DID Resolution

A work item of the W3C CCG

16th January 2020

IPR rules apply!

Zoom Call:
https://zoom.us/j/7077077007

Text Chat:
http://irc.w3.org/?channels=ccg
(type q+ to get into the queue)

This call is being recorded!

Agenda Thursday 16 January 2020

Welcome and CCG IPR reminder

Call info and scribe selection

Agenda creation/review/prioritization

Matrix parameters vs. query parameters

Preparation for DID WG F2F in Amsterdam?

AOB

Next meeting

Matrix parameters? What does this mean?

```
did:ex:123;service=files/myresume/doc?version=latest#intro
```

DIDs (mostly) resolve to 1. public keys and 2. service endpoints

Some applications just need public keys

`did:ex:123`

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:ex:123",
  "authentication": [{
    "id": "did:ex:123#keys-1",
    "type": "Ed25519VerificationKey2018",
    "publicKeyBase58": "H3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
  }]
}
```

Many applications just need public keys and one service endpoint

```
did:ex:123
```

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:ex:123",
  "authentication": [{
    "id": "did:ex:123#keys-1",
    "type": "Ed25519VerificationKey2018",
    "publicKeyBase58": "H3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
  }],
  "service": [{
    "id": "did:ex:123",
    "type": "AgentService",
    "serviceEndpoint": "https://agency.com/myagent"
  }]
}
```

But I can have many “services” (= URLs associated with my DID)

`did:ex:123`

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:ex:123",
  "service": [{
    "id": "did:ex:123#homepage",
    "serviceEndpoint": "https://alice.me/home/"
  }, {
    "id": "did:ex:123#work",
    "serviceEndpoint": "https://acmecorp.com/employees/alice-7332"
  }, {
    "id": "did:ex:123#linkedin",
    "type": "SocialNetworkService",
    "serviceEndpoint": "https://www.linkedin.com/in/alice-b26187c4/"
  }, {
    "id": "did:ex:123#message",
    "type": "ActivityPubService",
    "serviceEndpoint": "https://chaos.social/@alice01"
  }
}
```

Not all “services” necessarily require a special protocol or DID-based auth/encryption..

Other mechanisms for discovering links from a URL

```
GET https://alice.me/ HTTP/1.1
```

```
<link rel="stylesheet" href="/media/example.css">
```

Used by IndieWeb community for service/link discovery:

```
<link rel="micropub" href="https://alice.me/pub">

<link rel="profile" href="https://profiles.com/alice">
```

Or using the HTTP “Link” Header

```
Link: <https://profiles.com/alice>; rel="profile"
```

...and XRD/JRD,
and WebID, etc.

PURLs (Persistent URLs)

From Wikipedia:

“A Persistent URL is an address on the World Wide Web that causes a redirection to another Web resource. “

“If a Web resource changes location (and hence URL), a PURL pointing to it can be updated.”

```
http://purl.org/some/path → http://example.com/another/path
http://purl.org/some/path → http://selfhosted.me:8080/
```

“The PURL service includes a concept known as partial redirection.”

```
http://purl.org/some/path/and/some/more/data → http://example.com/another/path/and/some/more/data
```

“The concept of **partial redirection allows hierarchies of Web-based resources to be addressed via PURLs** without each resource requiring its own PURL.”

(“PURLs have been criticized for their need to resolve a URL, thus tying a PURL to a network location.”)

But I can have many “services” (= URLs associated with my DID)

```
did:ex:123
```

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:ex:123",
  "service": [{
    "id": "did:ex:123#homepage",
    "serviceEndpoint": "https://alice.me/home/"
  }, {
    "id": "did:ex:123#work",
    "serviceEndpoint": "https://acmecorp.com/employees/alice-7332"
  }, {
    "id": "did:ex:123#linkedin",
    "type": "SocialNetworkService",
    "serviceEndpoint": "https://www.linkedin.com/in/alice-b26187c4/"
  }, {
    "id": "did:ex:123#message",
    "type": "ActivityPubService",
    "serviceEndpoint": "https://chaos.social/@alice01"
  }
}
```

```
did:ex:123#profilepic
```

```
did:ex:123;service=profilepic
```

