

DID/VC 超入門

雰囲気を理解する

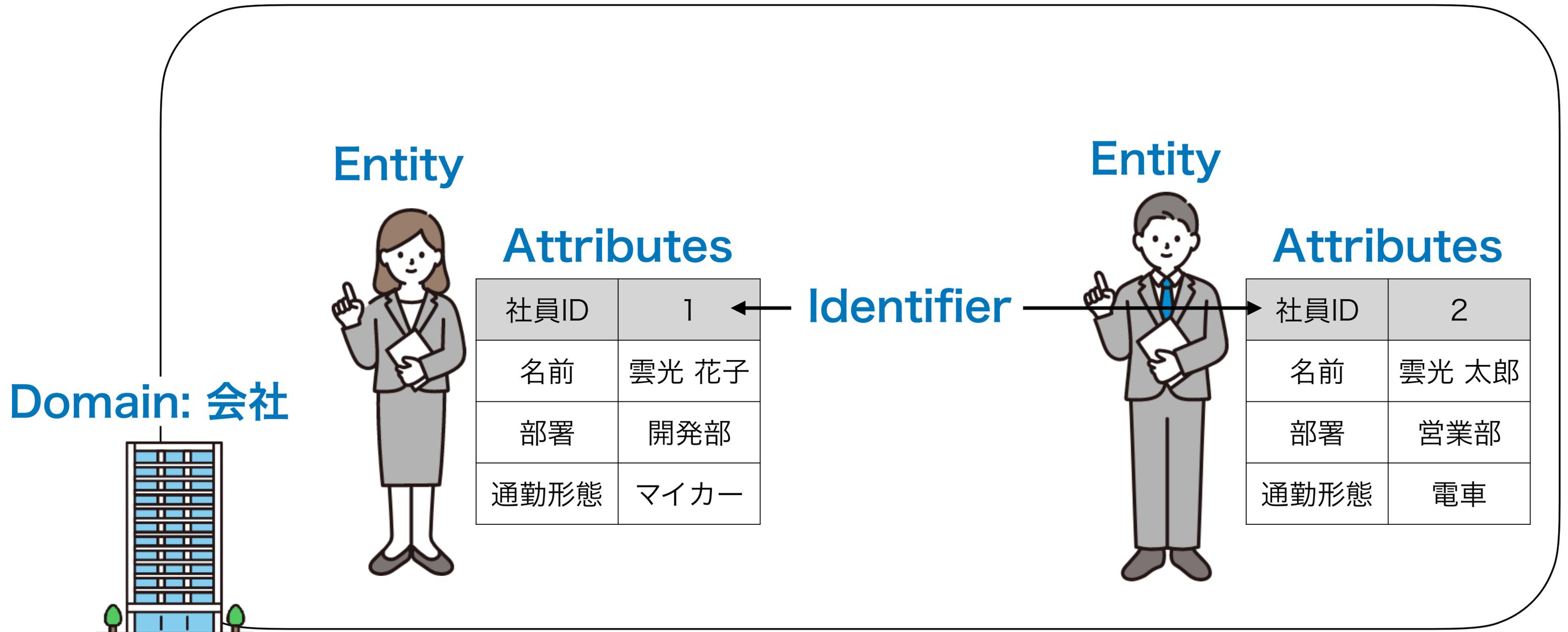
Identity

定義

- **アイデンティティ (Identity)**の定義:
エンティティに関連する属性の集合
(Set of attributes related to an entity)
- **エンティティ (Entity)**の定義:
ドメインの運用目的に関連する項目で、認識できるほど明確な存在を持つもの
(Item relevant for the purpose of operation of a domain that has recognizably distinct existence)
- **属性 (Attribute)**の定義:
エンティティの特徴または性質
(Characteristic or property of an entity)
- **識別子 (Identifier)**の定義:
ドメイン内のアイデンティティを一意に特徴付ける属性(の集合)
(Attribute or set of attributes that uniquely characterizes an identity in a domain)
- **Claim**の定義:
発行者 (Issuer) が正しいと主張した属性

Identity

定義(図解)

