DID Resolution Architecture [... in Aries]

Markus Sabadello
markus@danubetech.com

2019-07-03 Aries Working Group Call

DANUBE TECH GMBH

DID Resolution

```
resolve (did, input-options)

--> did-document
--> resolution-result
```

```
did:sov:WRfXPg8dantKVubE3HX8pw -->
```

```
{
  "@context": "https://w3id.org/did/v1",
  "id": "did:sov:WRfXPg8dantKVubE3HX8pw",
  "publicKey": [
    {
      ...
    }
  ],
  "service": [
    {
      ...
    }
  ]
}
```

DANUBE TECH GMBH

DID Resolution

- Local vs. remote “Read” operation

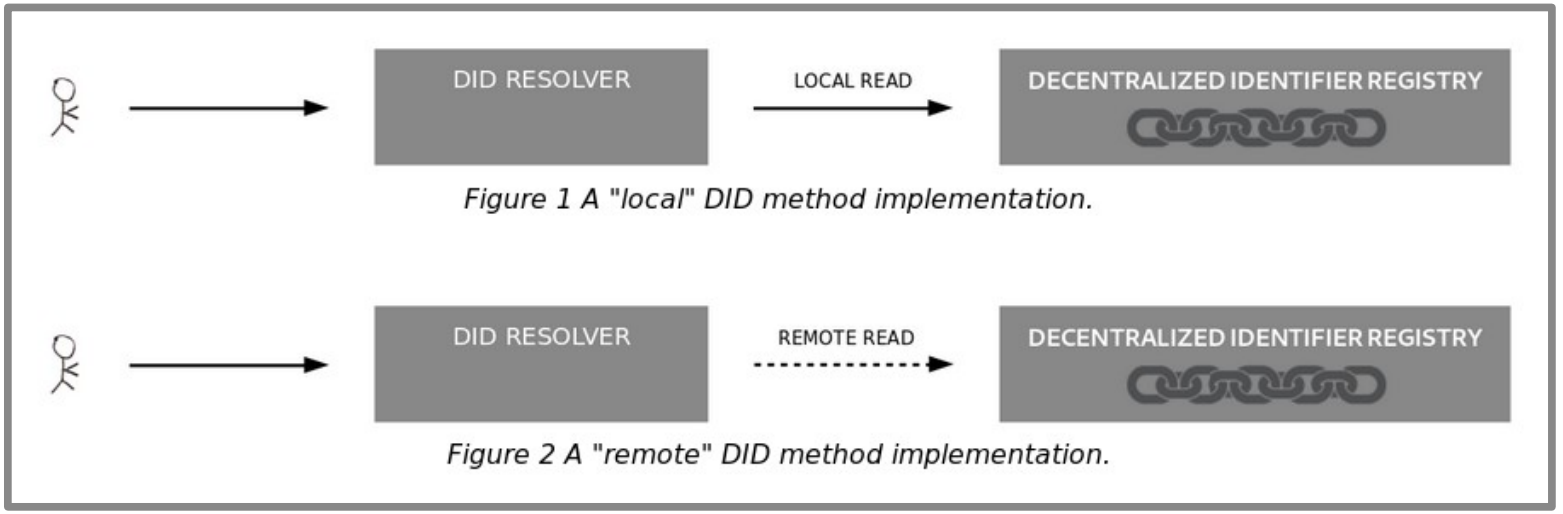


Figure 1 A "local" DID method implementation.

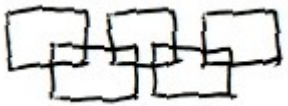
Figure 2 A "remote" DID method implementation.

- Blockchain full nodes, light clients, peer DIDs, wrapped public keys, etc.

DANUBE TECH GMBH

did:sov:WRfXPg8dantKVubE3HX8pw

Target System



Sovrin Ledger

NYM: [18,{"dest":"WRfXPg8dantKVubE3HX8pw","identifier":
"BrYDA5NubejDVHkCYBbpY5","reqId":1501522732982387,"signature":"5HGRA...",
"verkey":"~P7F3BNs5VmQ6eVpwkNKJ5D"}]

ATTRIB: [19,{"dest":"WRfXPg8dantKVubE3HX8pw","identifier":
"WRfXPg8dantKVubE3HX8pw","raw":"0249fedf5246b...","reqId":1504718156368788,
"signature":"3jL1ZNjLAzyAm5"}]

...

