**T/CIE 095-2020**

# Formal Expression of Blockchain Smart Contract

(Blockchain— Smart Contract— Formal Expression)

REFERENCE VERSION 1.0

| | |
|---|---|
| Date Issued: | 12/21/2020 |
| Effective Date: | 01/02/2021 |
| Issuing Authority: | Chinese Institute of Electronics |

Chinese Institute of Electronics

## Contents

## Foreword

This document was drafted according to the regulation of GB/T 1.11-2020, *Guidelines for Standardization Part 1: Structure and Drafting Rules of Standardization*.

This document was proposed by University of Science and Technology Beijing.

This document is under the technical jurisdiction of the Blockchain Branch in Chinese Institution of Electronic.

The drafted institutions of this document: University of Science and Technology Beijing, Peking University, Beijing Municipal Bureau of Economy and Information Technology, Beijing Society Public Trust Construction Promotion Association, JoToo (Tianjin) Information Technology Company Co., Ltd., Beihang University, Sinochem Energy Company Co., Ltd., Beijing Wenzhang Wuyou Information Technology Co., Ltd., Ke Holdings (Beijing) Inc., JD Digits Technology Holding Co., Ltd..

The main drafters of this document: Yan Zhu, Bohan Qin, Xiao He, Di Wang, Yizi Sun, Qian Guo, Weijing Song, Shengdian Wang, Jingyi Hong, Qian Yao, Jining Li, Guowei Liu, Ran Yi, Rongquan Feng, Tao Zhang, Kai Hu, Shuangquan Xu, Xinhui Han, Wei Ying and Guohua Gan.

## Introduction

Blockchain is an append-only distributed ledger system with chain structure protected by cryptographic techniques. Beyond blockchain, smart contract further permits developers to write auto-executing programs using programming language for various applications related to value exchange and preservation of evidences. Meanwhile, the demand for standardization of smart contracts is becoming increasingly strong with the development of digital economy. However, there are several practical problems for the existing smart contracts, such as strong specialty, bad readability and low productivity, so as to perform conversion difficultly from real-world legal contract to executable program codes. It not only affects industry applications and cross-boundary cooperation between computer and legal workers, but also impedes the legalization process of smart contracts.

This document provides an advanced smart contract language with concrete grammar rules complying with law, on which a kind of smart contract, called smart legal contract, is built as a transitional legal file between real-world legal contract and smart contract. As shown in Figure 1, a real-world legal contract typically written in natural language can be translated into a smart legal contract in the proposed language, further transformed into a smart contract program written by smart contract language.

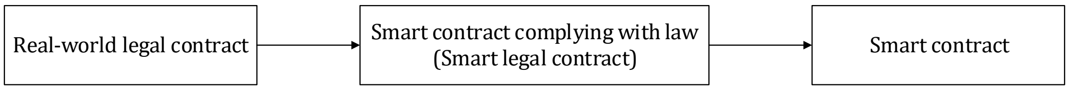
Real-world legal contract
Smart contract complying with law (Smart legal contract)
Smart contract

**Figure 1— Diagram of contract transformation relationship**

Smart legal contracts make use of programming codes to express contractual terms for connecting real-world legal contracts with programming codes in cyberspace. It ensures that smart legal contract possesses not only legal characteristic and easy-to-understand of real-world legal contract, but also the normalization of programming code. Moreover, it can promote the cross-boundary cooperation between the professionals of computer and law. On the strength of blockchain's capability on confirmation of rights, smart legal contracts can use digital assets to express physical assets, e.g., houses, healthy data and copyright, and ensure that digital assets can circulate in blockchain network as freely as they normally do by integrating them with programmable digital legal currency. Therefore, the standardization of smart legal contract promotes the rapid, sustained, and healthy development of digital economy.

The issue institution of this document draws attention to the fact that it is claimed that compliance with this document may involve the use of a patent.

The issue institution of this document takes no position concerning the evidence, validity and scope of this patent right.

The holder of this patent has assured the issue institution that he/she is willing to negotiate licences—reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent rights is registered with the issue institution. Information may be obtained by the following contact:

Patent: 202010381549.5 an executable smart contract construction for execution method and system of legal contracts.

The holder of this patent: University of Science and Technology Beijing.

Address: No.30 Xueyuan Road, Haidian District, Beijing.

Attention is drawn to the possibility that some of the components of this document may still involve patents. The issue institution shall not be held responsible for identifying any or all such patent rights.

# **Formal Expression of Blockchain Smart Contract** (Blockchain — Smart Contract — Part 1: Formal Representation)

## 1 Scope

This document defines the structure and grammar of smart contract language, and fixes the corresponding terminologies and definitions.

NOTE The examples of smart legal contract and smart contract provided in this document are shown in Annex A.

This document is suitable for the design, development and application of smart contracts, as well as the reference of smart contract platform constructed by blockchain manufacturers and users.

## 2 Normative references

There are no normative references in this document.

## 3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

**3.1**
**Smart contract**
Computer program, deployed on blockchain, which is intended to automatically execute with evidence preservation according to the predefined terms of contract.

**3.2**
**Smart contract complying with law**
Computer program, consisting of contract essential elements, which is intended to perform the agreements by contracting parties according to offer and promise.

NOTE In this standard, it is abbreviated as smart legal contract, in the case without causing confusion.

**3.3**
**Smart contract language**
Formal specification, including vocabularies and grammar rules, that is used to define smart contracts.

**3.4**
**Smart contract language complying with law**
Program language that is used to develop smart contract for meeting the legal requirements.

NOTE In this standard, it is abbreviated as smart legal contract language, in the case without causing confusion.

**3.5**
**Smart contract platform**
Information network system that supports the development, generation, deployment, execution and verification of the executable program of smart contract.

**3.6**
**Account**
The carrier with a special format and structure to describe parties, operations, the increase and decrease of elements, i.e., contract objects, and the outcomes of changing in smart legal contract.

