ICS 点击此处添加 ICS 号
CCS 点击此处添加 CCS 号

团 体 标 准

T/CIIA XXXX—XXXX

分布式数字身份

第 1 部分：技术框架

Decentralized identifier-
part 1: Technique Architecture

（征求意见稿）

在提交反馈意见时，请将您知道的相关专利连同支持性文件一并附上。

XXXX－XX－XX 发布　　XXXX－XX－XX 实施

中国信息协会　　发 布

目　　录

前　言

本文件按照GB/T 1.1-2020《标准化工作导则 第1部分:标准化文件结构和起草规则》的规定起草。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别专利的责任。

本文件由中国信息协会提出并归口。

本文件起草单位：中国民航信息网络股份有限公司、北京航空航天大学、广州民航信息技术有限公司。

本文件起草人：王刚、刘懿中、范修伟、郭彪、刘增智、郭宗宝、蔡修明、冉悦、彭跟耀、黄晓亮、王航、甘国操、张笑天、彭俊文、云雄

引　言

随着下一代互联网的发展，分布式数字身份技术作为基础技术，正逐步在各行业内落地应用。在此过程中，需要一套适用于信息化行业的分布式数字身份技术规范，形成信息化行业统一的分布式数字身份标准，降低系统理解难度及接入复杂度，进而降低第三方服务商接入成本，从而提升信息化行业旅客服务能力。

本系列团体标准主要用于指导信息化各行业建设分布式数字身份系统，健全并完善行业标准体系建设。本系列标准主要包含以下几部分：

第 1 部分：技术框架：描述信息化行业分布式数字身份系统的专业术语、缩略语、总体架构，工作流程等。

第 2 部分：可信机构通用服务规范：规范行业可信机构建设标准，交互模型、跨行业可信机构数据交互标准等。

第 3 部分：可验证凭证术语表：本规范定义支持语义网的行业业术语表，包含凭证类型、状态，凭证模板等内容。

第 4 部分：民航业分布式数字身份规范：规范民航业分布式数字身份体系建设细则，包含民航业可信机构管理模式、民航业可验证凭证数据模型、民航业术语表等。

本部分为该标准的第 1 部分。

分布式数字身份
第1部分：技术框架

1 范围

本文件规定了信息化行业分布式数字身份建设相关规定和要求，行业内和行业间关键凭证语义模型定义和校验规则。

本文件适用于相关行业分布式数字身份建设参考，行业内或跨行业凭证数据交换和验证。

2 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

3 术语和定义

下列术语和定义适用于本文件。

3.1 分布式账本 Distributed ledger (DLT)

一种非中心化的事件记录系统。这些系统建立的信任机制，使参与者足以信赖其他参与者记录的数据来做出决策。它们通常使用区块链作为分布式数据库，账本交易节点使用共识协议来确认加密签名交易的顺序并形成一个交易区块。随着时间的推移，交易区块依次链接形成交易账本，这种账本生成机制可确保交易数据有效性和不可篡改性。

3.2 分布式数字身份 Decentralized identifier(DID)

3.3 分布式数字身份文档 DID document

一组描述DID主体的数据，及其相关机制，如公钥验证机制，用于验证DID主体或DID委托人与DID的关联。一个DID文档可能有一个或多个不同的表示形式。

3.4 分布式数字身份主体 DID subject

由DID标识并由DID文档描述的实体。任何事物都可以是DID主体：人、团体、组织、物理事物、数字事物、逻辑事物等。

3.5 分布式数字身份控制者 DID controller

能够对DID文档进行更改的实体。一个DID可能有多个DID控制者。DID控制者可以由DID文档顶层的可选控制者属性表示。DID控制者也可以是DID主体。

3.6 分布式数字身份委托人 DID delegate

一个身份使用权受控的实体，可由DID控制者通过DID文档向其授予使用与DID相关联的验证方法。

例如：一个父母控制孩子的DID文档，可能允许孩子使用其个人设备以进行认证。在这种情况下，孩子是DID代理。孩子的个人设备将包含私钥加密材料，使孩子能够使用DID进行认证。然而，未经父母许可，孩子可能无法添加其他个人设备。

3.7 分布式数字身份方案 DID scheme

去中心化标识符的正式语法。通用DID方案以前缀“did:”开头。每个DID方法规范都定义了一个特定的DID方案，该方案与该特定DID方法一起工作,DID method为did的第一部分。

例如：假设did为：did:example:abc，则DID scheme为“did”。

3.8 分布式数字身份方法 DID method

实施特定 DID 方法方案的定义。DID 方法由 DID 方法规范定义，该规范规定了创建、解析、更新和停用 DID 和 DID 文档的精确操作,DID method为did的第二部分。

例如：假设did为：did:example:abc，则DID method为“example”。

3.9 分布式数字身份统一资源定位地址 DID URL

用于定位分布式数字身份某项资源的路径表达式。里面可以包括可选的DID路径（带前导“/”字符）、可选的DID查询（带前导“？”字符）和可选的DID片段（带前导“#”字符）。

例如：did:example:123?service=files&relativeRef=/resume.pdf

3.10 分布式数字身份路径 DID path

DID URL的一部分，以第一个正斜杠（/）字符开头并包括该字符，以问号（？）字符、片段哈希符号（#）字符或DID URL结尾。

例如：did:example:123/path

3.11 分布式数字身份查询 DID query

DID URL中第一个问号字符（？）后面的部分。

例如：did:example:123?versionId=1

3.12 分布式数字身份片段 DID fragment

DID URL中第一个哈希符号字符（#）后面的部分。

例如：did:example:123#public-key-0

3.13 分布式数字身份统一资源定位地址解引用 DID URL dereferencing

以DID URL和一组输入元数据作为输入并返回资源的过程。此资源可能是DID文档加上其他元数据，可能是DID文档中包含的次级资源，也可能是DID文档所引用的外部资源。该过程使用DID解析来获取DID URL中包含的DID所指示的DID文档。然后，解引用过程可以在DID文档上执行额外的处理，以返回DID URL所指示的解引用资源。

3.14 分布式数字身份统一资源定位地址解引用器 DID URL dereferencer

为给定的DID URL或DID文档执行DID URL解引用功能的软件或硬件系统。

3.15 分布式数字身份解析 DID resolution

以DID和一组解析选项作为输入，并以符合规范的表示形式返回DID文档和附加元数据的过程。此过程依赖于适用DID方法的“读取”操作。

3.16 分布式数字身份解析器 DID resolver

一个执行DID解析功能的软件或硬件组件，它接受一个DID作为输入并产出一个符合规范的DID文档作为输出。

3.17 公钥描述 public key description

一种DID文档中的数据对象，包含使用公钥或验证密钥所需的所有元数据。

3.18 服务 services

通过一个或多个服务端点与DID主体或相关实体进行通信或交互的手段。示例包括发现服务、代理服务、社交网络服务、文件存储服务和可验证的凭据存储库服务。

3.19 服务端点 service endpoint

代表DID主体运行服务的网络地址，如HTTP URL。

3.20 统一资源标识符 Uniform Resource Identifier (URI)