...

...

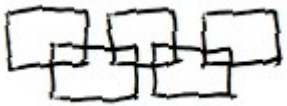
DID Document


```
{
  "@context": "https://w3id.org/did/v1",
  "id": "did:sov:WRfXPg8dantKVubE3HX8pw",
  "publicKey": [
    {
      "id":"did:sov:WRfXPg8dantKVubE3HX8pw#key-1",
      "type": "Ed25519VerificationKey2018",
      "publicKeyBase58": "H3C2AVvLMv6gmMNam3uVAj..."
    }
  ],
  "service": {
    "type": "xdi",
    "serviceEndpoint": "http://127.0.0.1/xdi/"
  }
}
```


DANUBE TECH GMBH

did:btcr:xz35-jzv2-qqs2-9wjt

Target System



Bitcoin Blockchain

BLOCK 1202316

TX #80: 5310788c3f8c47d2e0336a4de7ecaceb52405699b571bd1254bf4580caf6

TXIN #1: P2PKH muorV4hjg9EFxE7U1MScUnpQ5gFqCtMdzh
TXOUT #1: P2PKH mkhu17qayX84QK6Hvj3BQPPjhf93hQmYvU
TXOUT #2: OP_RETURN https://btcr.host.com/peacekeeper/self.ddo

TX #81: a8150d3d1e7e635314ca0bd2b8976aa5d98d46f7bd64dfc850969586afb2

TXIN #1: P2PKH muAA7os3wCEDB46bmveP4eVKNwC6jz75KF
TXOUT #1: P2PKH mvysHdp7Fnqda8ivgWAduTvC3DvGhr6Qjk

...

https://btcr.host.com/peacekeeper/self.ddo

```
{
    "@context": "https://w3id.org/did/v1",
    "id": "did:btcr:xz35-jzv2-qqs2-9wjt",
    "service": [ {
        "type": "xdi",
        "serviceEndpoint": "http://127.0.0.1/xdi/"
    } ],
    "signature": { ... }
}
```

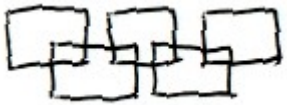
DID Document

```
{
  "@context": "https://w3id.org/did/v1",
  "id": "did:btcr:xz35-jzv2-qqs2-9wjt",
  "publicKey": [
    {
      "id":"did:btcr:xz35-jzv2-qqs2-9wjt#key-1",
      "type": "EdDsaSAPublicKeySecp256k1",
      "publicKeyHex": "H3C2AVvLMv6gmMNam3uVAj..."
    }
  ],
  "service": {
    "type": "xdi",
    "serviceEndpoint": "http://127.0.0.1/xdi/"
  }
}
```

DANUBE TECH GMBH

did:v1:test:nym:3AEJTDMSxDDQpyUftju

Target System



Veres One Ledger

```
{
  "@context": "https://w3id.org/did/v1",
  "id": "did:v1:test:nym:3AEJTDMSxDDQpyUftju",
  "publicKey": [
    {
      "id":"did:v1:test:nym:3AEJTDMSxDDQpyUftju#key-1",
      "type": "Ed25519VerificationKey2018",
      "publicKeyBase58": "H3C2AVvLMv6gmMNam3uVAj..."
    }
  ],
  "service": {
    "type": "xdi",
    "serviceEndpoint": "http://127.0.0.1/xdi/"
  }
}
```

