases and scenarios of blockchain as a service

(This appendix does not form an integral part of this Recommendation.)

This appendix describes use cases and scenarios of blockchain as a service (BaaS):

- Clause I.1 describes utilization of BaaS for enterprises users.
- Clause I.2 describes developing DApp using blockchain as a service.
- Clause I.3 describes adding new blocks on the blockchain.
- Clause I.4 describes digital signature and controlling transaction.
- Clause I.5 describes communications among the nodes in multiple platforms.
- Clause I.6 describes developing and deploying DApp in multiple platforms with IDE.

I.1 Utilization of BaaS for enterprise users

Title	Utilization of BaaS for enterprise users
Description	<p>More and more enterprise users need to use blockchain technology to satisfy business needs in the fields of data security, evidence preservation, information sharing, etc. However, many enterprise users are not willing to develop a complete set of blockchain network from scratch, which is very costly. Even if enterprise users build a blockchain network based on the existing open source environment, the technical requirements will be quite high. One factor is that the configuration of software and hardware will be complicated. Another factor is the high cost of later maintenance and upgrading, and the lack of support at the infrastructure level.</p> <p>Therefore, CSP:BaaS provider provides an efficient and economical method for most enterprises to use BaaS.</p> <p>Enterprise users can directly use blockchain network provided by the CSP:BaaS provider, or customize the blockchain network according to business requirements.</p> <p>CSN:BaaS developer supports the design, development of various blockchain services, such as intelligent contract, encryption service and consensus mechanism, etc.</p> <p>CSP:BaaS provider provides CSC:BaaS customer with blockchain network together with a set of software development kit (SDKs), which help CSC:BaaS customers realize system access and DApp development. CSP:BaaS provider also supports configuration, testing and maintenance of blockchain network and is responsible for managing service levels.</p>
Roles/Sub roles	CSN:BaaS developer CSP:BaaS provider CSC:BaaS customer
Pre-conditions (optional)	
Post-conditions (optional)	

Title	Utilization of BaaS for enterprise users
Figure and operational flows (optional)	<p style="text-align: right;">Y.3530(20)_F1.1</p> <p style="text-align: center;">Figure I.1 – Utilization of BaaS for enterprise users</p>
Derived requirements	<ul style="list-style-type: none"> – refer to clause 8.1 requirement 5 – refer to clause 8.1 requirement 6 – refer to clause 8.1 requirement 7 – refer to clause 8.1 requirement 8 – refer to clause 8.1 requirement 10 – refer to clause 8.2 requirement 1

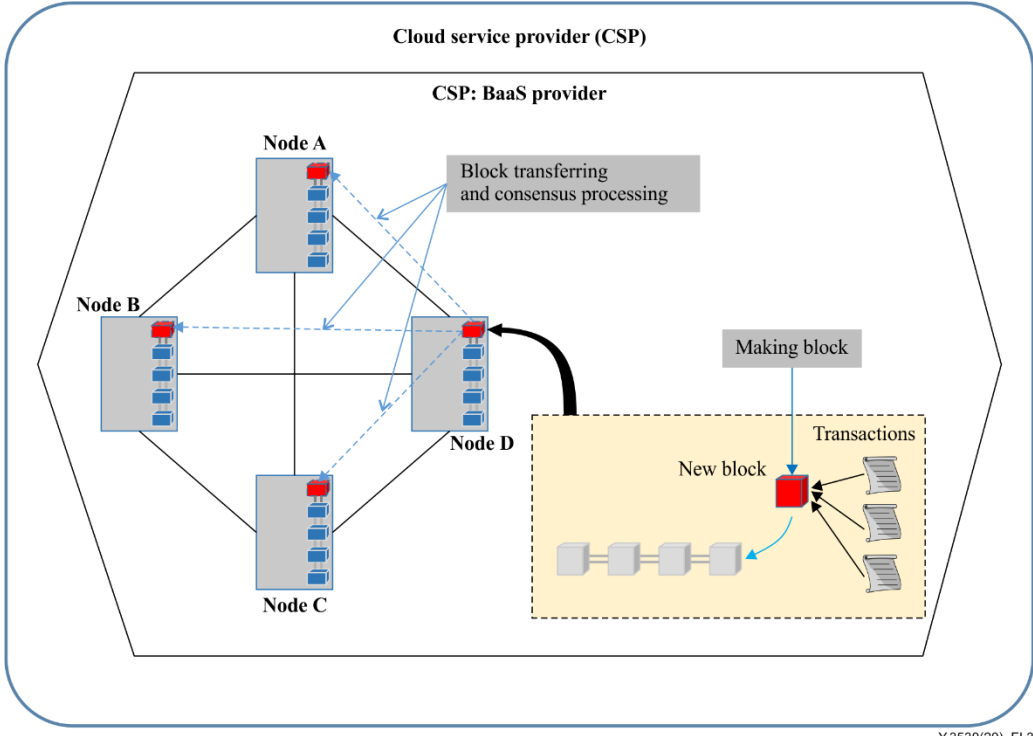
I.2 Developing DApp using blockchain as a service