为万维网上所有资源的标准标识符格式。DID就是一种URI方案。

3.21 可验证数据注册系统 verifiable data registry

一种便于创建、验证、更新或停用分布式数字身份和DID文档的系统，例如分布式账本、去中心化文件系统、任何类型的数据库、对等网络和其他形式的可信数据存储。

3.22 验证方法 verification method

基于非对称公私钥体系的验证方法，通过传入一组参数来验证“证明”。例如，公钥可以用于验证数字签名。

3.23 验证关系 verification relationship

DID主体与验证方法之间关系的表达。比如DID文档里面关于验证DID主体真实性的相关字段。

3.24 可验证凭证 verifiable credential

由W3C可验证凭证规范所定义的一种可加密验证的数字证书标准数据模型和表示格式。

3.25 发证方 issuer

可以给一个或多个DID主体创建并颁发可验证凭证的实体。

3.26 持证方 holder

拥有一个或多个可验证凭证的实体。持证方将可验证凭证存储在数据库中。

3.27 验证方 verifier

验证DID或者可验证凭证合法性的实体。

4 缩略语

以下缩略语使用于本文件。

DID：Decentralized identifier，分布式数字身份。

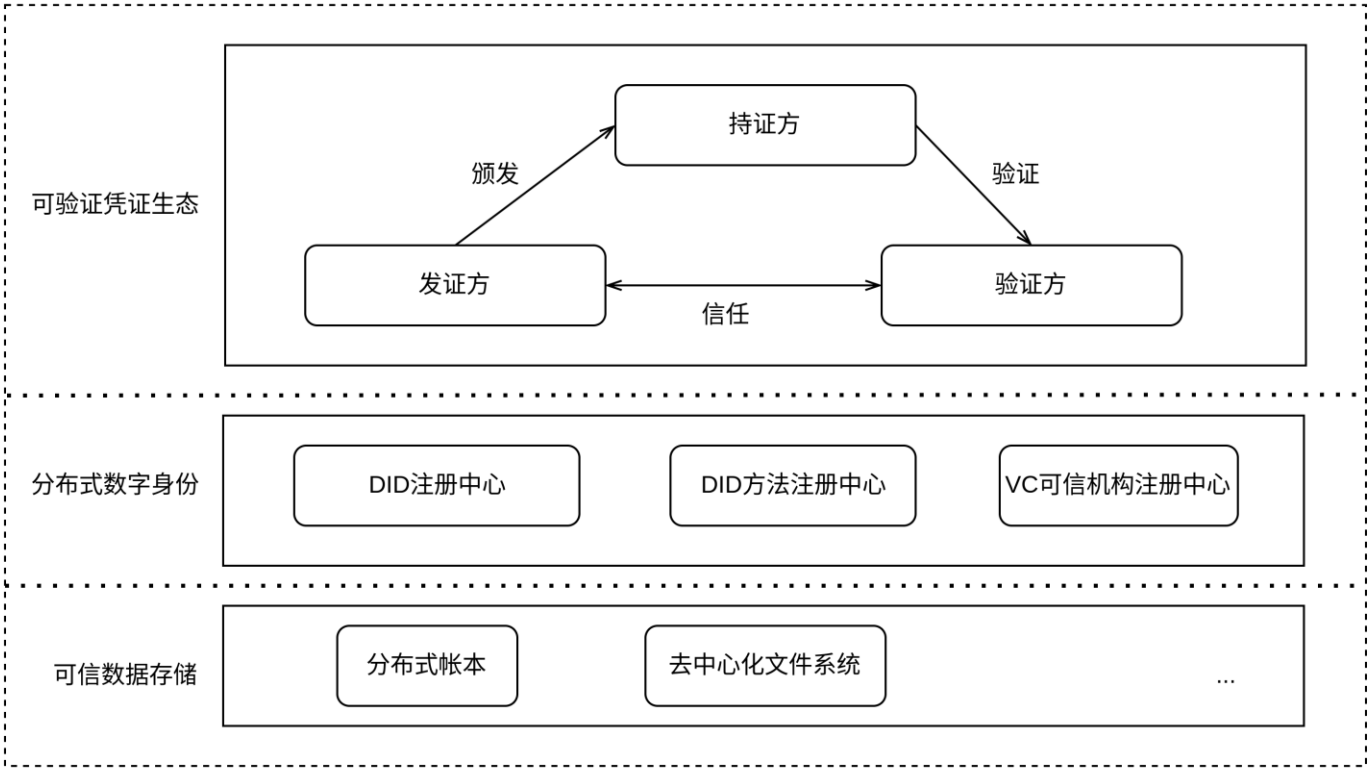
DLT：Distributed ledger，分布式账本。

DIF：Decentralized Identity Foundation，全球分布式数字身份社区。

DPKI: Distributed PKI，分布式公钥基础设施。

VC：Verifiable Credential，可验证凭证。

5 总体架构



5.1 可信数据存储

可信数据存储为支撑分布式数字身份系统数据（如 DID 文档、可验证凭证、可验证展示）存储的基础设施。任何去中心化的存储设施、系统或者平台都满足本层的要求，如分布式账本（区块链）、去中心化文件系统（去中心化存储）、任何类型的数据库、对等网络以及其他形式。对于经典的技术规范如下：

1. 分布式帐本

采用分布式帐本做为可信数据存储基础设施，在分布式数字身份中，可提供安全、可信和去中心化的身份管理解决方案。其关键作用包括：

——身份认证和验证：分布式帐本可以可以作为身份认证和验证的基础设施。通过在分布式帐本上注册用户的数字身份，可以实现去中心化的身份验证，使用户能够安全地管理自己的身份信息，并在需要时提供可验证的身份证明。

——数据安全：分布式帐本提供了一种安全的数据存储和传输机制。用户的身份信息可以被加密并安全地存储在分布式帐本链上，只有 DID 主题或其认定的 DID 控制器才能修改这些信息，从而保护用户的隐私和安全。

——去中心化身份管理：分布式帐本技术可以实现去中心化的身份管理，消除了传统身份管理系统中的单点故障和单点控制。每个用户都可以拥有自己的身份标识，并通过区块链上的智能合约来管理和

验证身份信息，从而实现更加安全和可信的身份管理。

——跨组织跨行业身份验证：分布式帐本可以作为跨组织跨行业身份验证的解决方案，通过智能合约和跨链技术，不同组织不同行业之间可以安全地验证用户的DID来开展业务，从而促进跨组织跨行业的业务合作。

2. 去中心化文件系统

采用中心化文件系统做为可信数据存储基础设施，在分布式数字身份中，可提供了安全、可靠和灵活的数据存储解决方案。其主要作用包括：

——安全性和可靠性：采用分布式网络中的节点集群来存储数据，可容忍一定数量的节点故障，提高了数据的安全性和可靠性。

——可扩展性和弹性：可以根据需求动态地添加或移除存储节点，以满足不断增长的数据存储需求，提高系统的可扩展性和弹性。

5.2 分布式数字身份

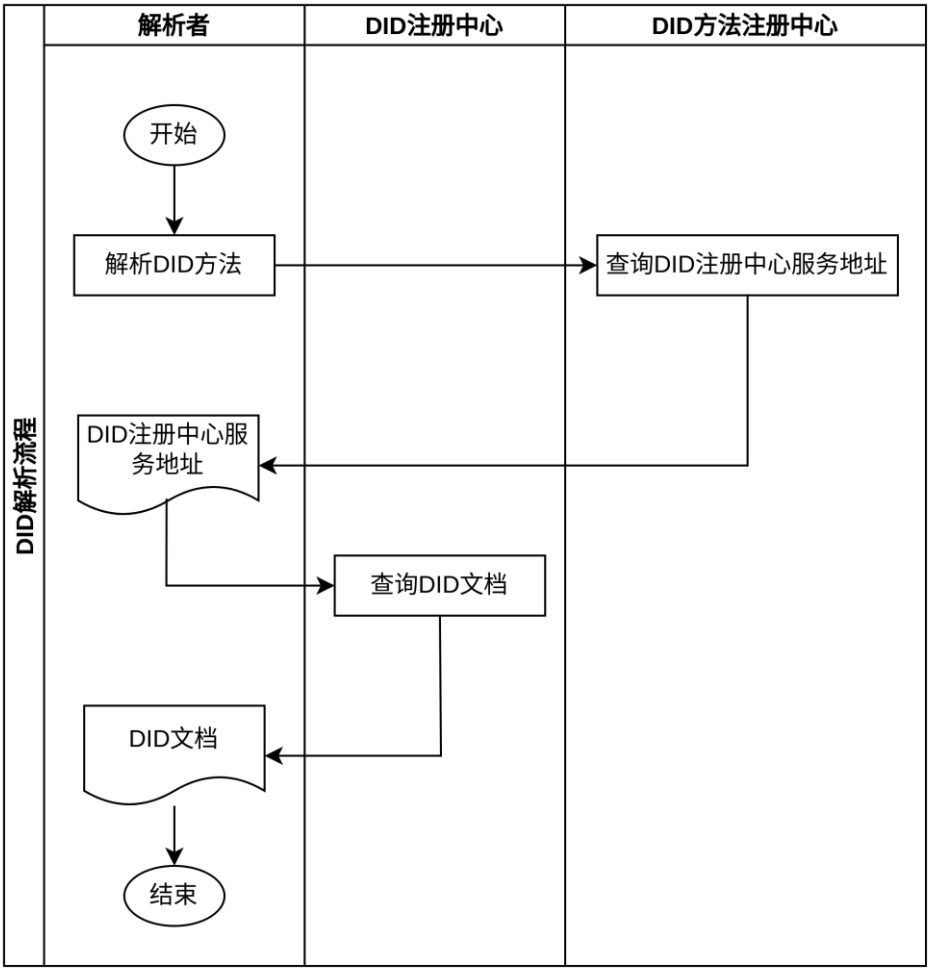
分布式数字身份层负责提供分布式数字身份认证的去中心化PKI网络能力，其核心包括分DID注册中心、DID方法注册中心和VC可信机构注册中心。

5.2.1 DID注册中心

DID注册中心关联一个或者多几DID方法，并提供对应的本文档中对DID规范的服务。这些服务至少包含DID注册服务和DID解析服务。

1. DID注册服务
   提供包括DID注册、DID文档保存、DID文档查询和修改功能。
2. DID解析服务
   提供DID解析器功能，通过将DID作为输入以获取DID文档或其子集来执行DID解析功能。

DID解析服务是DID操作中最核心的流程，接收一个DID作为输入，并输出DID文档，具体的交互流程如下：



参与角色：
1. 解析者：知道DID并希望获取DID文档的主体。
2. DID注册中心：提供DID操作服务组织。
3. DID方法注册中心：提供DID方法管理服务的组织。
解析流程：
1. 解析者根据DID的标识符规范（参考附录8.1 DID标识符）解析出DID方法；
2. 解析者将DID方法作为请求参数，向DID方法注册中心公开的服务地址请求查询该DID方法对应的服务地址；
3. 解析者将DID作为请求参数，向步骤2中获取的服务地址查询DID文档。

解析流程中任何一个步骤失败将导致整个解析流程失败。解析的DID文档以JSON数据格式返回给解析者。

5.2.2 DID 方法注册中心

DID方法注册中心为DID注册中心提供全局的DID方法申请、审核服务，并提供公开查询DID方法对应服务入口地址服务，该公开的服务地址必须是固定的且无访问限制。

5.2.3 VC 可信机构注册中心

VC可信机构注册中心为可验证凭证层提供可信机构列表管理服务，以建立验证方与发证方之间的信任关系。

VC可信机构注册中心同时负责管理和维护可信机构颁发颁发的可验证凭证类型及凭证模板，且公开的提供凭证类型及凭证模板的查询服务。这些查询服务地址必须是固定的且无访问限制。