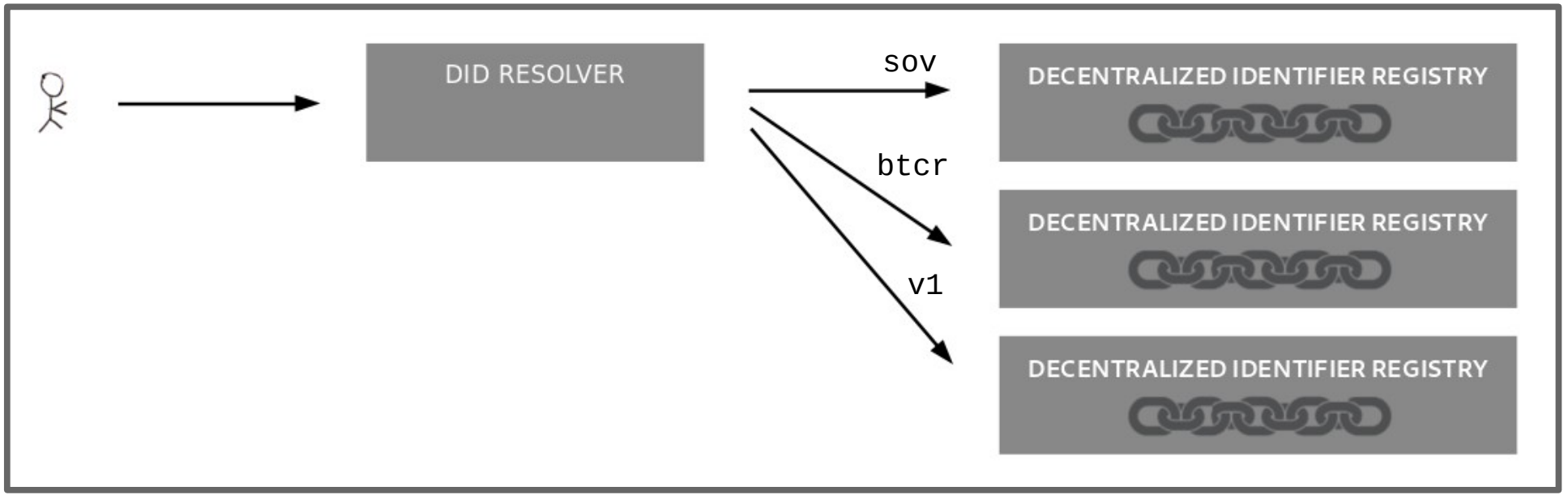
DID Document

```
{
  "@context": "https://w3id.org/did/v1",
  "id": "did:v1:test:nym:3AEJTDMSxDDQpyUftju",
  "publicKey": [
    {
      "id":"did:v1:test:nym:3AEJTDMSxDDQpyUftju#key-1",
      "type": "Ed25519VerificationKey2018",
      "publicKeyBase58": "H3C2AVvLMv6gmMNam3uVAj..."
    }
  ],
  "service": {
    "type": "xdi",
    "serviceEndpoint": "http://127.0.0.1/xdi/"
  }
}
```

DANUBE TECH GMBH

DID Resolution

- A DID Resolver can support multiple DID methods (using “drivers”):