Title	Developing DApp using blockchain as a service
Description	<p>This use case shows that a CSC:BaaS customer develops a DApp using BaaS platform provided by a cloud environment. When the CSC:BaaS customer informs CSP:BaaS provider that they are developing a DApp, the CSP: BaaS provider identifies the requestor and verifies permission of the CSC:BaaS customer.</p> <p>The CSP:BaaS provider provides development tools to the CSC:BaaS developer so that the CSC:BaaS developer can easily develop the service. The CSP:BaaS provider also provides an integrated development environment (IDE) which is a software application that gives computer programmers comprehensive capabilities for software development.</p>
Role/Sub-role	<p>CSP:BaaS provider</p> <p>CSC:BaaS customer</p>

Title	Developing DApp using blockchain as a service
Figure (optional)	<p style="text-align: right; font-size: small;">Y.3530(20)_FI.2</p> <p style="text-align: center;">Figure I.2 – Developing DApp using blockchain as a service</p>
Pre-conditions (optional)	
Derived Requirements	<ul style="list-style-type: none"> – refer to clause 8.1 requirement 1 – refer to clause 8.1 requirement 2 – refer to clause 8.1 requirement 3 – refer to clause 8.1 requirement 4 – refer to clause 8.2 requirement 7 – refer to clause 8.2 requirement 8 – refer to clause 8.3 requirement 1 – refer to clause 8.3 requirement 3 – refer to clause 8.3 requirement 4

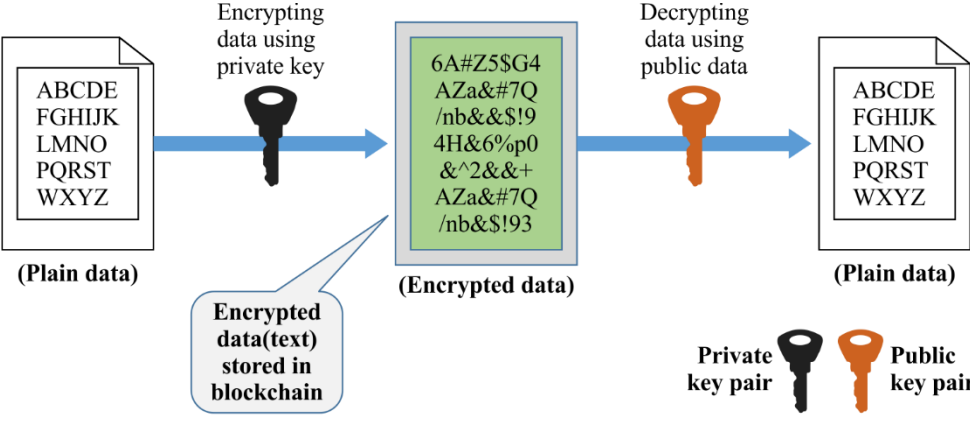
I.3 Adding new block on the blockchain

Title	Adding new block on the blockchain
Description	<p>Blockchain is a type of distributed ledger technology (DLT) that operates on a distributed node network. Individual nodes can attempt to create new block containing the received transactions and perform verification.</p> <p>The process of creating, disseminating, and agreeing on a block is as follows:</p> <ul style="list-style-type: none"> – [Collecting transactions] Node D tries to collect transaction details; – [Block proposal] Node D creates a block. When creating a block, the node D uses the hash from the previous block and the hash (root hash) generated from transactions; – [Information propagation] Node D propagates the block to all participating nodes through the blockchain network; – [Block validation] Node A, Node B, node C check the block for generation proofs and validity of enclosed transactions; – [Add new block on the blockchain] Node A, Node B, node C add the new block end

Title	Adding new block on the blockchain
	<p>of chain.</p> <p>Generally, the blockchain consensus algorithms should disclose technical details and it should support nodes to perform algorithmic operations independently, without relying on any other node data and state. And consensus algorithms should be able to prevent malware attacks and double spent problems. And it the consensus algorithm is recommended to specify the theoretical time required to reach consensus. Also, it should ensure convergence towards a single, immutable version of the ledger.</p>
Role/Sub-role	CSP:BaaS provider
Figure (optional)	 <p style="text-align: center;">Figure I.3 – Adding a new block to the blockchain</p>
Pre-conditions (optional)	
Derived requirements	<ul style="list-style-type: none"> – refer to clause 8.1 requirement 2 – refer to clause 8.2 requirement 2 – refer to clause 8.2 requirement 3 – refer to clause 8.2 requirement 4 – refer to clause 8.2 requirement 5