5.3 可验证凭证生态

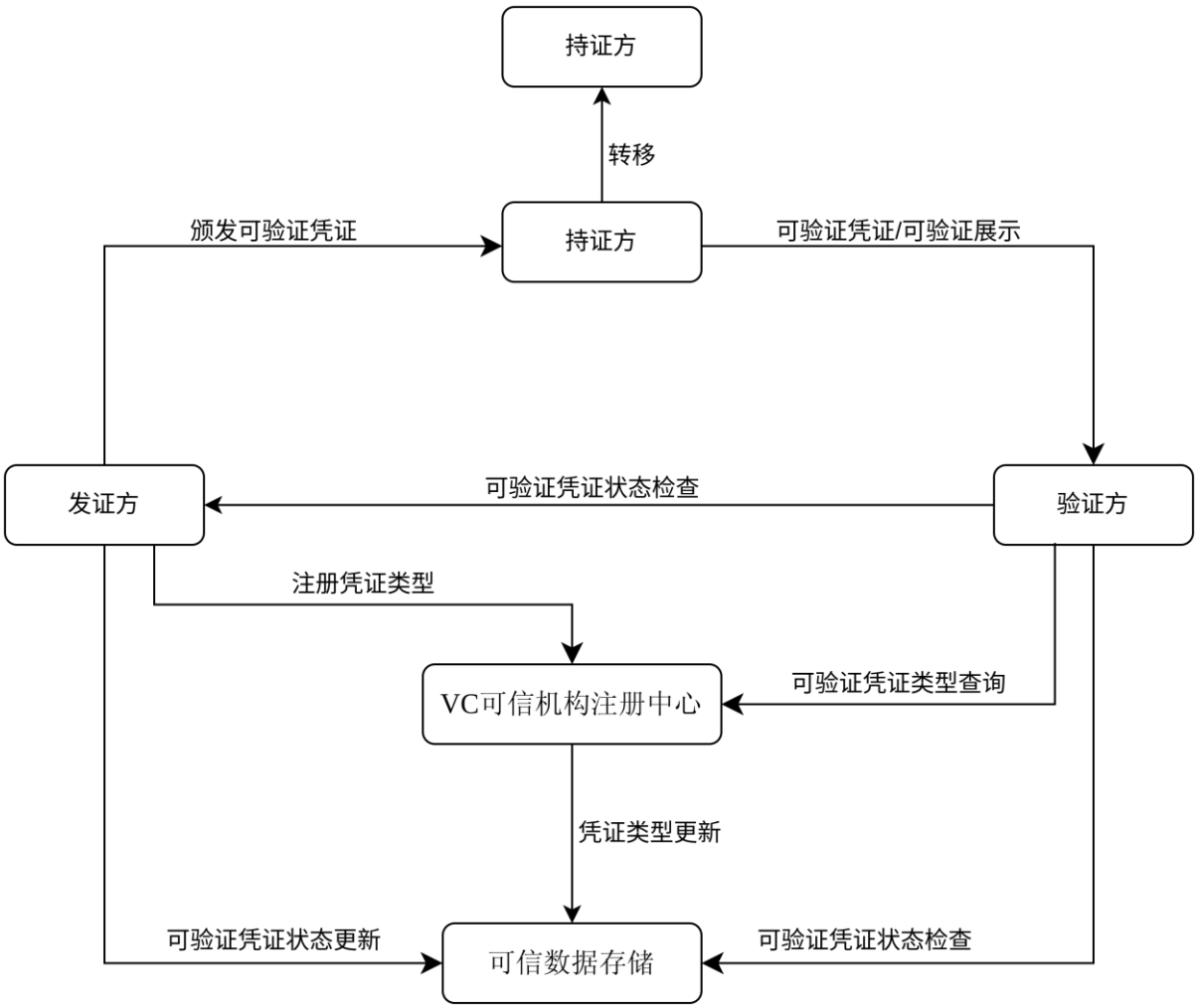
可验证任证层为参方间相建立安全身份认证与数据交换，该层处理可验证凭证在各个生态参与方之间的交互关系。

5.3.1 可信交换

可验证凭证的可信交换涉及持证方、发证方和验证方。在凭证申请阶段持证方可根据其业务需求向发证方发起申请，发证方对其身份验证后向其颁发可验证凭证。验证方可根据其业务需求对持证方的可验证凭证及关联DID发起验证。

5.3.2 交互关系

可验证凭证的生态参与者包含发证方、持证方、验证方、VC可信机构注册中心、可信数据存储。其关系如下图：



1. 发证者向持证者签发可验证凭证。凭证的签发必须优先于其他所有相关操作。
2. 持证者可将其持有的一个或多个可验证凭证转移给其他持证者。

3. 持证者可向验证者出示一个或多个可验证凭证，这些凭证可封装在可验证展示（Verifiable Presentation）中（此为可选操作）。
4. 验证者需验证所提交的可验证展示及可验证凭证的真实性，包括检查凭证状态（如是否已被撤销）。
5. 发证者可废弃已签发的可验证凭证。
6. 持证者可删除其持有的可验证凭证。
7. 可验证凭证的状态可以是有效、过期和废弃。
8. 颁发可验证凭证之后即有效，发布者也可主动废弃已颁发的可验证凭证。
9. 可验证凭证的状态由发证方维护，可选地保存到可信数据存储中，可信数据存储必须是可验证凭证参与方都可访问。

标准的交互序列：

1. 发证者向持证者签发凭证
2. 持证者向验证者出示凭证
3. 验证者完成凭证验证

可验证凭证的转移：

发证者向持证者签发凭证以及持证者向验证者出示凭证的方式有以下两种方式：

1. 离线转移：采用存储介质或者网络文件传输的方式转移，过程确保数据传输的安全。
2. 可信数据存储：发证者将颁发的凭证保存到可信数据存储设施中，发证者或者持证人只将获取凭证标识交给接收对象；持证者和验证者都根据获取凭证标识从可信数据存储中查询凭证。

6 建议实施技术

6.1 可信数据存储

可信数据存储应该由行业组织共同建设。可采用但不限以下技术建设：

1. 区块链：

区块链是分布式账本最佳的实现，采用区块链作为可信数据存储时，必须采用多方参与的区块链网络实施，支持业务参与方或监管单位参与共识。

2. IPFS：

IPFS 是去中心化的分布式文件存储与传输协议，能够实现更高效、安全、持久化的数据存储访问。采用 IPFS 作为可信数据存储，可在业务参与方访问内共享 DID 文档或其他静态数据。

6.2 分布式数字身份

6.2.1 DID 服务

1. DID文档数据要求应遵循[附录 9.2 分布式数字身份文档]要求。

2. 分布式数字身份控制为行业权威组织或者机构关联DID。
3. 行业可存在多个DID注册中心，但必须在DID方法注册中心完成注册，DID注册中心必须依据附录规范提供DID注册和解析服务，所有DID账户和文档，由行业权威组织或者机构管理和控制。

6.2.2 VC 可信机构注册中心

VC 可信机构注册中心由行业共同管理或由行业权威机构提供管理服务，可信机构必须在其 DID 文档中维护对应的 Service 项，申明其提供的服务类型和 Endpoint，由行业分别统一提供管理并以静态资源方式统一发布到固定的 URL，提供开放、稳定的查询服务。

6.2.3 可信机构

VC 可信机构注册中心管理的可信机构全生命周期应在可信数据存储设施中记录，且能够公开地查询到。

VC 可信机构注册中心需提供人类可读的可信机构查询服务，且查询服务应是公开的 URL，访问需不受限制，同时需在相同的 URL 提供机器可读的文档（如 JSON-LD 格式，RDF 格式）。

VC 可信机构注册中心应维护人类可读和机器可读的词汇表（参考 6.3.2 词汇表定义要求），且公开到固定的 URL，访问不受限制。

可信机构应至少包含以下信息：

- 可信机构名称
- 可信机构 DID
- 可信机构类型
- 可信机构可颁发 VC 类型

6.3 可验证凭证生态

6.3.1 凭证定义要求

1. 凭证必须符合【附录 7.4.1 可验证凭证数据模型】规范要求。
2. Type字段必须对VC类型应词汇表中定义的Type。
3. CredentialSchema字段必须定义，本标准涉及凭证均需提供JsonSchema，该字段必须为type类型所对应凭证所规定的Schema。
4. CredentialStatus 字段如存在，必须符合与Type匹配的Status定义以确定凭证状态。

6.3.2 词汇表定义要求

1. 标准所有凭证所使用类型，均需要定义词汇。
2. 词汇表需要提供人类可读文档（如HTML格式）和机器可读文档（如JSON-LD格式）
3. 词汇表由行业自行收集并以静态资源方式统一发布到固定的URL，提供开放、稳定的查询服务。
4. 词汇表必须支持版本化管理并体现到URL，所有发布版本均需保持历史版本信息以支持历史版本类型的识别和验证。
   例如：乘客类型（Passager）词汇发布URL为
   https://did.travelsky.com.cn/type/Passenger
   则该URL默认为最新版本文档
   https://did.travelsky.com.cn/type/Passenger/v1
   为V1版本文档。

6.3.3 凭证模板定义要求

1. 本标准涉及凭证必须提供凭证验证模板，以用于校验凭证数据的合规性。
2. 凭证模板以JSON Schema方式定义。
3. 凭证模板由行业自行收集整理后并并以静态资源方式统一发布到固定的URL，提供开放、稳定的查询服务，提供给验证方访问使用。
4. 凭证模板需支持版本化管理，所有发布版本均需保持历史版本信息以支持历史版本类型的识别和验证。
   例如：乘客凭证（PassagerType）凭证模板发布URL为
   https://did.travelsky.com.cn/schame/passage_type.json
   则该URL默认为最新版本文档
   https://did.travelsky.com.cn/schame/v1/passage_type.json
   为V1版本文档。