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:ex:123",
  "authentication": [{
    "id": "did:ex:123#keys-1",
    "type": "RsaVerificationKey2018",
    "publicKeyPem": "-----BEGIN PUB...01..END PUB -----\r\n"
  }],
  "service": [{
    "id": "did:ex:123#profilepic",
    "serviceEndpoint":
      "https://socialnetwork.com/user/123/profile?image"
  }]
}
```

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:ex:123",
  "authentication": [{
    "id": "did:ex:123#keys-1",
    "type": "RsaVerificationKey2018",
    "publicKeyPem": "-----BEGIN PUB...01..END PUB -----\r\n"
  }],
  "service": [{
    "id": "did:ex:123#profilepic",
    "serviceEndpoint":
      "https://newnetwork.com/account022/photo.jpg"
  }]
}
```

↓

https://socialnetwork.com/user/123/profile?image

↓

https://newnetwork.com/account022/photo.jpg

But what about hierarchies? Partial redirection as in PURLs?

did:ex:123#files~~/myresume/doc?version=latest#intro~~

did:ex:123;service=files/myresume/doc?version=latest#intro

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:ex:123",
  "publicKey": [{
    "id": "did:ex:123#keys-1",
    "type": "RsaVerificationKey2018",
    "publicKeyPem": "-----BEGIN PUB...01..END PUB -----\r\n"
  }],
  "service": [{
    "id": "did:ex:123#files",
    "serviceEndpoint":
      "https://filestore.org/user123/"
  }]
}
```

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:ex:123",
  "publicKey": [{
    "id": "did:ex:123#keys-1",
    "type": "RsaVerificationKey2018",
    "publicKeyPem": "-----BEGIN PUB...01..END PUB -----\r\n"
  }],
  "service": [{
    "id": "did:ex:123#files",
    "serviceEndpoint":
      "https://selfhosted.me:8080/"
  }]
}
```

https://filestore.org/user123/myresume/doc?version=latest#intro

https://selfhosted.me:8080/myresume/doc?version=latest#intro

```
did:ex:123;service=files/myresume/doc?version=latest#intro
```

Combines the best features of:

- Service (link) discovery, like <link rel=".." href=”..”>
- Persistent URL discovery, like PURL or URN resolution

Plus the other features of DIDs, i.e. decentralization, cryptographic verifiability!

TODO: RFC8141: URNs have “r-components”, “q-components”, “f-components”

```
did:ex:123;service=files/myresume/doc?version=latest#intro
```

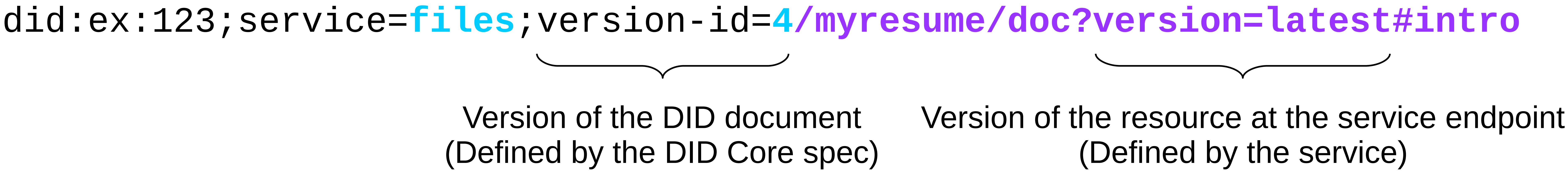
What if we use query parameters instead?

```
did:ex:123/myresume/doc?_did_service=files&version=latest#intro
```

How would this get dereferenced to this redirect URL?

```
https://selfhosted.me:8080/myresume/doc?version=latest#intro
```

More matrix parameters! (version-id)



```
did:ex:123;service=files;version-id=4/myresume/doc?version=latest#intro
```

Version of the DID document
(Defined by the DID Core spec)

Version of the resource at the service endpoint
(Defined by the service)

More matrix parameters! (hl)

```
did:ex:123;service=files;hl=zQmWvQ/myresume/doc?hl=zQrMLp#intro
```

Hash of the DID document
(Verified by the DID resolver)

Hash of the resource at the service endpoint
(Verified by the client application)