I.4 Digital signature and controlling transaction

Title	Digital signature and controlling transaction
Description	<p>This use case consists of two parts, the first use case provides general processors for digital signature to transaction. The second use case shows a case where multiple encryption is required because there is PII.</p> <p>In first use case, to carry out a transaction on the blockchain you need two things: a wallet, which is basically an address, and a private key. The private key is a string of random numbers, but unlike the address the private key must be kept secret. When someone decides to send coins to anyone else they must sign the message containing the transaction with their private key. The system of two keys is at the heart of</p>

Title	Digital signature and controlling transaction
	<p>encryption and cryptography. Once the message is sent it is broadcast to the blockchain network. The network of nodes then works on the message to make sure that the transaction it contains is valid. If it confirms the validity, the transaction is placed in a block and after that no information about it can be changed.</p> <p>The second use case shows a case where multiple encryption is required because there is PII. Blockchain can have associated personal information and the protection of this personal information.</p> <p>In blockchain, personally identifiable information (PII) data can be collected and stored. The problem is that once the personal information stored in the blockchain is recorded, the record cannot be changed. In this case, it is possible to protect personally identifiable information to some extent through multiple encryptions.</p> <p>The second figure shows an example of multiple encryption methods in a transaction. If there is PII in the transaction data, this transaction is handled specially. It classifies the security level according to the type of data and handles encryption differently according to the classified security level. In the case of high-security data, the operator performs encryption of the data at least once, and then performs the second encryption of the entire transaction, including the data after the first encryption.</p>
Role/Sub-role	CSP:BaaS provider (CSP:BaaS provider)
Figure (optional)	 <p style="text-align: center;">Figure I.4-1 – Case for storing transaction data into blockchain</p>