6.3.4 凭证状态定义要求

1. 本标准涉及凭证需要定义凭证状态时，应按【附录 7.4.1.9 状态】要求定义。
2. 凭证状态定义需要提供人类可读的文档（HTML）和机器可读可读文档（如JSON-LD格式）
3. 凭证状态定义由行业自行收集整理后并以静态资源方式统一发布到固定的URL，提供开放、稳定的查询服务。
4. 凭证状态定义需支持版本化管理，所有发布版本均需保持历史版本信息以支持历史版本类型的识别和验证。
   例如：客票凭证状态（TicketStatusList）发布URL为
   https://did.travelsky.com.cn/status/ticket_type
   则该URL默认为最新版本文档
   https://did.travelsky.com.cn/status/ticket_type/v1
   为V1版本文档。

7 附录

7.1 DID 标识符(DID Identifier)

DID标识符是一种符合 [RFC3986] 标准的统一资源标识符（URI）方案，是一个简单的文本字符串，由三部分组成：

1）“did” URI模式(scheme)标识符；

2）DID方法(did method)的标识符；

3）DID方法(did method)范围内的标识符，具有唯一性。

DID标识符的巴科斯范式（ABNF）定义如下，该定义采用了 [RFC5234] 中的语法以及针对字母字符（ALPHA）和数字字符（DIGIT）的相应定义。以下 ABNF 中未定义的所有其他规则名称均在 [RFC3986] 中有定义。所有的去中心化标识符(DID)都必须符合去中心化标识符(DID)语法的扩充巴科斯范式(ABNF)规则。

```
did                = "did:" method-name ":" method-specific-id
method-name        = 1*method-char
method-char        = %x61-7A / DIGIT
method-specific-id = *( *idchar ":" ) 1*idchar
idchar             = ALPHA / DIGIT / "." / "-" / "_" / pct-encoded
pct-encoded        = "%" HEXDIG HEXDIG
```

示例：

```
did:example:6ff07b8fa62e75be6e10f387451989d8
```

7.2 DID URL

DID URL 是特定资源的网络位置标识符。它可用于检索诸如 DID 主体的表示形式、验证方法、服务、DID 文档的特定部分或其他资源。

以下是使用［RFC5234］语法的 ABNF 定义。它基于 附录9.1 DID标识符 中定义的 “did” 方案构建。所有 DID URL都必须符合 DID URL 语法 ABNF 规则。

```
did-url = did path-abempty [ "?" query ] [ "#" fragment ]
```

path-abempty、query 和 fragment 在 [RFC3986] 中定义。

尽管根据 DID URL 语法规则可以使用分号（;）字符。为避免未来冲突，开发人员应避免使用该字符。

- **路径(**path-abempty)

DID 路径与通用 URI 路径相同，并且符合 RFC3986 中的对于 path-abempty 的 ABNF 规则。与 URI 一样，路径语义可以由 DID 方法指定，这反过来可能使 DID 控制器能够进一步细化这些语义。

- **查询(**query)

DID 查询与通用 URI 查询相同，并且符合 RFC 3986 中的对于 query 的 ABNF 规则。

- **片段(**fragment**)**

DID 片段的语法和语义与通用 URI 片段相同,并且符合 RFC3986 中的对于 fragment 的 ABNF 规则。DID 片段用作独立于方法的引用，指向 DID 文档或外部资源。

7.3 DID 文档（DID Document）

DID文档是一组描述DID主体的数据或者属性，其中包括诸如加密公钥之类的机制，DID 主体或 DID 代理人可利用这些机制来进行自我身份验证，并证明其与该 DID 的关联关系。

7.3.1 数据模型

DID 文档数据模型中的所有条目键都是字符串。所有条目值使用下表中的一种抽象数据类型表示，并且每种表示形式指定了每种数据类型的具体序列化格式。

数据类型	注意事项
映射	一个有限的有序键值对序列，没有键会出现两次。映射有时也被称为有序映射。符合INFRA规范中对于映射的定义。
列表	一个有限的有序项序列。符合INFRA规范中对于列表的定义。
集合	一个有限的有序项序列，其中不包含重复项，也被称为有序集。符合INFRA规范中对于集合的定义。

数据类型	注意事项
日期时间	一个日期和时间值。符合 XMLSCHEMA11-2 中对于日期时间的定义。
字符串	一个代码单元序列，通常用于表示人类可读的语言。符合INFRA规范中对于字符串的定义。
整数	一个没有小数部分的实数。符合RFC8259对于整数的定义。
双精度浮点数	一个通常用于近似任意实数的值。符合RFC8259对于整数的定义。
布尔值	一个取值为 true 或 false 的值。符合INFRA规范中对于布尔值的定义。
空值	一个用于表示没有值的值。符合INFRA规范中对于空值的定义。

7.3.1.1 预定义属性

本节定义了包含了本规范定义的预定义的DID文档属性的参考信息、预期值以及它们是否必需。

属性	是否必需	值约束
id	是	一个符合 附录 9.1 DID标识符 语法规则的字符串。
alsoKnownAs	否	一组符合 [RFC3986] 中 URI 规则的字符串。
controller	否	一个或一组符合 附录9.1 DID标识符 语法规则的字符串。
verificationMethod	否	一组符合验证方法属性规则的验证方法映射。
authentication	否	用于不同场景的验证关系。 一组符合验证方法属性规则的验证方法映射，或一组符合附录9.2 DID URL 语法规则的字符串。详细见相关附录9.3.1.2 验证关系。
assertionMethod	否	
keyAgreement	否	
capabilityInvocation	否	
capabilityDelegation	否	
service	否	一组符合服务属性规则的服务端点映射。

- verificationMethod（验证方法）

DID 文档可以表达验证方法,这些方法可用于对与 DID 主体或相关方的交互进行身份验证或授权。如加密公钥可以作为与数字签名相关的验证方法；在这种用法中，它验证签名者是否能够使用相关的加密私钥。验证方法可能需要多个参数。

属性	**是否必需**	**值约束**
id	是	一个符合附录9.2 DID URL 语法规则的字符串。
controller	是	一个符合附录9.1 DID标识符语法规则的字符串。
type	是	一个字符串，表示一种验证方法类型
publicKeyJwk	否	一个表示JSON Web Key且符合［RFC7517］的映射。该映射不能包含“d”或注册模板中描述的任何其他私有信息类成员。建议使用 JWKs［RFC7517］表示其公钥的验证方法将kid的值用作其片段标识符。建议 JWK 的kid值设置为公钥指纹[RFC7638]。一种验证参数。
publicKeyMultibase	否	一个符合［MULTIBASE］编码公钥的字符串。一种验证参数。

验证参数是应用验证方法的过程中使用的任何信息。验证方法的 type 用于确定其与这些过程的兼容性。验证参数属性包括 publicKeyJwk 或 publicKeyMultibase。加密套件规范负责指定验证方法的 type 及其相关的验证材料。禁止在一个验证方法中同时使用 publicKeyJwk 和 publicKeyMultibase 来表达密钥数据。

- service

属性	是否必需	值约束
id	是	一个符合［RFC3986］中 URI 规则的字符串。
type	是	一个字符串或一组字符串。
serviceEndpoint	是	一个符合［RFC3986］中 URI 规则的字符串、一个映射，或由

属性	是否必需	值约束
		一个或多个符合 [RFC3986] 中 URI 规则的字符串和 / 或映射组成的集合。

7.3.1.2 验证关系（Verification Relationship）

验证关系表达了 DID 主体与验证方法之间的关系。

不同的验证关系使相关的验证方法能够用于不同的目的。验证者有责任通过检查所使用的验证方法是否包含在 DID 文档的适当验证关系属性中，来确定验证尝试的有效性。

DID 主体与验证方法之间的验证关系在 DID 文档中是明确的。与特定验证关系无关的验证方法不能用于该验证关系。例如，authentication 属性值中的验证方法不能用于与 DID 主体进行密钥协商协议，进行此操作需要使用 keyAgreement 属性的值。

DID 文档不会通过验证关系来表达已撤销的密钥。如果引用的验证方法不在用于解引用它的最新 DID 文档中，则该验证方法被视为无效或已撤销。每个 DID 方法规范都应详细说明如何执行和跟踪撤销操作。

以下各节定义了几种有用的验证关系。DID 文档可以包含这些属性中的任何一个，或其他属性，以表达特定的验证关系。

- authentication 验证关系

authentication 验证关系用于指定 DID 主体预期的身份验证方式，关联值必须是一组一个或多个验证方法。每个验证方法可以是嵌入的或引用的。例如，登录网站或参与任何形式的挑战 - 响应协议。

如果身份验证成功，由 DID 方法或其他应用程序决定如何处理该信息。例如，特定的 DID 方法可能决定，作为 DID 控制器进行身份验证就足以更新或删除 DID 文档。另一种 DID 方法可能需要不同的密钥，或完全不同的验证方法，才能更新或删除 DID 文档，而不是使用用于身份验证的密钥。换句话说，身份验证检查之后的操作不在数据模型的范围内；DID 方法和应用程序应自行定义这些操作。

这对任何需要检查试图进行身份验证的实体是否确实提供了有效身份验证证明的身份验证验证者很有用。当验证者收到一些数据（以某种特定协议格式），其中包含为 “身份验证” 目的而创建的证明，并且该证明表明某个实体由 DID 标识时，验证者会检查该证明是否可以使用 DID 文档中 authentication 下列出的验证方法（例如公钥）进行验证。