DANUBE TECH GMBH

DID Resolution

- Local vs. remote bindings

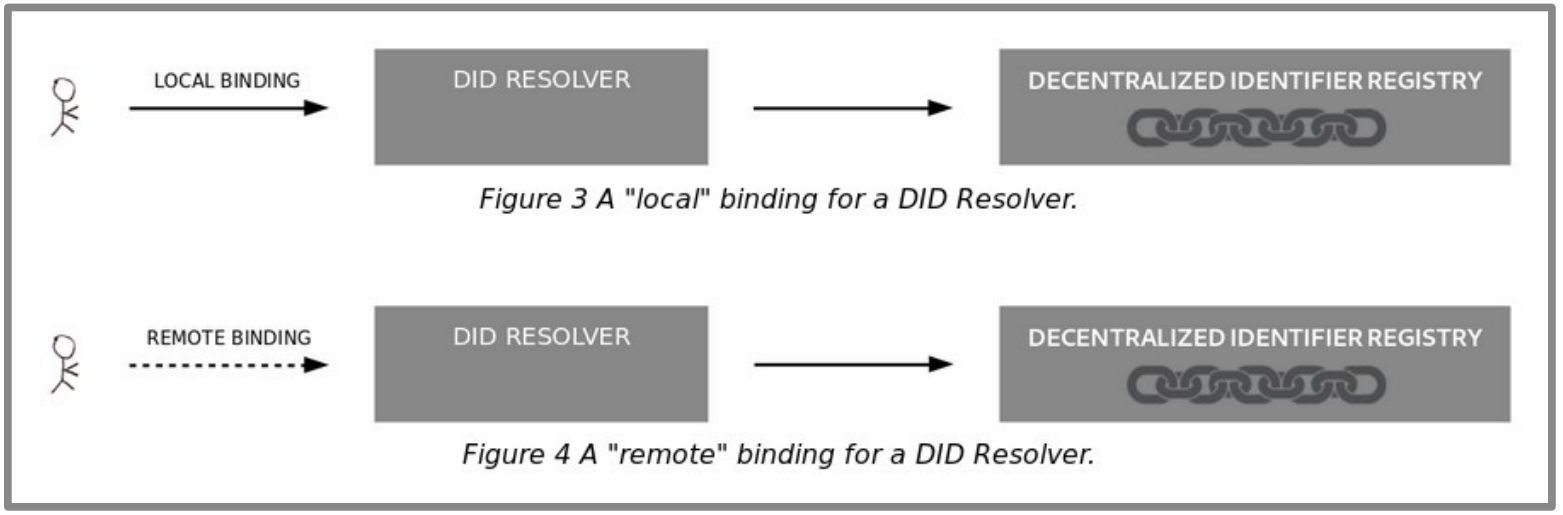


Figure 3 A "local" binding for a DID Resolver.

Figure 4 A "remote" binding for a DID Resolver.

- Local: API call, command line, etc.
- Remote: HTTP(S) GET, DIDComm, etc.

DANUBE TECH GMBH

DID Resolution

- DIF Universal Resolver

```
public ResolveResult resolve(String identifier, Map<String, String> options);
```

```
curl -X GET https://uniresolver.io/1.0/identifiers/did:sov:WRfXPg8dantKVubE3HX8pw
```

- Digital Bazaar's did-client – https://github.com/digitalbazaar/did-cli

```
$ ./did get did:v1:test:nym:EfmQqD9Be3MXc4PiGVf5ruEmZqGiVq7cKuYUMcaf2UYs
```

- uPort's DID Resolver – https://github.com/uport-project/did-resolver

```
resolve('did:eth:0xf3beac30c498d9e26865f34fcaa57dbb935b0d74').then(...)
```

- indy-sdk 014-did-doc.md/README.md

```
pub fn indy_resolve_did_doc(did_resolver_handle,did,options) -> Future<did_doc>
```

DANUBE TECH GMBH

DID Resolution

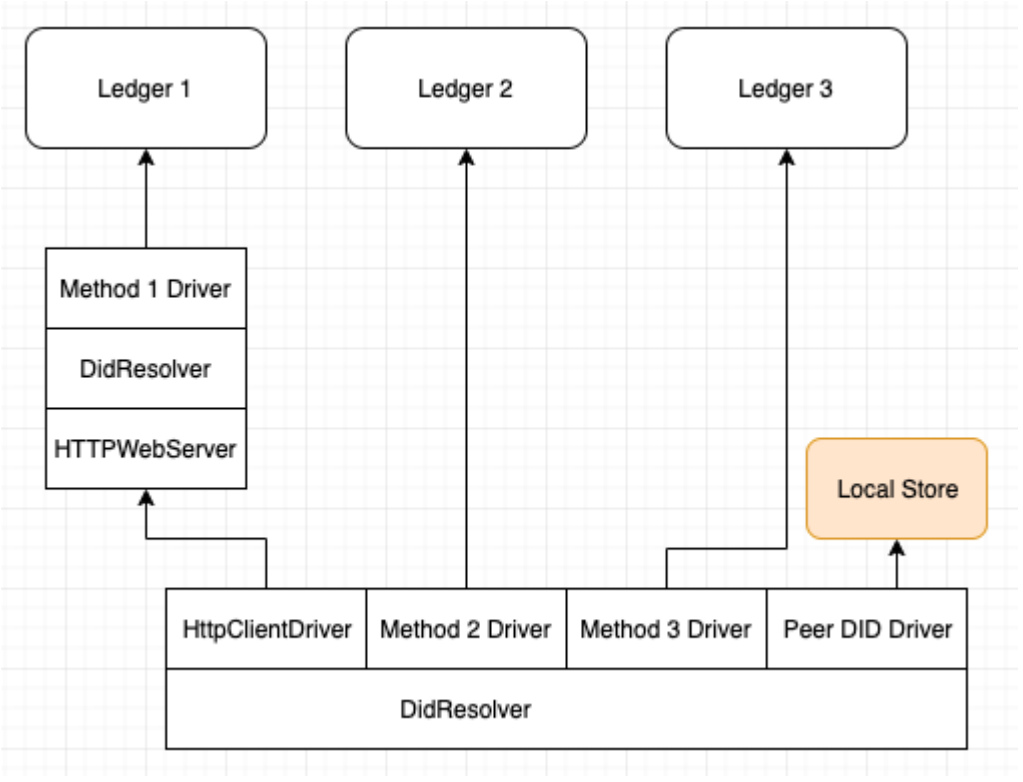
- DID Resolver invoked via one (local) binding, invoking another DID Resolver via (remote) binding:

- E.g. Aries DID Resolver invoked via local API, calls a DIF Universal Resolver via HTTP(S) or DIDComm

DANUBE TECH GMBH

DID Resolution

- Diagram by **@tplooker** at **https://github.com/hyperledger/aries-rfcs/issues/101**



DANUBE TECH GMBH

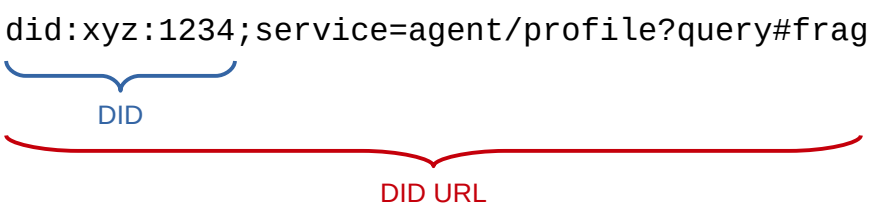
DID URL Dereferencing

```
dereference (did-url, input-options)

--> did-document
--> part of a did-document
--> service endpoint uri
--> other resource


did:xyz:1234;service=agent/profile?query#frag
did:xyz:1234;version-time=1554389617#keys-1
did:xyz:1234;type=schema;id=z9y8x7w6
```

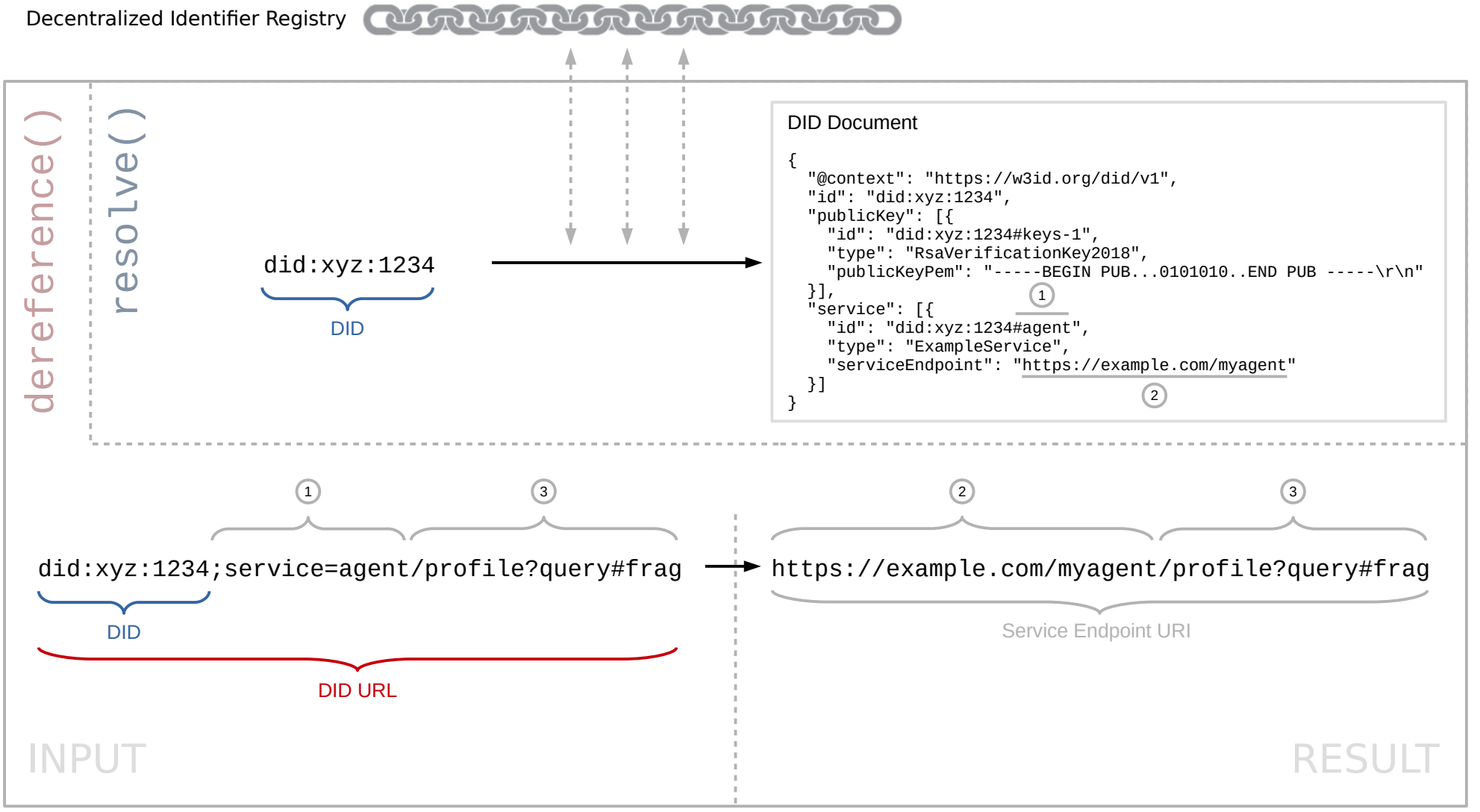