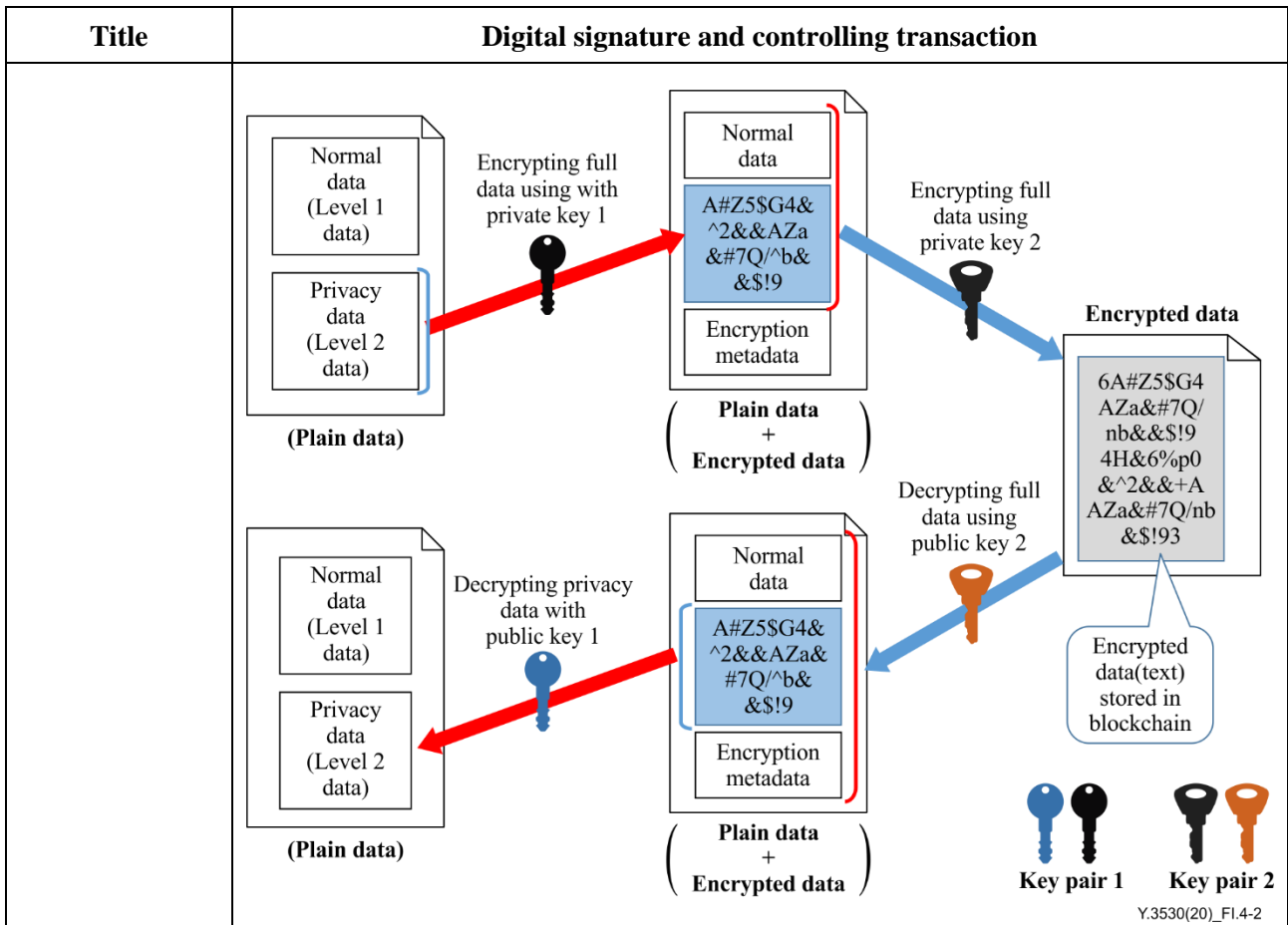
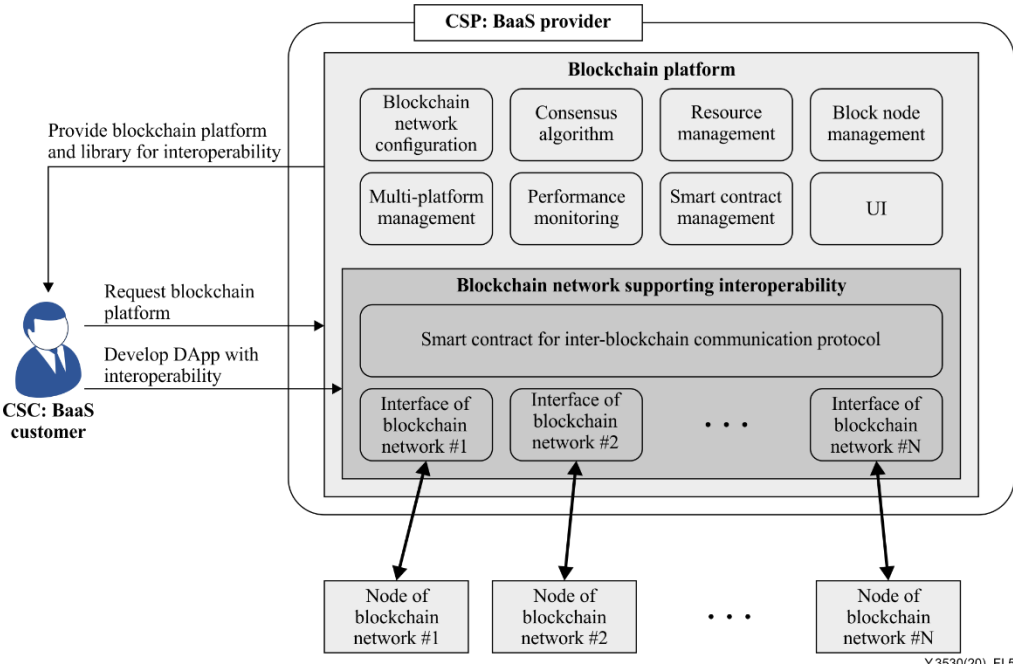