请注意，DID 文档的 authentication 属性指示的验证方法只能用于对 DID 主体进行身份验证。要对不同的 DID 控制器进行身份验证，与 controller 的值相关联的实体需要使用其自己的 DID 文档和相关的 authentication 验证关系进行身份验证。

示例：

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ],
  "id": "did:example:123456789abcdefghi",
  ...
  "authentication": [
    // 此方法可用于对 did:...fghi 进行身份验证
    "did:example:123456789abcdefghi#keys-1",
    // 此方法仅被批准用于身份验证，不可用于任何其他证明目的，因此其完整描述在此处嵌入，而不是仅使用引用
    {
      "id": "did:example:123456789abcdefghi#keys-2",
      "type": "Ed25519VerificationKey2020",
      "controller": "did:example:123456789abcdefghi",
      "publicKeyMultibase": "zH3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
    }
  ],
  ...
}
```

- assertionMethod

assertionMethod 验证关系用于指定 DID 主体预期如何表达声明，例如为了颁发可验证凭证。其关联值必须是一组一个或多个验证方法。每个验证方法可以是嵌入的或引用的。

在验证者处理可验证凭证时，此属性很有用。在验证过程中，验证者会检查可验证凭证是否包含由 DID 主体创建的证明，方法是查看用于断言该证明的验证方法是否与相应 DID 文档中的 assertionMethod 属性相关联。

示例：

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ],
  "id": "did:example:123456789abcdefghi",
  ...
  "assertionMethod": [
    // 此方法可用于以 did:...fghi 的身份断言声明
    "did:example:123456789abcdefghi#keys-1",
    // 此方法仅被批准用于声明的断言，不用于任何其他验证关系，因此其完整描述在此处嵌入，而非仅使用引用
    {
      "id": "did:example:123456789abcdefghi#keys-2",
      "type": "Ed25519VerificationKey2020", // 外部（属性值）
      "controller": "did:example:123456789abcdefghi",
      "publicKeyMultibase": "zH3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
    }
  ],
```

```
  ...
}
```

- keyAgreement

keyAgreement 验证关系用于指定一个实体如何生成加密材料，以便向 DID 主体传输机密信息，例如为了与接收方建立安全通信通道。其关联值必须是一组一个或多个验证方法。每个验证方法可以是嵌入的或引用的。在为 DID 主体加密消息时，此属性很有用。在这种情况下，通信对方使用验证方法中的加密公钥信息为接收方封装解密密钥。

示例：

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:example:123456789abcdefghi",
  ...
  "keyAgreement": [
    // 此方法可用于以 did:...fghi 的身份执行密钥协商
    "did:example:123456789abcdefghi#keys-1",
    // 此方法仅被批准用于密钥协商，不会用于任何其他验证关系，因此其完整描述在此处嵌入，而非仅使用引用
    {
      "id": "did:example:123#zC9ByQ8aJs8vrNXyDhPHHNNMSHPcaSgNpjjsBYpMMjsTdS",
      "type": "X25519KeyAgreementKey2019", // 外部（属性值）
      "controller": "did:example:123",
      "publicKeyMultibase": "z9hFgmPVfmBZwRvFEyniQDBkz9LmV7gDEqytWyGZLmDXE"
    }
  ],
```

```
  ...
}
```

- capabilityInvocation

capabilityInvocation 验证关系用于指定 DID 主体可能用于调用加密能力的验证方法，其关联值必须是一组一个或多个验证方法。每个验证方法可以是嵌入的或引用的。

例如，当 DID 主体需要访问受保护的 HTTP API，且该 API 需要授权才能使用时，此属性很有用。为了在使用 HTTP API 时进行授权，DID 主体使用与 HTTP API 公开的特定 URL 相关联的能力。能力的调用可以通过多种方式表达，例如作为放置在 HTTP 标头中的数字签名消息。

提供 HTTP API 的服务器是能力的验证者，它需要验证被调用能力引用的验证方法是否存在于 DID 文档的 capabilityInvocation 属性中。验证者还会检查正在执行的操作是否有效，以及该能力是否适用于正在访问的资源。如果验证成功，服务器通过加密方式确定调用者有权访问受保护的资源。

示例：

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ],
  "id": "did:example:123456789abcdefghi",
  ...
  "capabilityInvocation": [
    // 此方法可用于以 did:...fghi 的身份调用能力
    "did:example:123456789abcdefghi#keys-1",
    // 此方法仅被批准用于能力调用，不会用于任何其他验证关系，因此其完整描述在此处嵌入，而非仅使用引用
    {
      "id": "did:example:123456789abcdefghi#keys-2",
```

```
      "type": "Ed25519VerificationKey2020", // 外部（属性值）
      "controller": "did:example:123456789abcdefghi",
      "publicKeyMultibase": "zH3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
    }
  ],
  ...
}
```


- capabilityDelegation

capabilityDelegation 验证关系用于指定 DID 主体可能用于将加密能力委托给另一方的机制。其关联值必须是一组一个或多个验证方法。每个验证方法可以是嵌入的或引用的。例如，当 DID 控制器选择将其访问受保护 HTTP API 的能力委托给除自身之外的一方时，此属性很有用。为了委托该能力，DID 主体会使用与 capabilityDelegation 验证关系相关联的验证方法，以加密方式将该能力签名给另一个 DID 主体。

示例：


```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ],
  "id": "did:example:123456789abcdefghi",
  ...
  "capabilityDelegation": [
    // 此方法可用于以 did:...fghi 的身份执行能力委托
    "did:example:123456789abcdefghi#keys-1",
```

```
    // 此方法仅被批准用于授予能力，不会用于任何其他验证关系，因此其完整描述在此处嵌入，而非仅使用引用
    {
      "id": "did:example:123456789abcdefghi#keys-2",
      "type": "Ed25519VerificationKey2020", // 外部（属性值）
      "controller": "did:example:123456789abcdefghi",
      "publicKeyMultibase": "zH3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
    }
  ],
  ...
}
```

7.3.2 表示形式

通过序列化将 DID 文档转化成表示，通过反序列化操作将 DID 文档的表示转换成数据模型。表示形式必须满足如下条件：

- 表示形式必须为数据模型中指定的所有数据类型定义确定性的序列化和反序列化规则。
- 表示形式必须唯一地与一个在 IANA 注册的媒体类型相关联。
- 表示形式必须为其媒体类型定义符合片段中定义的片段处理规则的片段处理规则。
- 表示形式应使用数据模型数据类型的词法表示形式。如，JSON 和 JSON-LD 使用 XML Schema dateTime 的词法序列化来表示日期时间。一种表示形式可以选择使用不同的词法序列化来序列化数据模型数据类型，只要转换回数据模型的使用过程是无损的。例如，一些基于 CBOR 的表示形式使用整数来表示自 Unix 纪元以来的秒数来表达日期时间值。
- 表示形式可以定义特定于表示形式的条目，这些条目存储在特定于表示形式的条目映射中，以供生成和使用过程使用。这些条目在使用或生成过程中用于确保无损转换。

本规范定义 DID 文档的表示形式为 JSON。

DID 文档、DID 文档数据结构和表示特定条目映射必须根据以下生成规则序列化为 JSON 表示形式。

数据类型	JSON 表示类型
映射	JSON 对象，其中每个条目被序列化为 JSON 对象的一个成员，条目的键作为 JSON 字符串成员名，条目值根据其类型按照本表定义进行序列化。
列表	JSON 数组，列表中的每个元素按顺序序列化，作为数组的值，其序列化方式根据本表中定义的类型确定。
集合	JSON 数组，集合中的每个元素按顺序添加，作为数组的值，其序列化方式根据本表中定义的类型确定。
日期时间	JSON 字符串，序列化为规范化为 UTC 00:00:00 且无亚秒小数精度的 XML 日期时间格式。例如：2020-12-20T19:17:47Z。
字符串	JSON 字符串。
整数	无小数或分数部分的 JSON 数字。
双精度浮点数	带有小数和分数部分的 JSON 数字。
布尔值	JSON 布尔值。
空值	JSON 空字面量。

DID 文档的所有属性都必须包含在根 JSON 对象中。属性可以包含符合上述值表示规则的其他数据子结构。在序列化 DID 文档时，符合规范的生产者必须向下游应用程序指定媒体类型为 application/did+json。

JSON-LD 是一种基于 JSON 的用于序列化链接数据的格式。规范规定使用 JSON-LD 来描述的较难 DID 文档。

JSON-LD 包含@context 属性用于表示 DID 文档表示的上下文。@context 的序列化值必须是 JSON 字符串 https://www.w3.org/ns/did/v1，或者是一个 JSON 数组，其中第一个元素是 JSON 字符串 https://www.w3.org/ns/did/v1，后续元素根据 JSON 表示形式生成规则进行序列化。

JSON-LD 表示形式的符合规范的生产者不应生成包含未通过@context 定义的术语的 DID 文档，因为符合规范的消费者会预期删除未知术语。在序列化 DID 文档的 JSON-LD 表示形式时，符合规范的生产者必须向下游应用程序指定媒体类型为 application/did+ld+json

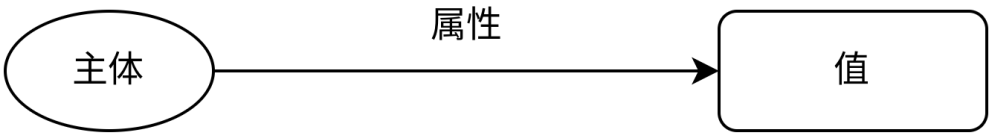
7.4 可验证凭证(Verifiable Credential)