DANUBE TECH GMBH

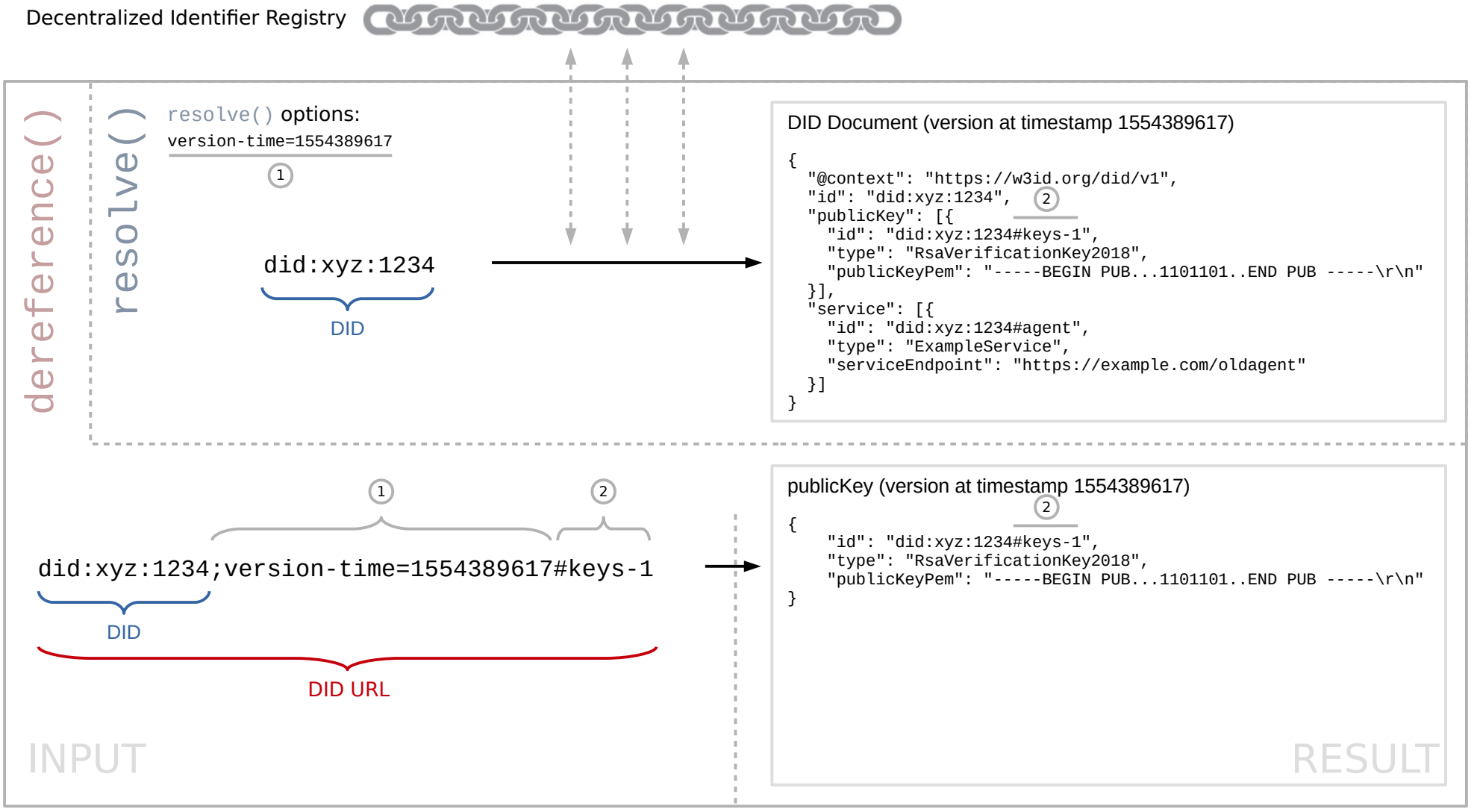
```
did                = "did:" method-name ":" method-specific-id
method-name        = 1*method-char
method-char        = %x61-7A / DIGIT
method-specific-id = *idchar *( ":" *idchar )
idchar             = ALPHA / DIGIT / "." / "-" / "_"
did-url            = did *( ";" param ) path-abempty [ "?" query ] [ "#" fragment ]
param              = param-name [ "=" param-value ]
param-name         = 1*param-char
param-value        = *param-char
param-char         = ALPHA / DIGIT / "." / "-" / "_" / ":" / pct-encoded
```