Figure I.4-2 – Case for storing transaction data with privacy into blockchain

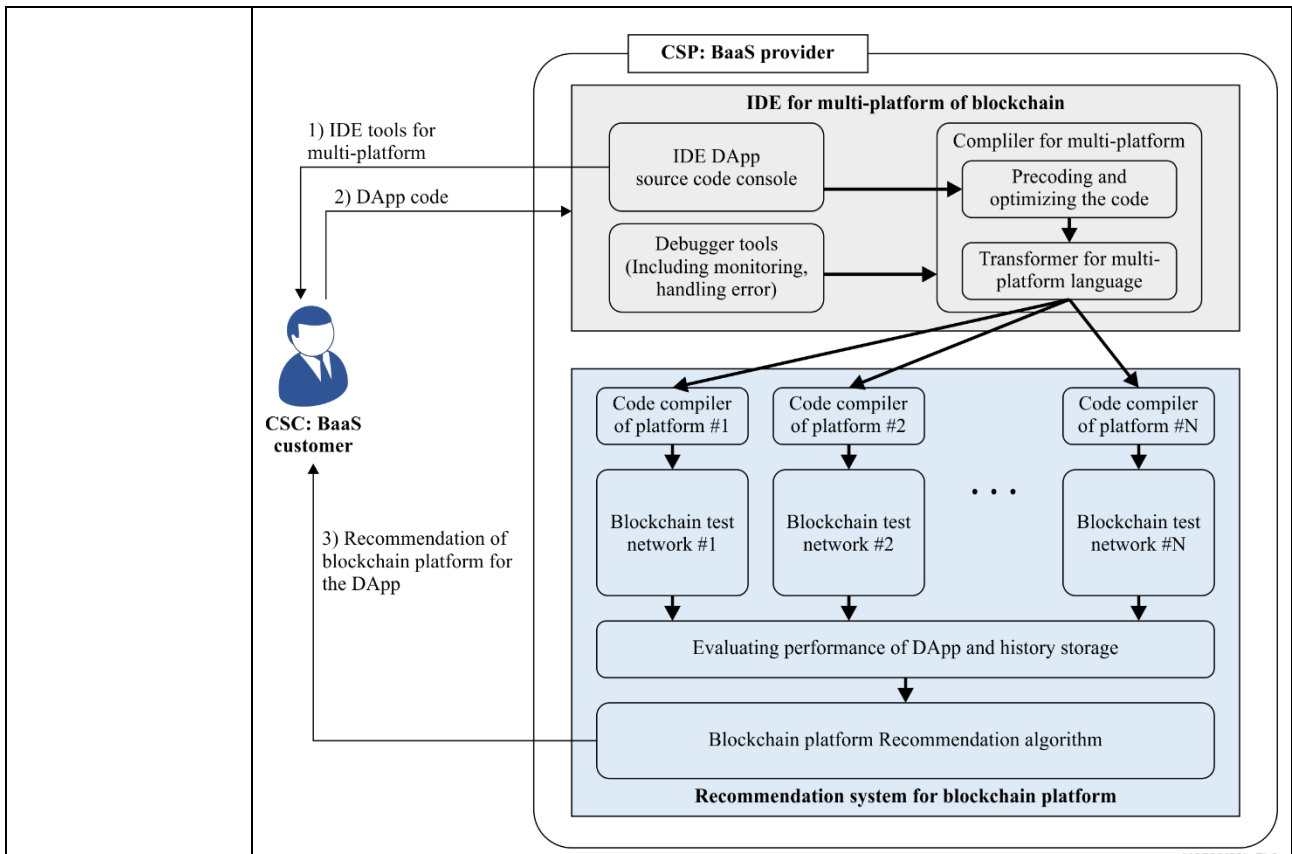
I.5 Communications among the nodes in multiple platforms

Title	Communications among the nodes in multiple platforms with blockchain network for interoperability
Description	<p>The interoperability of blockchain is the ability to share information across different blockchain networks or platforms. With interoperability, the nodes in different networks can communicate with each other even if the network operates with different consensus algorithms. The CSC:BaaS provider supports interoperability by operating the smart contract designed for inter-blockchain communication.</p> <p>This use case provides the procedure of communications among the nodes in multiple platforms with the smart contract for inter-blockchain communication. The smart contract for inter-blockchain communication executes the network protocol such as user datagram protocol (UDP) and transmission control protocol (TCP). The blockchain network for interoperability also acts as a hub for connecting multiple blockchain networks.</p>

Title	Communications among the nodes in multiple platforms with blockchain network for interoperability
	<p>The following are general steps for setting blockchain network for interoperability.</p> <ol style="list-style-type: none"> 1) CSC:BaaS customer set the configurations of blockchain platform performing high TPS. 2) CSC:BaaS customer set the configurations of blockchain network. 3-1) CSP:BaaS provider provide the smart contract code for inter-blockchain communication. 3-2) CSC:BaaS customer build the smart contract code for inter-blockchain communication. 4) CSC:BaaS provider execute mart contract code and maintain the blockchain network.
Role/Sub-role	<p>CSP:BaaS provider CSC:BaaS customer</p>
Figure (optional)	<p>The following figure shows system architecture of blockchain platform and network for interoperability.</p>  <p style="text-align: right; font-size: small;">Y.3530(20)_FI.5</p> <p style="text-align: center;">Figure I.5 – Example of blockchain platform for interoperability</p>
Pre-conditions (optional)	It is required that CSP:BaaS provider provide monitoring of resource utilization of blockchain network.
Post-conditions (optional)	
Derived requirements	<ul style="list-style-type: none"> – refer to clause 8.1 requirement 1 – refer to clause 8.2 requirement 6 – refer to clause 8.3 requirement 9