**3.6.1**
**Party account**
Account owned by a contracting party.

**3.6.2**
**Contract account**
Account created and hold by a particular smart legal contract when the contract is deployed to smart contract platform.

## 4 Symbols and keywords

### 4.1 Symbols

| | |
|---|---|
| @@ | The prefix of contract text expressed by natural language. |
| ::= | Denote defining, i.e. “be defined as”. |
| ? | Denote the pre-keyword is optional. |
| \| | OR operation of elements with same level. |
| { } ( ) | The set of alternative sentences. |
| . | The end symbol of sentences. |
| + | Zero or more sentences. |
| // | The interpretation symbols. |
| ‘ ’, ” ” | The type of strings. |
| 0x | The prefix of hexadecimal numbers. |
| , space | Denote the juxtaposition and separation of elements with same level, respectively. |

### 4.2 Keywords

The involved keywords and the corresponding implications in this document are shown in Table1.

## 5 Representation

**5.1**
Smart legal contract is in the form of data message. The relevant parties shall conclude a smart legal contract in the form of offer and acceptance.

**5.2**
Smart legal contracts, which have the same legal effect as other legal contracts, must conform to *Civil Code of the People’s Republic of China*, *Electronic Signature Law of the People’s Republic of China* and the other relevant laws and regulations. The sentences used by both smart legal contracts and real-world legal contracts shall be construed to have the same meaning. When there is a contradiction of sentences used by them, they shall be interpreted according to the corresponding contract terms, properties, objective and principle of honesty.

**Table1— The keywords and the corresponding implications**

| Keyword | Implication |
|---|---|
| **he, she, his, her, himself, herself, this, the** | The current entity, equalling to 'this' in program language. |
| **=, is** | Equals to. |
| **:** | The separated symbol. |
| **::** | The reference of attribute information. |
| **!=, <>, isn't** | Not equals to. |
| **all, for all, some, exist** | The universal and existential quantifiers. |
| **can, must, cannot** | The limitation of right, obligation and prohibition. |
| **origin** | The balance of account before the operation is executed. |
| **after, before, within** | Later, earlier or during a particular period of time. |
| **did** | Refer to a party has done something, which is usually used with **after**. |
| **true, false** | Boolean values. |
| **value** | The number of assets transferred by party. |
| **and, or, not, implies** | The logical symbols. |
| **>, >=, <, <=, belongs to** | The relationship symbols. |
| **Cname, Pname, Aname, Tname, Bname, Dname** | The name of contract, party, asset, general term, breach term, additional information, collectively referred as Entity. |
| **year, month, date, hour, minute, second, now** | Time symbols. |
| **String, Money, Date, Integer, Float, Boolean, Right, Time** | The type symbols of variables. |
| **when, while, where** | The reserved words of condition in terms. |
| **transfer, withdraw, deposit** | The reserved words of asset operations. |
| **contract, info, right, party, group, asset, term, breach term, arbitration term, contract conclusions, signature of party, additions, serial number, of, to, institution** | The other reserved words. |

**5.3**
Smart legal contract can be converted into executable programs represented in the form of data message. After smart legal contract is actually signed, the converted programs will be executed automatically with preserved evidences in a certain smart contract platform.

**5.4**
Smart legal contracts can be written in Chinese, English or Chinese-English bilingual form according to different scenarios.

**5.5**
The life cycle of smart legal contract consists of the following three stages:

a) Generation stage: after contracting parties reach an agreement, a smart legal contract will be written and further translated into an executable program.