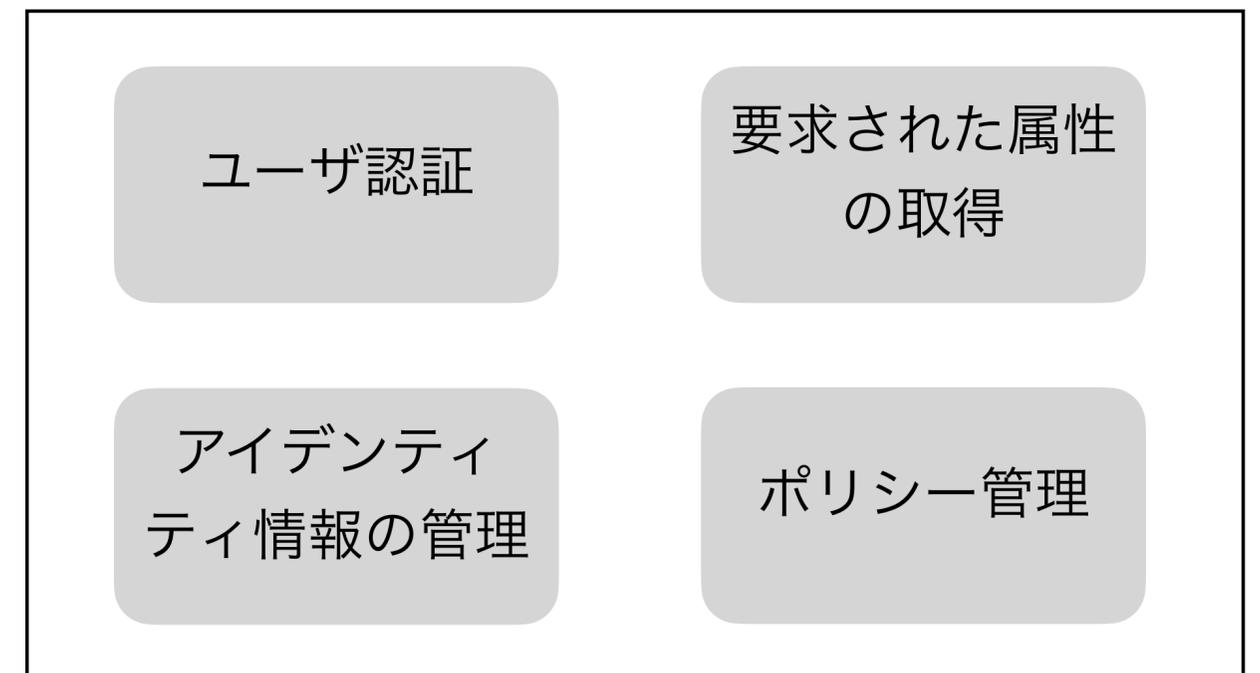


Identity管理

IdM/IMS

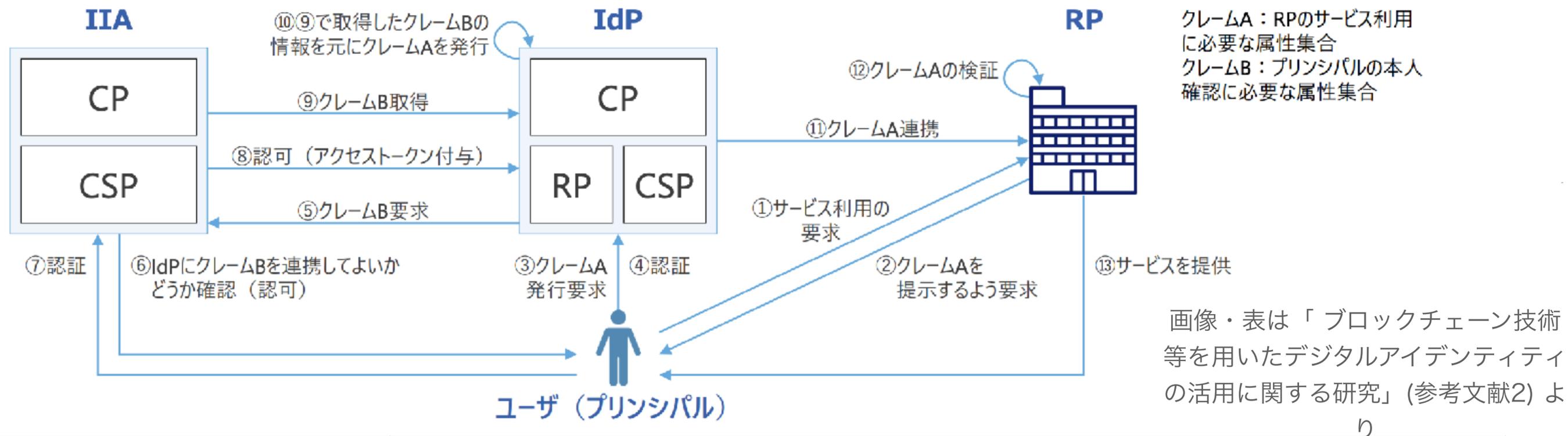
- **Identity Management (IdM):**
情報システムの利用者についてのアイデンティティや権限情報などを一元的に登録・管理すること
- **Identity Management System (IMS):**
アイデンティティ情報を管理するシステム