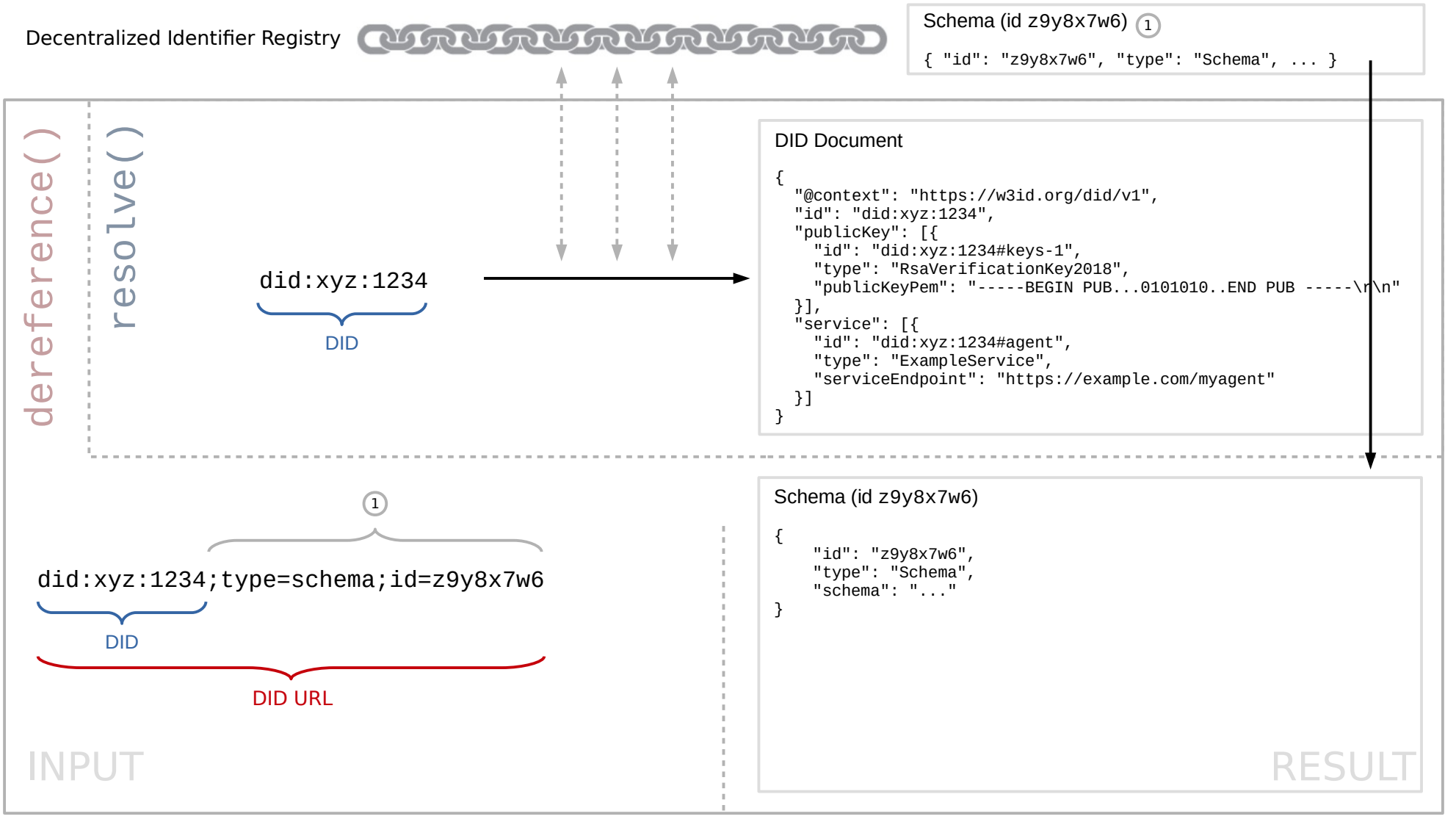
Example:



DANUBE TECH GMBH


Decentralized Identifier Registry
dereference()
resolve()
did:xyz:1234
DID
DID Document
{
"@context": "https://w3id.org/did/v1",
"id": "did:xyz:1234",
"publicKey": [{
"id": "did:xyz:1234#keys-1",
"type": "RsaVerificationKey2018",
"publicKeyPem": "-----BEGIN PUB...0101010..END PUB -----\r\n"
}],
"service": [{
"id": "did:xyz:1234#agent",
"type": "ExampleService",
"serviceEndpoint": "https://example.com/myagent"
}]
}
1
2
1
3
did:xyz:1234;service=agent/profile?query#frag
DID
DID URL
2
3
https://example.com/myagent/profile?query#frag
Service Endpoint URI
INPUT
RESULT


Decentralized Identifier Registry
dereference()
resolve()
resolve() options:
version-time=1554389617
1
did:xyz:1234
DID
DID Document (version at timestamp 1554389617)
{
"@context": "https://w3id.org/did/v1",
"id": "did:xyz:1234",
2
"publicKey": [{
"id": "did:xyz:1234#keys-1",
"type": "RsaVerificationKey2018",
"publicKeyPem": "-----BEGIN PUB...1101101..END PUB -----\r\n"
}],
"service": [{
"id": "did:xyz:1234#agent",
"type": "ExampleService",
"serviceEndpoint": "https://example.com/oldagent"
}]
}
1
2
did:xyz:1234;version-time=1554389617#keys-1
DID
DID URL
publicKey (version at timestamp 1554389617)
2
{
"id": "did:xyz:1234#keys-1",
"type": "RsaVerificationKey2018",
"publicKeyPem": "-----BEGIN PUB...1101101..END PUB -----\r\n"
}
INPUT
RESULT


Decentralized Identifier Registry
Schema (id z9y8x7w6) 1
{ "id": "z9y8x7w6", "type": "Schema", ... }
dereference()
resolve()
did:xyz:1234
DID
DID Document
{
"@context": "https://w3id.org/did/v1",
"id": "did:xyz:1234",
"publicKey": [{
"id": "did:xyz:1234#keys-1",
"type": "RsaVerificationKey2018",
"publicKeyPem": "-----BEGIN PUB...0101010..END PUB -----\r\n"
}],
"service": [{
"id": "did:xyz:1234#agent",
"type": "ExampleService",
"serviceEndpoint": "https://example.com/myagent"
}]
}
1
did:xyz:1234;type=schema;id=z9y8x7w6
DID
DID URL
Schema (id z9y8x7w6)
{
"id": "z9y8x7w6",
"type": "Schema",
"schema": "..."
}
INPUT
RESULT

More DID URL Examples

```
did:xyz:1234;service=agent/profile?query#frag
did:xyz:1234;version-time=1554389617#keys-1
did:xyz:1234;service-type=agent
did:xyz:1234;type=schema;id=z9y8x7w6
did:xyz:1234?type=schema&id=z9y8x7w6
did:xyz:1234/schemas/z9y8x7w6
```

DANUBE TECH GMBH

Next steps?

1) Define and implement Aries API for `resolve()` and `dereference()` functions?
   - Extensible driver architecture like Universal Resolver, did-client, etc.?
2) Define DIDComm protocol as remote binding for DID Resolvers?

Thank You.

https://github.com/hyperledger/aries-rfcs/issues/101

https://w3c-ccg.github.io/did-resolution/

DANUBE TECH GMBH