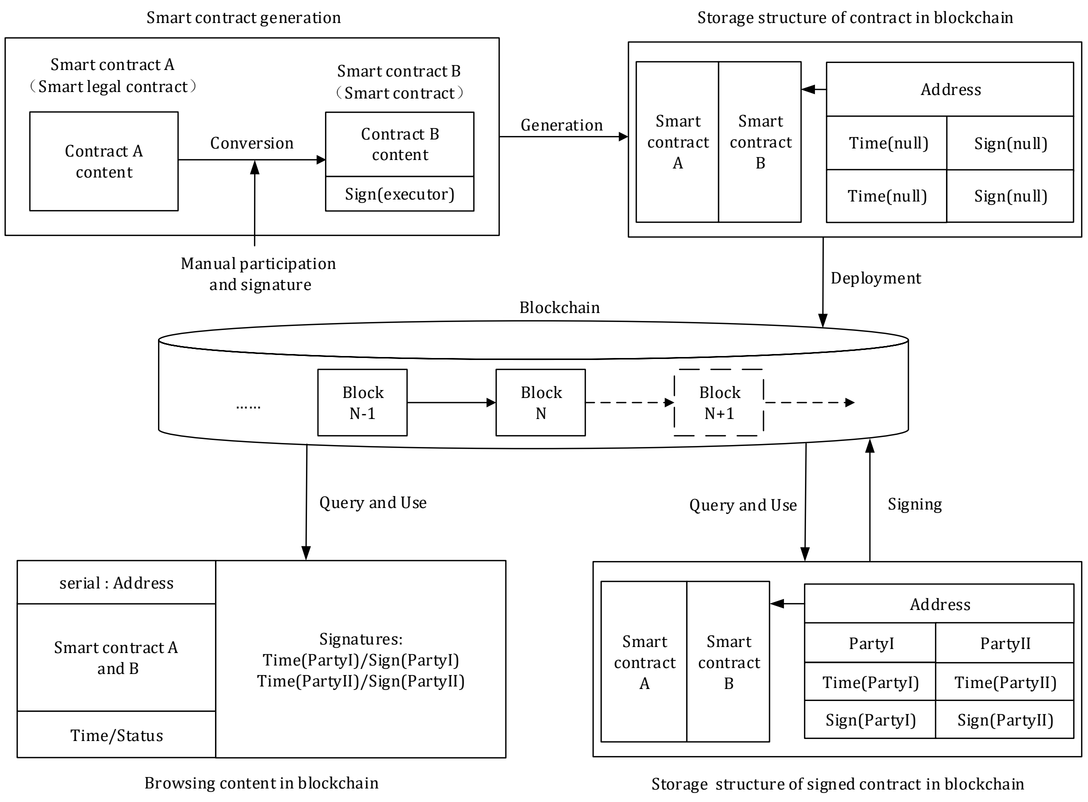

**Fig.2— Diagram of concluding, signing and acquiring process of smart contract**

b) Signing stage: the smart legal contract, combined with the translated executable program and null sign-up form, is deployed into blockchain, and then each of parties can obtain it and sign to the form for concluding, so that the signed form will be written into blockchain as preserved evidence. The signed smart legal contract can be acquired, checked and used at any time. The corresponding concluding, signing and acquiring process can refer to the example of Fig.2 (details as given in Annex B).

c) Execution stage: when the pre-condition in contract terms is satisfied, the blockchain nodes can acquire and run the corresponding executable program codes, then modify the contract status and write them into blockchain as preserved evidence until the termination of the contract.

## 6 Elements

Smart legal contract may contain parties' information, contract object, amount, quality, price or remuneration, time limit and manner on enforcement, liability for breach, way to solve controversy. The essential elements of smart legal contract shall include contract name, party description, object, contract terms, additional information and contract conclusion, where contract terms include general terms, breach terms and arbitration terms. The writing process of smart legal contract involves grammar regulations such as right, obligation, asset operation and expression, and time expression. The relationship of the constituting elements is shown as Fig.3.

## 7 Representation of elements

### 7.1 Contract framework

A smart legal contract consists of title and contract content, where contract content commonly includes party description, asset description, terms, additional information and contract conclusion.

***Contracts* ::= *Title{Parties+ Assets+ Terms+ Additions+ Signs+}***

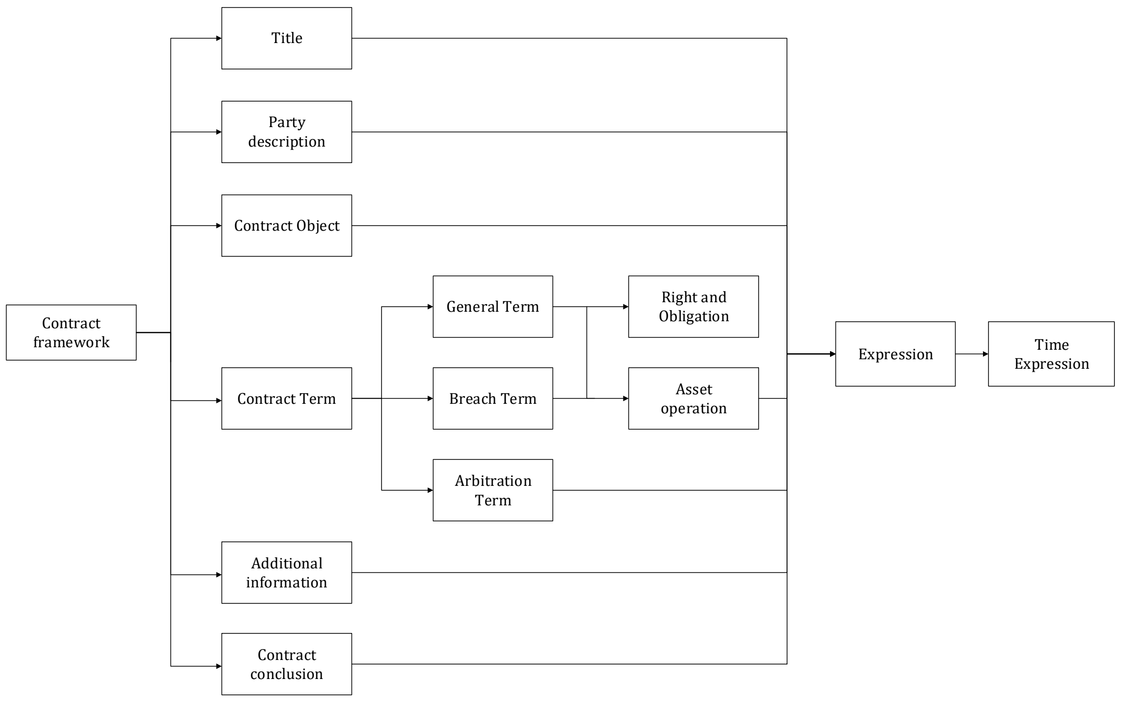

**Fig.3— Relationship diagram of constituting elements of smart legal contract**

## 7.2 Title

Title consists of contract name and contract serial number. The grammar is shown as

***Title*** ::= **contract** Cname (: **serial number** Chash)?

where

| | |
|---|---|
| Cname | is contract name; |
| Chash | is contract serial number. |

NOTE The contract serial number refers to a unique number generated by calculating the hash value of smart legal contract.

EXAMPLE

@@ purchase contract No. 0x827198…ab193

**contract** purchase : **serial number** 0x827198…ab193

## 7.3 Party description

Party description may include a set of party's attribute and its value, such as the party's name, surname, address and account. Entity authentication can be applied to guarantee the uniqueness of party identity. The grammar is shown as

***Parties* ::= party group**? Pname {*field*+}

where

| | |
|---|---|
| Pname | is the name of party; |
| field | is to describe party's attribute and its value, represented by a colon-separated two-tuple. |

The field can be described as

*field* ::= attribute : (constant|type)

where

| | |
|---|---|
| attribute | is the attribute name; |
| constant\|type | is the attribute value. |

NOTE The attribute value may be a constant value or the type of variable. When an attribute value is the type of variable, its initial value is null.

In this document, the reference of entity's attribute is expressed in the form **Entity:: attribute**, and the corresponding outcome refers to its value.