可验证凭证是由同一实体作出的一组包含一项或多项声明的集合。可验证凭证可能还包括一个标识符以及元数据，用于描述该凭证的属性，例如签发者、有效日期和时间段、代表性图像、验证材料、状态信息等等。可验证凭证是一组具有防篡改特性的声明和元数据，通过密码学手段证明其签发者的身份。可验证凭证的示例包括但不限于数字员工身份证、数字驾照和数字教育证书。

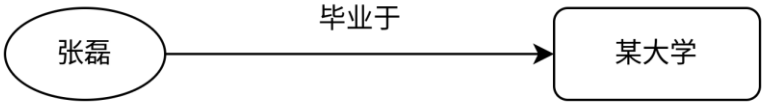
一份可验证凭证可以表示与实体凭证相同的所有信息。通过添加诸如数字签名之类的技术，可验证凭证相比实体凭证更能体现出防篡改特性，也更值得信赖。可验证凭证的持有者可以生成可验证展示，然后将这些可验证展示分享给验证者，以证明他们拥有具有特定特征的可验证凭证。可验证凭证和可验证展示都能够快速传输，这使得它们在远程建立信任关系时，比实体凭证更加便捷。

“可验证凭证”和“可验证展示”这两个术语中的“可验证”一词，指的是如本文档所定义的，凭证或展示具有能够被验证者验证的特性。一份凭证的可验证性并不意味着其中所编码声明的真实性。相反，在确定了一份可验证凭证或可验证展示的真实性和时效性之后，验证者会依据自身的业务规则，在采信这些声明之前对其进行验证。只有在根据一项或多项验证者策略，对凭证的签发者、证明、主体以及声明进行评估之后，才会采信这些声明。

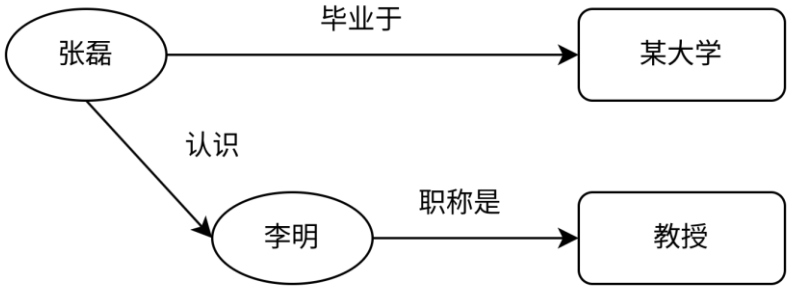
声明是关于主体的一种事实的陈述。主体是可以被做出声明的事物。声明通过“主体 - 属性 - 值”关系来表达。可以通过以下信息图表示：



如下面的声明：



单个声明可以合并在一起，以表达关于一个主体的信息图。上图示例中通过添加 “张磊认识李明” 以及 “李明是一名教授” 这两个声明，扩展了前一个声明：



7.4.1 数据模型

本节出定义在可验证凭证和可验证展示中出现的核心数据项。

7.4.1.1 上下文（context）

当两个软件系统需要交换数据时，它们需要使用双方都理解的术语。就像两个人交流一样，双方必须使用同一种语言，并且他们使用的词汇对彼此来说含义相同，这可以称为对话的上下文。

可验证凭证和可验证展示有许多通过统一资源标识符（URI，[RFC3986]）来标识的属性和值。然而，这些 URI 可能很长，不太便于人们使用。在这种情况下，简短且便于人们使用的别名会更有帮助。本规范使用@context 属性将这些简短的别名映射到特定可验证凭证和可验证展示所需的 URI。

在 JSON-LD 中，@context 属性还可用于传达其他细节，如数据类型信息、语言信息、转换规则等。这些超出了本规范的需求，但在未来或相关工作中可能会有用。

可验证凭证和可验证展示必须包含@context 属性。@context 属性的值必须是一个有序集合，其中第一个元素是值为 https://www.w3.org/2018/credentials/v1 的 URI。数组中的后续元素必须表达上下文信息，并且可以由 URI 或对象的任意组合构成。建议@context 中的每个 URI 在被解析时，都能得到一个包含有关@context 的机器可读信息的文档。

尽管本规范要求存在@context 属性，但不要求必须使用 JSON-LD 来处理@context 属性的值。这是为了支持使用普通 JSON 库进行处理，例如当可验证凭证编码为 JSON Web Token（JWT）时可能会用到的库。所有库或处理器都必须确保@context 属性中值的顺序符合特定应用程序的预期。支持 JSON-LD 的库或处理器可以按预期使用完整的 JSON-LD 处理方式来处理@context 属性。

示例：


```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id": "http://example.edu/credentials/58473",
  "type": ["VerifiableCredential", "AlumniCredential"],
  "issuer": "https://example.edu/issuers/565049",
  "issuanceDate": "2010-01-01T00:00:00Z",
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "alumniOf": {
      "id": "did:example:c276e12ec21ebfeb1f712ebc6f1",
      "name": [{
        "value": "Example University",
        "lang": "en"
      }, {
        "value": "Exemple d'Université",
        "lang": "fr"
      }]
    }
```

```
  },
  "proof": { ... }
}
```

上述示例使用基础上下文 URI（https://www.w3.org/2018/credentials/v1）来表明对话是关于可验证凭证的，第二个 URI（https://www.w3.org/2018/credentials/examples/v1）则表明对话涉及示例相关内容。

https://www.w3.org/2018/credentials/v1 上的数据是一个静态文档，永远不会更新，应该下载并缓存。可验证凭证数据模型的相关人类可读词汇表文档可在 https://www.w3.org/2018/credentials/获取。

7.4.1.2 标识符（Identifier）

在表达关于特定事物（如人、产品或组织）的声明时，使用某种标识符通常很有用，这样其他人就可以使用该标识符来表达关于同一事物的声明。本规范定义了可选的 id 属性来表示这些标识符。id 属性旨在明确地引用一个对象，如人、产品或组织。在可验证凭证中使用 id 属性，能够表达关于特定事物的声明。id 属性必须具有以下特征：

- id 属性必须表达一个标识符，预期其他人在表达关于该标识符所标识的特定事物的声明时会使用这个标识符。
- id 属性不能有多个值。
- id 属性的值必须是一个 URI。建议 id 中的 URI 在被解析时，能得到一个包含有关 id 的机器可读信息的文档。

7.4.1.3 类型（type）

可验证凭证和可验证展示必须具有 type 属性。也就是说，任何没有 type 属性的凭证或展示都不可验证，因此既不是可验证凭证也不是可验证展示。type 属性的值必须是，或通过对@context 属性的解释映射到，一个或多个 URI。如果提供了多个 URI，则这些 URI 必须被解释为一个无序集合。应该使用一些语法便利方式来方便开发者使用，这些便利方式可能包括 JSON-LD 术语。建议 type 中的每个 URI 在被解析时，都能得到一个包含有关 type 的机器可读信息的文档。

示例：

```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ],
  "id": "http://example.edu/credentials/3732",
  "type": [
    "VerifiableCredential",
    "UniversityDegreeCredential"
```

```
  ],
  "issuer": "https://example.edu/issuers/565049",
  "issuanceDate": "2010-01-01T00:00:00Z",
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "degree": {
      "type": "BachelorDegree",
      "name": "Bachelor of Science and Arts"
    }
  },
  "proof": {
    "type": "Ed25519Signature2020",
    "created": "2022-02-25T14:58:43Z",
    "verificationMethod": "https://example.edu/issuers/565049#key-1",
    "proofPurpose": "assertionMethod",
    "proofValue": "zeEdUoM7m9cY8ZyTpey83yBKeBcmcvbyrEQzJ19rD2UXArU2U1jPGoEt
rRvGYppdiK37GU4NBeoPakxpWhAvsVSt"
  }
}
```


就本标准中，下列的对象中必须指定 type 属性：

对象	类型
可验证凭证对象（凭证对象的一个子类）	必须是“可验证凭证（VerifiableCredential）”，并且可以选择指定一个更具体的可验证凭证类型。例如， “type": ["VerifiableCredential","UniversityDegreeCredential"]
凭证对象	必须是“可验证凭证（VerifiableCredential）”，并且可以选择指定一个更具体的可验证凭证类型。例如， “type": ["VerifiableCredential","UniversityDegreeCredential"]
可验证展示对象（展示对象的一个子类）	必须是“可验证展示（VerifiablePresentation）”，并且可以选择指定一个更具体的可验证展示类型。例如， “type": ["VerifiablePresentation","CredentialManagerPresentation"]
展示对象	必须是“可验证展示（VerifiablePresentation）”，并且可以选择指定一个更具体的可验证展示类型。例如， “type": ["VerifiablePresentation","CredentialManagerPresentation"]

对象	类型
证明对象	一个有效的证明类型。例如，“type":"RsaSignature2018"
凭证状态对象	一个有效的凭证状态类型。例如，“type":"CredentialStatusList2017"）
使用条款对象	一个有效的使用条款类型。例如，“type":"OdrlPolicy2017"
证据对象	一个有效的证据类型。例如，“type":"DocumentVerification2018"

7.4.1.4 凭证主体（Credential Subject）

可验证凭证包含关于一个或多个主体的**声明**。本标准定义了 credentialSubject 属性，用于表达关于一个或多个主体的声明。

可验证凭证必须有 credentialSubject 属性。

credentialSubject：credentialSubject 属性的值定义为一组对象，这些对象包含一个或多个与可验证凭证的主体相关的属性。每个对象可以包含一个 id。