I.6 Developing and deploying DApp in multiple platforms with IDE

Title	Developing and deploying DApp in multiple platforms with IDE
Description	<p>This use case describes how to develop and deploy DApp in multiple platforms with IDE which supports the transformation of language. The transformation of language can be implemented in the compiler by supporting intermediate representation between different languages.</p> <p>In this use case, CSC:BaaS customer can develop the DApp without concerning the target platform at the beginning of DApp design. In addition, CSC:BaaS customer can deploy the DApp into multiple platforms, which executes identical functionalities.</p> <p>For those users advantage, the CSP:BaaS provider supports IDE and tools for transformation of language. Also, CSP:BaaS can support exploration of the blockchain platform for CSC:BaaS customers to meet their requirements of DApp. By evaluating the performance of DApp, CSC:BaaS recommends the configuration set for blockchain platform.</p> <p>The following are examples of processes for this use case. The examples describe the DApp deployment into multiple platforms.</p> <ol style="list-style-type: none"> 1) CSC:BaaS customer generates the code for DApp and input it to the compiler. 2) The compiler of CSP:BaaS provider performs pre-processing and optimizing code, then, transforming the code into multiple language. 3) CSP:BaaS provider tests the performance of DApp with multiple configurations and multiple platforms. 4) CSP:BaaS recommends the candidate platforms and the configurations for platforms. 5) CSC:BaaS customer chooses the platforms and requests deployment of DApp.
Role/Sub-role	<p>CSP:BaaS provider</p> <p>CSC:BaaS customer</p>
Figure (optional)	<p>The following figure shows the architecture of blockchain IDE for a multiple platform and recommendation system.</p>



Y.3530(20)_F1.6

Figure I.6 – Example of blockchain IDE for multi-platform and recommendation system

Pre-conditions (optional)	
Post-conditions (optional)	
Derived requirements	<ul style="list-style-type: none"> – refer to clause 8.1 requirement 1 – refer to clause 8.1 requirement 9 – refer to clause 8.1 requirement 11 – refer to clause 8.1 requirement 12 – refer to clause 8.3 requirement 1 – refer to clause 8.3 requirement 2 – refer to clause 8.3 requirement 6

Appendix II

Use cases and scenarios of applications for blockchain as a service

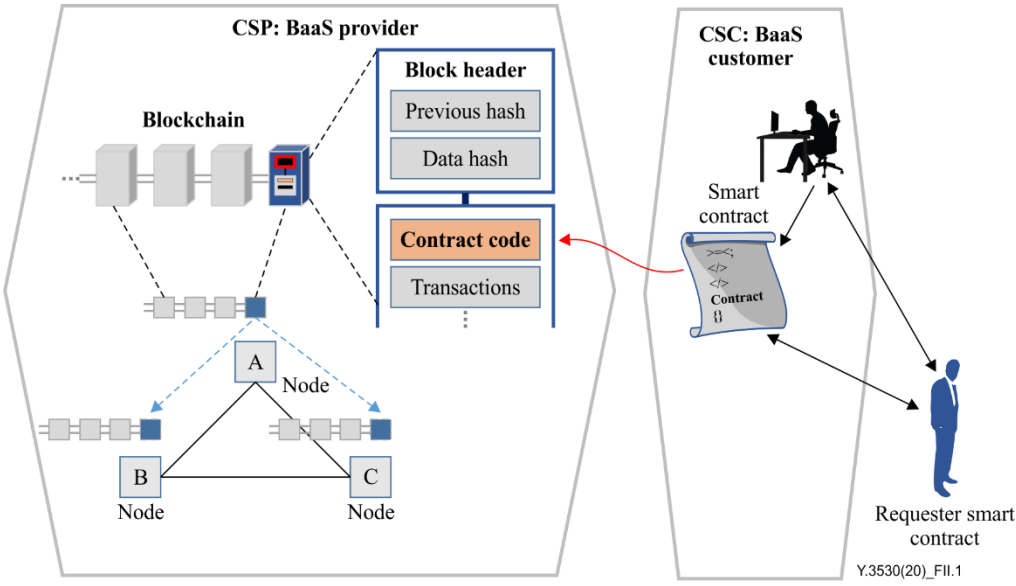
(This appendix does not form an integral part of this Recommendation.)

This appendix describes use cases and scenarios of applications for blockchain as a service (BaaS).

Table II.1 describes creating smart contracts in CSP:BaaS provider.

Table II.2 describes the deployment and operation of a smart contract.

II.1 Creating smart contract in CSP:BaaS provider

Title	Creating smart contracts in CSP:BaaS provider
Description	<p>A smart contract is executable programs. They are usually written in high-level computer programming languages in order to represent arbitrary business logic or pre-determined criteria to trigger transfer of values. A smart contract is a self-enforcing agreement embedded in computer code (contract code) managed by a blockchain. The contract code contains a set of rules under which the parties of that smart contract agree to interact with each other. If and when the predefined rules are met, the agreement is automatically enforced. A contract code is the set of instructions forming the software which is executed by a computer.</p> <p>The procedure for creating a smart contract is as follows:</p> <ol style="list-style-type: none"> 1) The blockchain user who wants to create a smart contract requests the CSC:BaaS user with transactions and related information. 2) First, a blockchain user who wants to trade using smart contract requests the CSC:BaaS user to generate smart contract. 3) The CSC:BaaS user receives the requestor's transactions and conditions of contracts. 4) The CSC: BaaS user writes and authenticates the smart contract in a new block. 5) Blocks with proof of work are delivered to all nodes and saved.
Role/Sub-role	<p>CSP:BaaS provider CSC:BaaS customer</p>
Figure (optional)	 <p style="text-align: center;">Figure II.1 – Creating smart contracts in CSP:BaaS provider</p> <p style="text-align: right; font-size: small;">Y.3530(20)_FII.1</p>
Pre-conditions	