EXAMPLE 1 **party** Seller {account: 0x7c84e8…2934 name: 'Zhang San'}

NOTE The party account refers to his account address in blockchain.

EXAMPLE 2 **party group** Voters {account: Integer}

NOTE The group party can be represented as a dynamic list of parties.

## 7.4 Contract Object

Contract object refers to the thing related to both right and obligation between parties, which is divided into goods, action, intellectual property, etc. A contract object is represented by assets in a smart legal contract, and there shall exist the description of these assets in blockchain. The grammar is shown as

***Assets*** **::= asset** Aname {**info**{*field*+} **right**{*field*+}}

where

| | |
|---|---|
| Aname | is the name of the asset; |
| **info** | is to describe the asset's attributes and their values; |
| **right** | is to specify the ownership of the asset, such as the right of ownership (ownershipRight), use (useRight), possession (possessRight), usufruct (usufructRight), disposal (disposeRight). |

NOTE The right of the asset can be used as a type. Also, new right can be defined according to actual demands.

NOTE The definition of asset shall include name, attributes and ownerships.

EXAMPLE 1

```
asset House{
    info {sn : 0x71a2f8…78d93  area : 50  usage : "business"  price : Money}
    right {houseBenefitRight : usufructRight  houseUse : useRight}
}
```

NOTE Both usufructRight and useRight are used to record ownership information; and Money is used to represent monetary asset.

In this document, asset expression is used for the reference of a certain asset in contracting terms. The definition is shown as

***AssetExpressions* ::= $** (amount)? (right **of**)? Aname

where

| | |
|---|---|
| Aname | is the claimed asset in a smart legal contract; |
| right | is the claimed right of the defined asset, and the default value is ownership right if there is no ownership description. |

EXAMPLE 2 $ 20 RMB

NOTE The predefined monetary assets include RMB, USD, etc.

EXAMPLE 3 $ House

NOTE The house asset claimed in smart legal contract.

EXAMPLE 4 $ 120%*principal

NOTE The principal is an asset of Money type.

EXAMPLE 5 $ 50% ownershipRight **of** House

NOTE The 50% ownership of house asset claimed in smart legal contract.

## 7.5 Contract Term

Contract term is divided into general term, breach term and arbitration term.

***Terms* ::= *GeneralTerms* | *BreachTerms* | *ArbitrationTerms***

### 7.5.1 General term

General term consists of term's name, party of term, party's right and obligation (must, can or cannot do action), required condition before execution, asset transfers, and required condition after execution. The grammar is shown as

*@@ The term stipulates what kind of actions a party must, can or cannot do when what pre-conditions, while what asset operations, and where what post-conditions that should be satisfied.*

***GeneralTerms* ::= term** Tname : Pname **(must|can|cannot)** action (*field*+)
(**when** preCondition)?
(**while** transactions+)?
(**where** postCondition)?

where

| | |
|---|---|
| preCondition | is the pre-condition that consists of the pre-condition expressions; |
| transactions | are asset operations in the execution of term; |
| postCondition | is the post-condition that consists of the post-condition expressions. |

NOTE The preCondition will be checked before the execution of term. If it is satisfied, the term is allowed to be executed; otherwise, the term cannot be executed.

NOTE The postCondition will be checked after the execution of term. If it is satisfied, the term is executed successfully; if not, the term fails to be executed.

NOTE The action, which refers to the operation performed on its following list of attributes, will be implemented by programs in smart contract platform.

EXAMPLE

@@ term 1: The bidder can bid after bidding starts, and then transfers the funds, greater than the current highest price, to the contract account. Finally, the bidder with the highest price is the winner.

**term** no1: bidder **can** bid()
    **when after** bidBegin
    **while deposit value** > highestPrice
    **where** winner = **this** bidder.

### 7.5.2 Breach term

Breach term refers to the term for legal liability which the party faces, if a party fails to perform any of its obligations under this smart legal contract, or if any representation by the party under the term is materially untrue or inaccurate. When the post-condition of the designated terms is not satisfied and the pre-condition of the breach term is satisfied, the related party must or can take action for setting the defaults, which may require to enforce asset operations and satisfy the post-condition of the breach term for the result of enforcement.

*@@ The breach term stipulates what kind of actions a party must or can do based on a liability against the obligation of which of terms when what pre-conditions, while what asset transactions, and where what post-conditions that should be satisfied.*

***BreachTerms*** **::= breach term** Bname (**against** Tname+)? : Pname **(must|can)** action(*field*+)
(**when** preCondition)?
(**while** transactions+)?
(**where** postCondition)?.

NOTE The breach term commonly stipulates the actions that must be executed by defaulter or can be executed by victim.

EXAMPLE

@@ The owner must compensate the buyer for liquidated damages if the house owner rents out the house after the buyer has booked it.

**breach term** no6 **against** no4 : houseOwner **must** compensate()
    **when** houseOwner **did** lend **after** buyer **did** order
    **while transfer** default **to** buyer.

### 7.5.3 Arbitration term

Arbitration term stipulates the method to solve controversy in smart legal contracts. The specific controversy can be stated by nature language and an arbitration institution may be designated. The syntax is shown as

***ArbitrationTerms*** **::= arbitration term** : (The statement of any controversy)?
administered by **institution** : instName.

NOTE The nodes with jurisdiction in blockchain can be designated as the arbitration institutions.

EXAMPLE

**arbitration term**: Any labor controversy or claim arising out of or relating to this contract, or the breach thereof, shall be settled by arbitration administered by **institution** : Beijing Labor Arbitration Commission.

## 7.6 Right and obligation

### 7.6.1 Right

Right restriction shall use the keyword **can**, which is used to state that a party can execute the term or not when the pre-condition is satisfied.

EXAMPLE

@@ term 2: The voter can vote after the chair person has made a proposal.