示例：

```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id": "http://example.edu/credentials/3732",
  "type": ["VerifiableCredential", "UniversityDegreeCredential"],
  "issuer": "https://example.edu/issuers/565049",
  "issuanceDate": "2010-01-01T00:00:00Z",
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
```

```
    "degree": {
      "type": "BachelorDegree",
      "name": "Bachelor of Science and Arts"
    }
  }
}
```


在一个可验证凭证中表达与多个主体相关的信息是允许的。如以下示例中


```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id": "http://example.edu/credentials/3732",
  "type": ["VerifiableCredential", "RelationshipCredential"],
  "issuer": "https://example.com/issuer/123",
  "issuanceDate": "2010-01-01T00:00:00Z",
  "credentialSubject": [{
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "name": "Jayden Doe",
    "spouse": "did:example:c276e12ec21ebfeb1f712ebc6f1"
  }, {
    "id": "did:example:c276e12ec21ebfeb1f712ebc6f1",
```

"name": "Morgan Doe",

"spouse": "did:example:ebfeb1f712ebc6f1c276e12ec21"

}]

}

7.4.1.5 签发者(issuer)

本规范定义了一个属性，用于表示可验证凭证的**签发者**。

可验证凭证必须具备“issuer”（**签发**者）属性。

issuer 属性的值必须是一个统一资源标识符（URI），或者是一个包含“id”属性的对象。建议当解析“issuer”中的 URI 或其“id”时，能够得到一个包含有关发行者的机器可读信息的文档，这些信息可用于验证凭证中所表述的信息。

示例：

{

"@context": [

"https://www.w3.org/2018/credentials/v1",

"https://www.w3.org/2018/credentials/examples/v1"

],

"id": "http://example.edu/credentials/3732",

"type": ["VerifiableCredential", "UniversityDegreeCredential"],

"issuer": "https://example.edu/issuers/14",

"issuanceDate": "2010-01-01T19:23:24Z",

"credentialSubject": {

"id": "did:example:ebfeb1f712ebc6f1c276e12ec21",

"degree": {

"type": "BachelorDegree",

```
      "name": "Bachelor of Science and Arts"
    }
  }
}
```


通过将一个对象与 issuer（**签发者**）属性相关联，来表达关于发行者的更多额外信息也是可行的。


```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id": "http://example.edu/credentials/3732",
  "type": ["VerifiableCredential", "UniversityDegreeCredential"],
  "issuer": {
    "id": "did:example:76e12ec712ebc6f1c221ebfeb1f",
    "name": "Example University"
  },
  "issuanceDate": "2010-01-01T19:23:24Z",
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "degree": {
      "type": "BachelorDegree",
      "name": "Bachelor of Science and Arts"
```

```
    }
  }
}
```

7.4.1.6 签发日期(issuanceDate)

本规范定义了一个属性，用于表示可验证凭证的签发日期。

可验证凭证必须具有 issuanceDate（签发日期）属性。issuanceDate”属性的值必须是一个符合 [ISO 8601] 标准的日期时间字符串。日期时间字符串必须采用 “基本” 格式（例如 “2010-01-01T00:00:00Z”）或 “扩展” 格式（例如 “2010-01-01T00:00:00.000Z”）。日期时间字符串必须包含一个 “Z” 后缀，以表示协调世界时（UTC）。

示例：

```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id": "http://example.edu/credentials/3732",
  "type": ["VerifiableCredential", "UniversityDegreeCredential"],
  "issuer": "https://example.edu/issuers/14",
  "issuanceDate": "2010-01-01T19:23:24Z",
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "degree": {
      "type": "BachelorDegree",
      "name": "Bachelor of Science and Arts"
    }
```

```
  }
}
```

7.4.1.7 证明（Proofs）

本规范定义了一个属性，用于表达对可验证凭证或可验证展示的证明。证明用于验证可验证凭证或可验证展示的完整性，以及证明其来源。

可验证凭证和可验证展示必须具有 proof 属性。

proof 属性的值必须是一个对象，该对象包含用于验证可验证凭证或可验证展示的完整性和来源的信息。这个对象必须至少包含以下属性：

type：type 属性的值必须是一个有效的证明类型，如 “RsaSignature2018” 或 “Ed25519Signature2020”。建议 “type” 中的 URI 在被解析时，能得到一个包含有关证明类型的机器可读信息的文档。

created：created 属性的值必须是一个符合 [ISO 8601] 标准的日期时间字符串。日期时间字符串必须采用 “基本” 格式（例如 “2010-01-01T00:00:00Z”）或 “扩展” 格式（例如 “2010-01-01T00:00:00.000Z”）。日期时间字符串必须包含一个 “Z” 后缀，以表示协调世界时（UTC）。

verificationMethod：verificationMethod 属性的值必须是一个 URI，该 URI 引用用于验证证明的公钥或其他验证方法。建议 verificationMethod 中的 URI 在被解析时，能得到一个包含有关验证方法的机器可读信息的文档。

proofPurpose：proofPurpose 属性的值必须是一个字符串，用于描述证明的目的。例如，“assertionMethod”（断言方法）表示该证明用于验证凭证中声明的真实性，“authentication”（身份验证）表示该证明用于验证持有者的身份。

此外，proof 对象还可能包含其他属性，具体取决于所使用的证明类型。例如，“proofValue”（证明值）属性通常用于包含实际的签名值。

示例：

```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id": "http://example.gov/credentials/3732",
  "type": ["VerifiableCredential", "UniversityDegreeCredential"],
  "issuer": "https://example.edu",
  "issuanceDate": "2010-01-01T19:23:24Z",
```

```
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "degree": {
      "type": "BachelorDegree",
      "name": "Bachelor of Science and Arts"
    }
  },
  "proof": {
    "type": "Ed25519Signature2020",
    "created": "2021-11-13T18:19:39Z",
    "verificationMethod": "https://example.edu/issuers/14#key-1",
    "proofPurpose": "assertionMethod",
    "proofValue": "z58DAdFfa9SkqZMVPxAQpic7ndSayn1PzZs6ZjWp1CktyGesjuTSwRdo
                   WhAfGFCF5bppETSTojQCrfFPP2oumHKtz"
  }
}
```


7.4.1.8 过期时间(Expiration)

本规范定义了一个属性，用于表示可验证凭证的过期时间。

可验证凭证可以选择具有 “expirationDate”属性。expirationDate 属性的值必须是一个符合 [ISO 8601] 标准的日期时间字符串。日期时间字符串必须采用 “基本” 格式（例如 “2010-01-01T00:00:00Z”）或 “扩展” 格式（例如 “2010-01-01T00:00:00.000Z”）。日期时间字符串必须包含一个 “Z” 后缀，以表示协调世界时（UTC）

过期日期可用于确定在特定时间点可验证凭证是否仍然有效。验证者可以检查当前日期和时间是否早于过期日期，以决定是否接受该凭证。如果未提供 expirationDate 属性，则可验证凭证没有明确的过期时间，并且在其他方面有效的情况下，可被视为无限期有效。

示例：


```
{
```

```
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id": "http://example.edu/credentials/3732",
  "type": ["VerifiableCredential", "UniversityDegreeCredential"],
  "issuer": "https://example.edu/issuers/14",
  "issuanceDate": "2010-01-01T19:23:24Z",
  "expirationDate": "2020-01-01T19:23:24Z",
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "degree": {
      "type": "BachelorDegree",
      "name": "Bachelor of Science and Arts"
    }
  }
}
```


7.4.1.9 状态（Status）

本规范定义了一个属性，用于表示可验证凭证的状态。可验证凭证的状态可能表明该凭证是有效、已撤销、已过期或其他相关状态。

可验证凭证可以选择具有 credentialStatus 属性。credentialStatus”属性的值必须是一个对象，该对象包含有关可验证凭证状态的信息。这个对象必须至少包含 type 属性，其值必须是一个有效的凭证状态类型，例如 “CredentialStatusList2017” 或其他被认可的凭证状态类型。建议 type 中的 URI 在被解析时，能得到一个包含有关凭证状态类型的机器可读信息的文档。

除了 type 属性外，credentialStatus 对象还可能包含其他属性，具体取决于所使用的凭证状态类型。例如，对于“CredentialStatusList2017”类型，可能包含一个指向凭证状态列表的 URI，以及用于在列表中标识该凭证的索引或标识符。

示例：

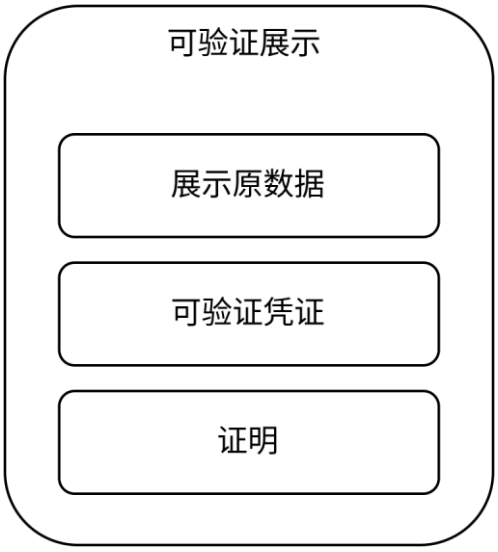

```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id": "http://example.edu/credentials/3732",
  "type": ["VerifiableCredential", "UniversityDegreeCredential"],
  "issuer": "https://example.edu/issuers/14",
  "issuanceDate": "2010-01-01T19:23:24Z",
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "degree": {
      "type": "BachelorDegree",
      "name": "Bachelor of Science and Arts"
    }
  },
  "credentialStatus": {
    "id": "https://example.edu/status/24",
    "type": "CredentialStatusList2017"
  }
```

}

7.4.2 可验证展示(Verifiable Presentation)