Title	Creating smart contracts in CSP:BaaS provider
(optional)	
	<ul style="list-style-type: none"> – refer to clause 8.3 requirement 2 – refer to clause 8.3 requirement 3 – refer to clause 8.3 requirement 4 – refer to clause 8.3 requirement 5 – refer to clause 8.3 requirement 7 – refer to clause 8.3 requirement 8

II.2 Deployment and operation of a smart contract

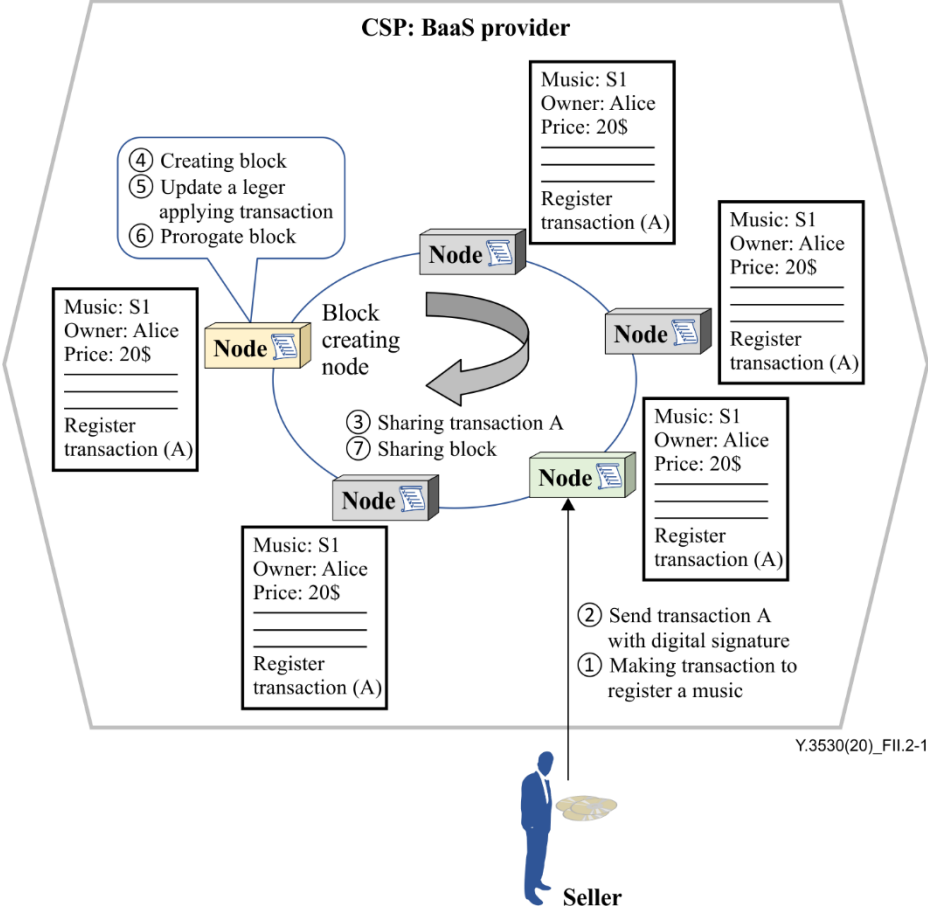
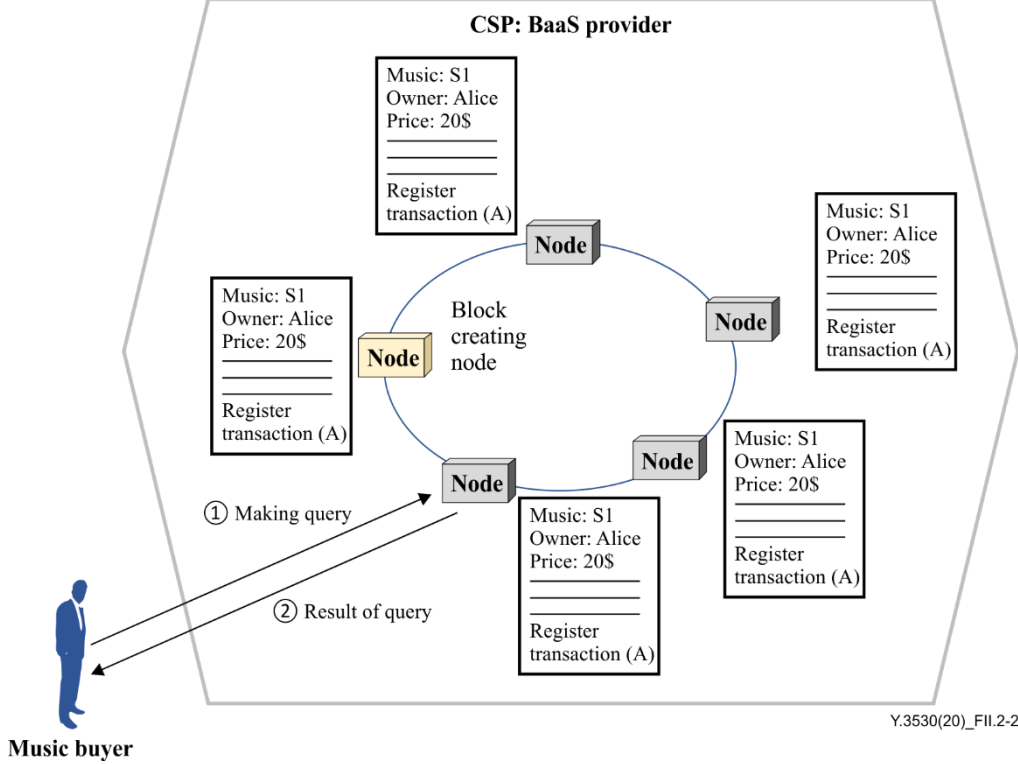
Title	Deployment of a smart contract
Description	<p>The procedure for creating a smart contract is as follows:</p> <ol style="list-style-type: none"> 1) making transaction to register a music 2) send transaction (A) with digital signature 3) sharing transaction (A) 4) creating block 5) update a ledger applying transaction 6) prorogate the block 7) sharing the block
Role/Sub-role	<p>CSP:BaaS provider CSC:BaaS customer</p>
Figure (optional)	 <p>The diagram illustrates the deployment of a smart contract within a CSP: BaaS provider. A Seller (represented by a person icon) initiates the process by sending a transaction (A) with a digital signature to a Node. This transaction registers music (S1) with an owner (Alice) and a price (20\$). The Node then shares this transaction (A) with other Nodes in the network. The Nodes perform the following steps: 1) Making transaction to register a music, 2) Send transaction A with digital signature, 3) Sharing transaction A, 4) Creating block, 5) Update a ledger applying transaction, 6) Prorogate block, and 7) Sharing block. The diagram shows multiple Nodes, each with a ledger containing the music details and the transaction (A). A central Node is labeled 'Block creating node'. The diagram is labeled 'CSP: BaaS provider' and includes the reference 'Y.3530(20)_FII.2-1'.</p>