**term** no2: voter **can** vote (target)
    **when after** chairPerson **did** propose.

NOTE Voting is a right. A voter can exercise his right to vote for the proposal, or does not exercise his right to abstain from voting.

### 7.6.2 Obligation

Obligation restriction includes the required restriction and prohibited restriction.

——The required restriction shall use the keyword **must**, which is used to state that a party must execute this term when the pre-condition is satisfied.

EXAMPLE 1

@@ term 3: The borrower is required to repay the loan within two years after loaning.

**term** no3: borrower **must** return (loan)
    **when within** 2 year **after** borrower **did** lend.

NOTE The action, return (loan), belongs to the obligation, the party need to fulfil his obligation within the stated period.

——The prohibited restriction shall use the keyword **cannot**, which is used to state that a party cannot execute this term when the pre-conditions are satisfied.

EXAMPLE 2

@@ term 4: The house owner cannot rent the house after the buyer orders it.

**term** no4: houseOwner **cannot** rent ()
    **when after** buyer **did** order.

A term with prohibited restriction can stipulate multiple ways for restriction, in which the restriction may be placed in either pre-condition or post-condition of action execution.

EXAMPLE 3

For “A voter cannot vote for himself”, if a voter votes for the candidate directly, there are two expressions:

@@ term 5_1: The voter cannot vote when his voting target is himself (as pre-condition of action execution).

**term** no5_1: voter **cannot** elect (target)
    **when** target = **this** voter.

@@ term 5_2: The voter cannot vote if the election result is to increase his vote by one (as post-condition of action execution).

**term** no5_2: voter **cannot** elect (target)
    **where this** voter :: candidate = **this** voter :: **origin** candidate + 1.

NOTE For the term 5_1, the situation will be permitted if the voter sends his vote to an agent and this agent votes for this voter. However, it can be avoided if using the term 5_2 to restrict the execution result.

## 7.7 Asset operation

Asset operation refers to the different ways of operating assets, which are usually used to realize the transference of contract object between different accounts, during the execution of smart legal contract. Asset operations can be divided into three categories.

### 7.7.1 Deposit

The party can deposit assets voluntarily from his party account to contract account. The deposit operation is applied into the transaction of action execution in the term. The party can designate the deposited assets directly by asset expression, and restrict the assets through comparing two values by relational operation to determine the relationship between them. The latter is used to grant the permission for transferring the designated assets only if the relationship is satisfied. The syntax is shown as

***Deposits*** ::= **deposit** (**value** RelationOperator)? ***AssetExpression***

where

| | |
|---|---|
| RelationOperation | is a relational operation symbol. |

EXAMPLE 1

@@ To deposit a higher amount of money than the current highest price.

**deposit value** > highestPrice

NOTE In the term, the auction operation can be performed under the condition that the deposited amount is higher than the current highest price highestPrice, where the highestPrice is an asset of Money type.

EXAMPLE 2

@@ To deposit a higher amount than 10 RMB.

**deposit value** > $10 RMB

### 7.7.2 Withdraw

The party can withdraw assets from contract account in the execution of terms, where the assets are designated by the asset expression. The syntax is shown as

***Withdraws*** ::= **withdraw** ***AssetExpression***

EXAMPLE

@@ To withdraw the principal and its interest (multiple amount in the AssetExpression by (1+rate)).

**withdraw** principal * (1+rate)

### 7.7.3 Transfer

The party can transfer assets from contract account to other party account in the execution of terms. The syntax is shown as

***Transfers*** ::= **transfer** ***AssetExpression*** **to** target

where

| | |
|---|---|
| target | is a party account hold by the recipient of the transferred assets. |

EXAMPLE 1

@@ To transfer the welfare to the seller.

**transfer** welfare **to** seller

NOTE The welfare, belongs to Money type, is an asset deposited by the buyer in advance. The welfare in contract account can be transferred to the seller when the buyer confirms good receiving.

A term can contain multiple statements on asset operations.

EXAMPLE 2

@@ term 6: The borrower can mortgage the house by depositing the ownership of the house into contract account, and withdraw the agreed funds, that is specified by HousePrice.

**term** no6: borrower **can** mortgage ()
    **while deposit $**House **and withdraw** HousePrice.

## 7.8 Symbols of expression

Smart legal contracts use expressions of programming language to standardize contract content. The expression is a syntactic entity that may be evaluated to determine its value.

NOTE The result of the pre-condition or post-condition expression is Boolean in the term.

The symbols used in the expressions include:

——Logical symbols, including: **and**, **or**, **not**, **implies**;

——Relationship symbols, including: >, >=, <, <=, =, !=, **belongs to**;

——Arithmetic symbols, including: +, -, *, /, %;

——Constant symbols, including: 0-9, A-Z, a-z, **true**, **false**;

——Time symbols, including: **month**, **day**, **year**, **hour**, **minute**, **second**, **now**;

——Type symbols, including: **String**, **Money**, **Date**, **Integer**, **Float**, **Boolean**, **Right**, **Time**.

## 7.9 Time representation

Time representation includes time point expression and time range expression.

### 7.9.1 Time point expression

Time point expression includes four types: time variable, time constant, query for global time, and query for action enforced time.

——Time variable refers to a variable with **Date** type.

——Time constant refers to a time point that cannot be changed in the execution of term.

EXAMPLE November 18, 2019.

——Query for global time, provided by smart contract platform, refers to a time point related to the execution of smart legal contract.

EXAMPLE The effective_date is to obtain the entry-into-force time of smart contract in blockchain.

EXAMPLE **now** is to obtain the current time.

——Query for action enforced time refers to the time point when an action is accomplished by the party. The syntax is shown as

***ActionEnforcedTimeQuery*** **::=** (**all**|**some**|**this**)? party **did** action

NOTE According to the type of parties, the query can be divided into two cases:

——When the party is an individual, it is not necessary for the articles, **all**, **some**, and **this**.

