1 DID

A decentralised identifier (DID) is an identifier that isn't issued by a centralised body. The DID is owned by an entity which can verify control of it using a cryptographically signed document.

The document consists of the following parts:

1. context: see https://w3c-ccg.github.io/did-spec/
2. id: the DID
3. authentication: the details to authenticate
   (a) id: Identifier for the authentication entry
   (b) type type of the authentication entry
   (c) controller the DID providing authentication
   (d) publicKeyPem the public key of the DID

1.1 Decentralised Identifiers

A DID is a unique identifier which must be persistent and immutable.

The DID method needs to ensure that two entities cannot claim ownership of the same DID.

There is a comparison in the specification of UUID's which are decentralised as a result of their uniqueness, in the same way that Bitcoin addresses are also generated to avoid collisions.

An entity can use a DID to pass to another entity and verify that they are the owner of that DID. This mechanism is similar somewhat to providing a service with your email address and then clicking a verify link in an email that is subsequently sent to that address.

I'm not sure how well the persistence requirement falls in line with GDPR as there is no right to forget functionality here. Persistence automatically reduces privacy in a system and so goes against the privacy-by-design principal.

1.2 DID Document

A DID Document contains metadata about a DID. The DID document is what is exchanged during a interaction between two entities.

The DID Document contains a public key to verify ownership of the DID, using a DID Method

2 method

Creating a DID requires that a DID can be linked to a public key. The common implementation of this requires that a reference to the DID exists with a reference to the private key.

Bitcoin generates addresses that can be linked back to public keys and so having a UID generated in the same way that a bitcoin address is generated would satisfy the need to derive the UID from a public key.

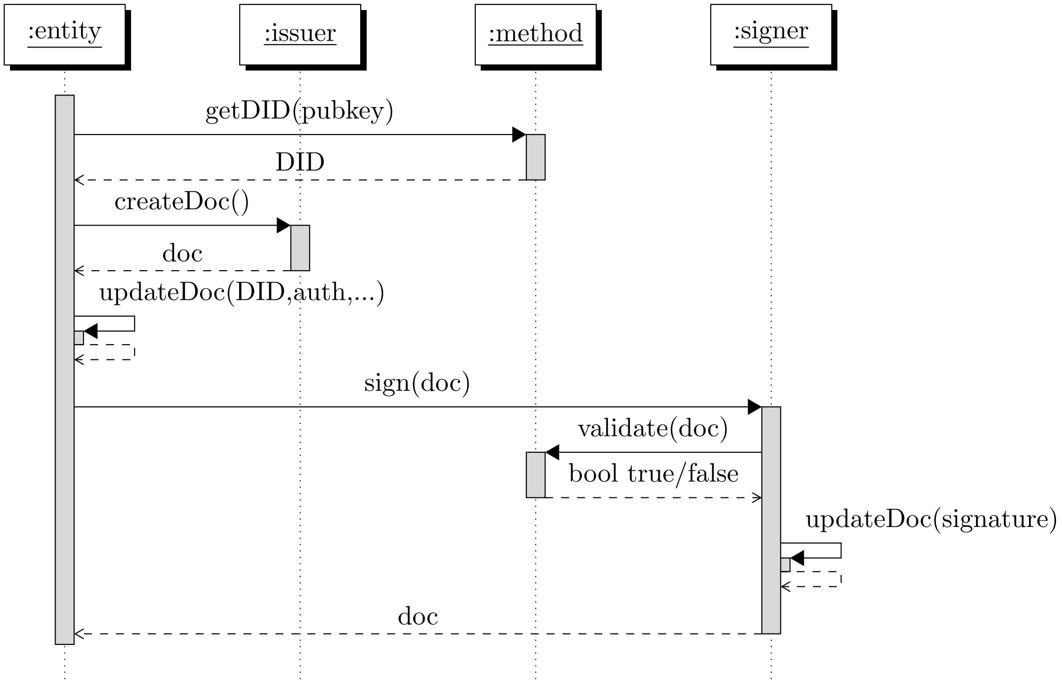
2.1 document creation

Entity, Issuer and Signer can all be the same entity, different entities or any mixture of.

1. entity gets a DID using the DID Method

2. entity requests a blank document from the issuer

3. the issuer creates a document using a Document Descriptor Object


:entity
:issuer
:method
:signer
getDID(pubkey)
DID
createDoc()
doc
updateDoc(DID,auth,...)
sign(doc)
validate(doc)
bool true/false
updateDoc(signature)
doc