Figure II.2-1 – Deployment of a smart contract

Title	Deployment of a smart contract
	 <p style="text-align: center;">CSP: BaaS provider</p> <p>Music: S1 Owner: Alice Price: 20\$ Register transaction (A)</p> <p>Node</p> <p>Block creating node</p> <p>Node</p> <p>Node</p> <p>Node</p> <p>Node</p> <p>Music: S1 Owner: Alice Price: 20\$ Register transaction (A)</p> <p>Music: S1 Owner: Alice Price: 20\$ Register transaction (A)</p> <p>Music: S1 Owner: Alice Price: 20\$ Register transaction (A)</p> <p>Music: S1 Owner: Alice Price: 20\$ Register transaction (A)</p> <p>① Making query</p> <p>② Result of query</p> <p>Music buyer</p> <p style="text-align: right;">Y.3530(20)_FII.2-2</p> <p style="text-align: center;">Figure II.2-2 – Deployment of a smart contract</p>
Pre-conditions (optional)	
	<p>refer to clause 8.1 requirement 5</p> <p>refer to clause 8.3 requirement 3</p> <p>refer to clause 8.3 requirement 4</p> <p>refer to clause 8.3 requirement 5</p> <p>refer to clause 8.3 requirement 7</p>

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.1401] Recommendation ITU-T X.1401 (2019), *Security threats of distributed ledger technology*.
- [b-ITU-T X.1402] Recommendation ITU-T X.1402 (2020), *Security framework for distributed ledger technology*.
- [b-ITU-T X.1601] Recommendation ITU-T X.1601 (2015), *Security framework for cloud computing*.
- [b-ITU-T X.1631] Recommendation ITU-T Y.1631 (2015), *Information technology – Security techniques – Code of practice for information security controls based on ISO/IEC 27002 for cloud services*.
- [b-FG-DLT RA] ITU-T Deliverable of FG-DLT (2019), *Distributed ledger technology reference architecture*.
- [b-FG-DLT Terms] ITU-T Deliverable of FG-DLT (2019), *Distributed Ledger Technology terms and definitions*.
- [b-ISO CD 22739] ISO CD 22739, *Blockchain and distributed ledger technologies – Terminology*.

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