EXAMPLE 1

@@ The time point when the buyer's payment is completed.

buyer **did** pay

——When the party is a group, the specific time can be queried by using the article, **all**, **some**, or **this**.

- The article **all** can be used to obtain the time point when the last individual completes a certain action in the group.

  NOTE The query result is 'incomplete' only if any individual does not complete it.

  EXAMPLE 2

  @@ The time point when all voters complete voting.

  **all** voter **did** vote

- The article **some** can be used to obtain the latest time point when an individual completes a certain action in the group.

  NOTE The query result is 'incomplete' only if none of individuals completes it in the group.

  EXAMPLE 3

@@ The latest time point when an individual completes the auction in the group of bidders.

**some** bidder **did** bid

- The article **this** can be used to obtain the time point when the executor, who belongs to the group associated with the current term, completes a certain action.

  EXAMPLE 4

  @@ The time point when the voter completes voting.

  **this** voter **did** vote

### 7.9.2 Time range expression

Time range expression includes four types: time variable, time constant, time predicate, and bounded time predicate. Two latters belongs to time range predicate that is a condition expression over time that evaluates to a Boolean value, either true or false.

——Time variable refers to a variable with **Time** type.

——Time constant refers to a time range that cannot be changed in the execution of term.

EXAMPLE 1 day, 2 hours.

——Time predicate can be used to evaluate whether or not the target time is before (or after) a given base time. The syntax of time predicate is shown as

***TimePredicate*** **::= (**targetTime)? (**is|isn't**) (**before|after**) baseTime

where

| | |
|---|---|
| targetTime | is a time point expression; |
| baseTime | is a time point expression. |

NOTE The base time will be compared with the current time by default if the target time is not set.

EXAMPLE 1

@@ Whether or not the current time is after the current executor votes and before all voters complete voting.

(**after this** voter **did** vote) **and** (**before all** voter **did** vote)

EXAMPLE 2

@@ Whether or not the time point when the current executor votes is after the entry-into-force time of contract.

**this** voter **did** vote **is after** effective_date

——Bounded time predicate can be used to evaluate whether or not the current time is within a specific time boundary before (or after) a given base time. The syntax of bounded time predicate is shown as

***BoundedTimePredicate*** **::=** (**within**)? boundary (**before|after**) baseTime

where

| | |
|---|---|
| boundary | is a time variable or constant; |
| baseTime | is a time point expression. |

EXAMPLE 3

@@ Whether or not the current time is within three days before the end of the auction.

**within** 3 day **before** auctionEnd

EXAMPLE 4

@@ Whether or not the current time is more than three days before the end of the auction.

3 day **before** auctionEnd

## 7.10 Additional information

Additional information can define necessary supplementary information, including entity's attribute, contract object, the property and signature of guarantor, additional term, program variable, and the declaration of data structure, in smart legal contract. The syntax is shown as

***Additions*** ::= *field+* **|** (**addition** Dname{*field +*})

NOTE Additional information can be placed in any position of smart legal contract.

NOTE The reference of additional information includes two forms, (Cname::)? attribute or Dname::attribute, where Cname is the contract title for referencing a field without Dname.

EXAMPLE

@@ To define the highest auction amount and the stop time of the auction.

highestPrice: **Money**
biddingStopTime: **Date**

## 7.11 Contract conclusion

Contract conclusion can introduce representations as mutually agreeable statements of fact in entering into the contract. The electronic signature of the contract must be used to prove the conclusion of the contract in smart legal contracts. The syntax is shown as

@@ **Contract conclusion** : (the statement of all parties)?

```
Signs ::= Contract conclusion : ( The statement of all parties.)?
{   Signature of party Pname :
        {    printedName:  String,
             signature:    String,
             date:         Date
        },+
 }
```

EXAMPLE

**Contract conclusion:** This contract may not be modified in any manner unless in writing and signed by both parties. This document and any attachments hereto constitute the entire agreement between the parties. This Contract shall be binding upon the parties, their successors and assigns. By signing this agreement, all parties agree to the terms as described above. Each of parties will receive one copy of this agreement, and will be responsible for upholding its terms. Both parties agree with conversion from this contract to computer programs on smart contract platform, and approve that the programs' implementation has the same legal effect.

**Signature of party** auctioneer：

```
{     printedName：Yao San,
      signature：   0x2319…8DE393,
      date：        2020/7/12
}
```