可验证展示是来自一个或多个可验证凭证的数据，并且以一种可验证数据来源的方式进行打包。如果直接展示可验证凭证，它们就成为了可验证展示。从可验证凭证派生出来、经过加密可验证但本身不包含可验证凭证的数据格式，也可能属于可验证展示。

展示中的数据通常是关于同一主体的，但也可能由多个发行者发布。这些信息的聚合通常体现了一个人、组织或实体的某个方面。



下图完整地描绘了一个可验证展示，它通常至少由四个信息图组成。其中第一个信息图，即展示图（Presentation Graph），表达可验证展示本身，包含展示元数据。展示图中的 verifiableCredential 属性指向一个或多个可验证凭证，每个可验证凭证都是第二个信息图，即一个自包含的凭证图（Credential Graph），凭证图又包含凭证元数据和声明。第三个信息图，即凭证证明图（Credential Proof Graph），表达凭证图的证明，通常是一个数字签名。第四个信息图，即展示证明图（Presentation Proof Graph），表达展示图的证明，通常也是一个数字签名。

7.4.2.1 数据模型

可验证展示是可验证凭证的一种集合呈现形式，用于向验证者证明某些声明。

可验证展示的目的是让持有者能够有选择地向验证者展示一组可验证凭证，以证明某些声明，同时保护持有者的隐私，只展示必要的信息。验证者可以通过检查可验证展示的属性和证明，来验证展示的真实性和所包含声明的有效性。

本小节介绍可验证展示的基本属性和要求。

可验证展示必须具有以下属性：

@context：“@context” 属性的值必须是一个有序集合，其中第一个元素是值为 https://www.w3.org/2018/credentials/v1 的 URI。这用于表明该展示是关于可验证凭证相关内容的。数组中的后续元素必须表达上下文信息，并且可以由 URI 或对象的任意组合构成。和可验证凭证一样，建议@context 中的每个 URI 在被解析时，都能得到一个包含有关@context 的机器可读信息的文档。

type：type 属性的值必须是，或通过对@context 属性的解释映射到，一个或多个 URI。至少必须包含 VerifiablePresentation，并且可以选择包含一个更具体的可验证展示类型，例如 “CredentialManagerPresentation”。如果提供了多个 URI，则这些 URI 必须被解释为一个无序集合。同样建议 type 中的每个 URI 在被解析时，都能得到一个包含有关 type 的机器可读信息的文档。

verifiableCredential：verifiableCredential 属性的值必须是一个数组，其中包含一个或多个可验证凭证。这些可验证凭证可以是完整的可验证凭证对象，也可以是对可验证凭证的引用（例如，通过 URI 引用）。

此外，可验证展示还可以包含 proof 属性，其要求与可验证凭证中的 proof 属性类似。proof 属性的值必须是一个对象，用于验证可验证展示的完整性和来源。这个对象必须至少包含 type（证明类型）、created（创建时间，符合 [ISO 8601] 标准的日期时间字符串，包含 “Z” 后缀表示协调世界时）、verificationMethod（验证方法，引用用于验证证明的公钥或其他验证方法的 URI）和 proofPurpose（证明目的，描述证明用途的字符串）等属性。

示例：

```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id": "urn:uuid:3978344f-8596-4c3a-a978-8fcaba3903c5",
  "type": ["VerifiablePresentation", "CredentialManagerPresentation"],
  "verifiableCredential": [{ ... }],
  "proof": [{ ... }]
}
```


7.4.2.2 派生方式的可验证展示

可验证展示也可以包含从一个或多个原始可验证凭证派生而来的凭证。派生凭证是通过对原始凭证中的信息进行处理、转换或组合而创建的新凭证。这种方式允许持有者在不泄露原始凭证所有信息的情况下，向验证者展示特定的声明。

例如，持有者可能拥有一个包含详细个人信息的教育程度凭证，但验证者只需要知道持有者是否拥有特定类型的学位。在这种情况下，持有者可以从原始的教育程度凭证派生出一个只包含学位类型信息的新凭证，并在可验证展示中包含这个派生凭证。

派生凭证的创建应该遵循一定的规则和流程，以确保其真实性和有效性。通常，派生凭证会包含一个指向原始凭证的引用，以及描述如何从原始凭证派生的信息。

可验证展示中包含派生凭证时，必须满足与包含常规可验证凭证相同的基本要求。即展示必须具有 @context、type 和 verifiableCredential 属性，并且如果包含 proof 属性，也必须符合相应的要求。

7.4.3 表示语法

可验证凭证和可验证展示可以使用多种语法进行编码，以适应不同的应用场景和技术需求。本节介绍了一些常见的语法及其特点，这些语法旨在实现可验证凭证和可验证展示的创建、存储、传输和验证。

本节规定了数据模型如何在 JSON-LD（关联数据的 JavaScript 对象表示法）和纯 JSON 中得以实现。尽管仅针对这两种语法提供了语法映射，但应用程序和服务可以使用任何其他能够表达该数据模型的数据表示语法，例如 XML（可扩展标记语言）、YAML（一种人性化的数据标记语言）或 CBOR（二进制对象表示法）。由于验证和确认要求是依据数据模型来定义的，所以所有的序列化语法都必须能够确定地转换为数据模型，以便进行处理、验证或比较。本规范对支持任何特定的序列化格式不做要求。

本规范中属性值的预期元数，以及存储这些值所产生的数据类型，可能会因属性的不同而有所差异。如果存在以下属性，则将其表示为单个值：

- id 属性
- issuer 属性
- issuanceDate 属性
- expirationDate 属性

所有其他属性（如果存在），则表示为单个值或值的数组。

可验证凭证和可验证展示的核心数据模型基于 JSON-LD（JavaScript Object Notation for Linked Data），这是一种用于在 JSON 中表示链接数据的标准。JSON-LD 提供了一种灵活的方式来定义上下文、标识符、类型和其他相关属性，使得可验证凭证和可验证展示能够在不同的系统之间进行互操作。

7.4.3.1 JSON

可验证凭证和可验证展示章节中的数据模型中所描述的数据模型，可以通过将属性值映射到 JSON 类型的方式，用 JavaScript 对象表示法（JSON）[RFC8259] 进行编码，具体映射方式如下：

- 可表示为 IEEE754 格式的数值类型，应表示为 JSON 中的 “Number”（数字）类型。
- 布尔值类型，应表示为 JSON 中的 “Boolean”（布尔）类型。
- 序列值类型，应表示为 JSON 中的 “Array”（数组）类型。
- 无序的值集合，应表示为 JSON 中的 “Array”（数组）类型。
- 属性集合，应表示为 JSON 中的 “Object”（对象）类型。
- 空值，应表示为 JSON 中的 “null”（空）值。
- 其他值，必须表示为 JSON 中的 “String”（字符串）类型。

7.4.3.2 JSON-LD

JSON-LD 是一种基于 JSON 的格式，用于对关联数据进行序列化处理。这种语法的设计目的是便于融入那些已经在使用 JSON 的已部署系统中，并提供了从 JSON 到 [JSON-LD] 的平滑升级路径。它主要被用作一种在基于 Web 的编程环境中使用关联数据的方式，用于构建可互操作的 Web 服务，并将关联数据存储在基于 JSON 的存储引擎中。

在扩展本规范中所描述的数据模型时，JSON-LD 非常有用。数据模型的实例在 JSON-LD 中的编码方式与在 JSON 中的编码方式相同，只是增加了@context 属性。JSON-LD 的上下文在 JSON-LD 规范中有详细描述。

可以使用或组合多个上下文，以惯用的 JSON 格式来表达关于可验证凭证的任何任意信息。可在 https://www.w3.org/2018/credentials/v1 获取的 JSON-LD 上下文，是一个静态文档，永远不会更新，因此可以在客户端下载并缓存。可验证凭证数据模型的相关词汇表文档可在 https://www.w3.org/2018/credentials 获取。

JSON-LD 提供的一些最为值得关注的语法糖特性如下：

- @id 和 @type 关键字分别是 id 和 type 的别名，这使得开发人员能够像使用标准的 JSON 那样来运用本规范。
- 诸如整数、日期、度量单位和统一资源定位符（URL）等数据类型会被自动赋予类型，从而为那些对类型有严格要求的用例提供更可靠的类型保障。
- verifiableCredential（可验证凭证）和 proof（证明）属性被视为图容器。也就是说，它们是用于隔离由不同实体所声明的数据集的机制。例如，这可确保每个发行者提供的数据图与展示可验证凭证的持有者所提供的数据图之间实现恰当的密码学隔离，从而保证每个数据图中信息的来源得以保留。
- JSON-LD 1.1 中的 @protected（受保护）属性特性用于确保本规范所定义的术语不会被覆盖。这意味着，只要在可验证凭证或可验证展示的顶部进行相同的 @context（上下文）声明，那么对于数

据模型的使用者而言，无论他们是否使用 JSON-LD 处理器，所有能被理解的术语都能保证互操作性。

8 参考

[1]. Infra Standard; https://infra.spec.whatwg.org/
[2]. xmlschema11-2; Datatypes.https://www.w3.org/TR/xmlschema11-2/
[3]. RFC8259; https://www.rfc-editor.org/rfc/rfc8259
[4]. RFC3986; https://www.rfc-editor.org/rfc/rfc3986
[5]. MULTIBASE; https://datatracker.ietf.org/doc/html/draft-multiformats-multibase-03
[6]. RFC7517: https://www.rfc-editor.org/rfc/rfc7517
[7]. JSON Web Signature 2020;https://w3c-ccg.github.io/lds-jws2020/