## Annex A
(informative)

## Examples of smart legal contract and smart contract

### A.1 Example 1 of smart legal contract

EXAMPLE A smart legal contract corresponding to online auction contract.

```
@@ Here is an online auction contract
contract SimpleAuction{
    @@ Party A: Auctioneer, registered information includes:
        User account: 0x712379218...C4E80.
    party auctioneer{
        account : 0x712379218...C4E80
    }

    @@ Party B: Bidder, considered as a group, whose registered information includes:
        User account: [0x93A8BCD...793968, 0x48BD38... 92AC93];
        The cumulative value of previous bids: Money.
    party group bidders{
        account : [0x93A8BCD...793968, 0x48BD38...92AC93]
        amount : Money
    }

    @@ Additional information including: current highest bid, highest bidder, and end time of
bidding.
    highestPrice : Money
    highestBidder : bidders
    biddingStopTime : Date

    @@ Bidding goods: The auctioneer needs to provide its name, quantity and other relevant
        information of the auction.
    asset good{
        info{
            name : Name
            quantity : Integer
            price : Money
            package : String
        }
    }

    @@ Term No.1: The auctioneer can initiate a bid, and the current maximum price shall be the
        reserve price entered by the auctioneer after the action is executed, meanwhile the ending
        time shall be the current time plus the predefined duration of the bid.
    term no1 : auctioneer can StartBidding(reservePrice : Money, auctionDuration : Date)
        when before auctioneer did StartBidding
        where highestPrice = reservePrice and biddingStopTime = auctionDuration + now.

    @@ Term No.2: Bidders can place bids after the auctioneer initiates the auction until the end of
        the auction, and the bid is successful if the bid is greater than the highest price currently
        given.
    term no2 : bidders can Bid
        when after auctioneer did StartBidding and before biddingStopTime
        while deposit value > highestPrice
        where highestPrice = value and highestBidder = this bidder and
```

```
                    this bidder::amount = this bidder::origin amount + value.

    @@ Term No.3_1: If the bidder is not the highest bidder and has balance in his contract account,
          he can retrieve all previous bids, after which the bidder's deposit will be cleared.
    term no3_1 : bidders can WithdrawOverbidMoney
         when this bidder::amount > 0 and this bidder isn't highestBidder
         while withdraw this bidder::amount
         where this bidder::amount = 0.

    @@ Term No.3_2: If the bidder is the current highest bidder and has the previous failed bids in
          his contract account, he can retrieve the invalid bids and be registered as the current highest
          bidder.
    term no3_2 : bidders can WithdrawOverbidMoney
         when this bidder::amount > highestPrice and this bidder is highestBidder
         while withdraw this bidder::amount - highestPrice
         where this bidder::amount = highestPrice.

    @@ Term No.4: The auctioneer can collect the auction price at the end of the bidding.
    term no4 : auctioneer can StopBidding
         when after biddingStopTime and before auctioneer did StopBidding
         while withdraw highestPrice.

     Arbitration term : Any controversy or claim arising out of or relating to this contract, or the
     breach thereof, shall be settled by arbitration administered by institution :
     BeijingInternetCourt.

     Contract conclusion: This contract may not be modified in any manner unless in writing and
     signed by both parties. This document and any attachments hereto constitute the entire
     agreement between the parties. This Contract shall be binding upon the parties, their successors
     and assigns. By signing this agreement, all parties agree to the terms as described above. Each of
     parties will receive one copy of this agreement, and will be responsible for upholding its terms.
     Both parties agree with conversion from this contract to computer programs on smart contract
     platform, and approve that the programs' implementation has the same legal effect.

          Signature of party auctioneer:
          {       printedName:  Yao San,
                  signature:      0x23198de...393,
                  date:           2020/7/12
          }
          Signature of party bidders:
          {       printedName:  Wan Liu,
                  signature:      0x877238...201,
                  date:           2020/7/12
          }
          {       printedName:  Yuan Qin,
                  signature:      0x9340593...495,
                  date:           2020/7/12
          }
}
```

## A.2 Example 2 of smart legal contract

EXAMPLE A smart legal contract corresponding to residential tenancy contract.

```
@@ Here is a residential tenancy contract
contract HouseLease{
    @@ Party A: Landlord, registered information includes: User account: 0x82384a68...90e72.
    party Landlord{
        account : 0x82384a68…90e72
    }
    @@ Party B: Tenant, registered information includes: User account: 0x9845a6b...73c4e.
    party Tenant{
        account : 0x9845a6b…73c4e
    }
    @@ Additional information including: Landlord's deposit, Tenant's deposit, housing rent, total
        rent, start time of contract, end time of contract, time to pay rent, rental payment period,
        etc.
    addition infos {
        renterBail:Money
        renantBail:Money
        rental:Money
        totalRental:Money
        startLeasingTime:Date
        endLeasingTime:Date
        payDate:Date
        payDuration:Date
    }

    @@ House asset: Landlord should provide the specific house’s information, such as the property
number, address, area, usage, price, and the transferred house’s ownership.
    asset House{
        info{
            ownershipNumber: Integer
            location: Address
            area: Integer
            usage: String
            price: Money
        }
        right{
            houseUseright : useRight
        }
    }
    @@ Term No.1: The landlord can register the house information while he pays the pledge.
    term term1: Landlord can registerHouse
        while deposit infos::renterBail.

    @@Term No.2: The tenant can confirm the lease by paying the tenant's pledge after the landlord
        registers the premises. After the action is executed, it is required to automatically record
        the current time as the start time of the contract, and update the end time, the time to pay
        next rent, and the rental payment period.
    term term2: Tenant can confirmLease
        when after Landlord did registerHouse
        while deposit infos::tenantBail
        where infos::startLeasingTime = now and
             infos::endLeasingTime = endLeasingDuration +now
             and infos::payDate = payDuration+now and infos::payDuration = payDuration.
    @@ Term No.3: The landlord must transfer the right of use to the tenant within 7 days after the
        tenant confirms the lease.
    term term3: Landlord must transferHouse
        when within 7 day after Tenant did confirmLease
        while deposit $ houseUseright of house.

    @@ Term No.4: The tenant must pay the rent before the time to pay next rent and after the
        landlord confirms the transfer of house's right to use.
    term term4: Tenant must payRent
        when before infos :: payDate and after Landlord did transferHouse
        while deposit infos ::rental.

    @@ Term No.5: The tenant shall checks out and transfer the right of use back to the landlord after
        the contract expires and before the landlord checks the rent.
    term term5: Tenant must returnHouse
        when after infos :: endLeasingTime and before Landlord did checkHouse
        while transfer $ houseUseright of house to Landlord.

    @@ Term No.6: The landlord can inspect the house after the tenant checks out.
    term term6: Landlord can checkHouse
        when after Tenant did returnHouse.

    @@ Term No.7: The landlord can withdraw the full rent within 15 days after the house inspection.
    term term7: Landlord can collectRent
        when within 15 day after Landlord did checkHouse
        while withdraw infos::rental.

    @@ Term No.8: The landlord can withdraw the landlord's pledge within 15 days after the house
        inspection.
    term term8: Landlord can collectBail
        when within 15 day after Landlord did checkHouse
        while withdraw infos :: renterBail.

    @@ Term No.9: The tenant can withdraw the tenant's pledge within 15 days after the house
        inspection.
    term term9: Tenant can collectBail
        when 15 day after Landlord did checkHouse
        while withdraw infos :: tenantBail.
   Arbitration term : Any controversy or claim arising out of or relating to this contract, or the
   breach thereof, shall be settled by arbitration administered by institution : BeijingInternetCourt.
   Contract conclusion: This contract may not be modified in any manner unless in writing and
   signed by both parties. This document and any attachments hereto constitute the entire agreement
   between the parties. This Contract shall be binding upon the parties, their successors and assigns.
   By signing this agreement, all parties agree to the terms as described above. Each of parties will
   receive one copy of this agreement, and will be responsible for upholding its terms. Both parties
   agree with conversion from this contract to computer programs on smart contract platform, and
   approve that the programs' implementation has the same legal effect.
        Signature of party Landlord:
        {  printedName:  Mike Micheal,
           signature:     0x9045f7a...80d4,
           date:          2020/8/20
        }
        Signature of party Tenant:
        {  printedName:  Shuang Jiang,
           signature:     0x46b9d3e...a983,
           date:          2020/8/20
        }
}
```

## A.3 Example of smart contract

Example: The smart contract written by the language Solidity corresponding to the smart legal contract in A.1.

```
pragma solidity >=0.5.0 <0.6.0;

import "./bidders.sol";
import "./auctioneer.sol";

contract SimpleAuction {

  biddersT bidders;
  auctioneerT auctioneer;

  uint highestPrice;
  address highestBidder;
      uint biddingStopTime;


  constructor() public{
    bidders = new biddersT();
    auctioneer = new auctioneerT();
    auctioneer.regist(msg.sender);
          auctioneer.name = "Yao San";
          auctioneer.signature = "0x23198de…393";
          auctioneer.signDate = 2020/7/12;
          bidders.add("Wan Liu","0x877238…201",2020/7/12);
          bidders.add("Yuan Qin", "0x9340593…495", 2020/7/12);
  }

  modifier onlybidders{
    require(bidders.contains(msg.sender));
    _;
  }

  modifier onlyauctioneer{
    require(auctioneer.getAddress() == msg.sender);
    _;
  }

  function StartBidding(uint reservePrice, uint auctionDuration) onlyauctioneer() public {
    //RECORD
    auctioneer.StartBiddingDone();
    //USER CODE HERE
    highestPrice = reservePrice;
    biddingStopTime = auctionDuration + now;
    //CHECK
    assert(highestPrice == reservePrice && biddingStopTime == auctionDuration + now);
  }

  function Bid() public payable {
    if(!bidders.contains(msg.sender))
      bidders.add(msg.sender);
    //REQUIRE
    require(now > auctioneer.StartBiddingTime() && now < biddingStopTime);
    require(msg.value > highestPrice);
    uint this_bidder_Ori_amount = bidders.getamount(msg.sender);
    //USER CODE HERE
    highestPrice = msg.value;
    highestBidder = msg.sender;
    bidders.setamount(msg.sender,bidders.getamount(msg.sender) + msg.value);
    //CHECK
    assert(highestPrice == msg.value && highestBidder == msg.sender &&
    bidders.getamount(msg.sender) == this_bidder_Ori_amount + msg.value);
  }

  function WithdrawOverbidMoney() onlybidders() public payable {
    //REQUIRE
    if(msg.sender != highestBidder && bidders.getamount(msg.sender) > 0){
      //USER CODE HERE
      msg.sender.transfer(bidders.getamount(msg.sender));
      bidders.setamount(msg.sender, 0);
      //CHECK
      assert(bidders.getamount(msg.sender) == 0);
    }
    //REQUIRE
    else if(msg.sender == highestBidder && bidders.getamount(msg.sender) > highestPrice) {
      //USER CODE HERE
      msg.sender.transfer(bidders.getamount(msg.sender) -  highestPrice);
      bidders.setamount(msg.sender, highestPrice);
      //CHECK
      assert(bidders.getamount(msg.sender) == highestPrice);
    }
    else{
      revert();
    }
  }

  function CollectPayment() onlyauctioneer() public payable {
    //REQUIRE
    require(now > biddingStopTime && now < auctioneer.CollectPaymentTime());
    //RECORD
    auctioneer.CollectPaymentDone();
    //USER CODE HERE
    msg.sender.transfer(highestPrice);
  }
}
```

# Annex B
(informative)

## Signing process of smart legal contract

The signing process of smart legal contract includes generation, deployment, producing signature, and preserving evidence of the corresponding smart contract codes. Once the negotiation and composition of smart legal contract are accomplished by parties, it can be converted into the executable codes by translator, further the codes are deployed in the smart contract platform and signed by parties for confirmation. The signed smart legal contract and its codes can be acquired and reviewed at any time.

The storage form of smart legal contract includes itself, its evolution, and information defined by users, as well as basic contract attributes and status, such as blockchain address (denoted as Address in Fig.2), signature (Sign(party)) and timestamp (Time(party)) of contracting party.

As shown in Fig.2 of Chapter 4, the signing process of smart legal contract is described as follows:

a) As the smart legal contract language is used to write the contract A, the default is all attribute values are null in A.

b) When the executable smart contract B is generated from A, the operator shall sign it if the manual operations are necessary.

c) The Address is generated as its unique identification during the deployment of B, after that, the party (e.g. Party I) appends Time(Party I) and Sign(Part I) to record the signing time and the signature on both A and B.

d) For the query and use of signed contract, the contract attributes can be updated at any time into the smart contract A to generate a browsable version.

## Bibliography

[1] GB/T 35285: 2017, *Information security technology—Public key infrastructure—Technical requirements of reliable electronic signature generation and verification based on digital certificates*

[2] YD/T 3204: 2016, *Reference architecture of eID*

[3] *Civil Code of the People's Republic of China*

[4] *Law of the People's Republic of China on Electronic Signatures*