IMS



Identity管理

IdP/RP



アクター	定義
プリンシパル(Principal)	IMSによりアイデンティティ情報が保存・管理されているエンティティ
アイデンティティ情報オーソリティ (Identity Information Authority: IIA)	アイデンティティの一つ以上の属性値の正しさについて、証明可能な記述を行うことができるエンティティ
アイデンティティプロバイダ (Identity Provider: IdP)	利用可能なアイデンティティ情報を提供し、アイデンティティ情報を作成・維持しているエンティティ (IIAがIdPの役割を担うケースあり)
クレデンシャルサービスプロバイダ (Credential Service Provider: CSP)	クレデンシャル(アイデンティティの表明)の管理責任を負う、信頼されたエンティティ
クレームプロバイダ (Claims Provider: CP)	クレームを提供するエンティティ
リライングパーティー (Relying Party: RP)	特定のエンティティのアイデンティティ情報の検証に依拠するエンティティ

Identity管理

IdP/RP



続行するには Spotify にログインしてください。

FACEBOOK で続ける

APPLE で続ける

GOOGLE で続ける

または

Eメールアドレスまたはユーザー名

Eメールアドレスまたはユーザー名

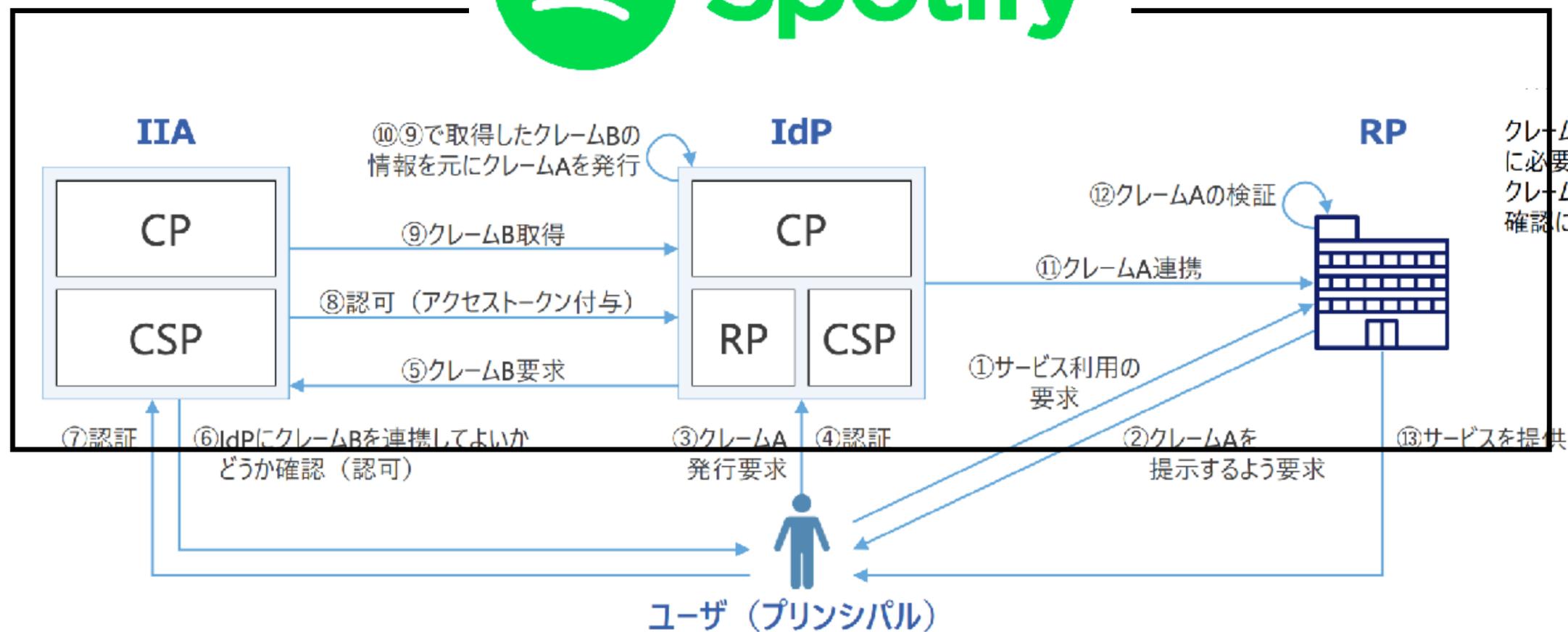
パスワードを設定してください。

パスワードを設定してください。

[パスワードをお忘れですか？](#)

ログイン情報を記憶する

ログイン



Identity管理

IdP/RP



続行するには Spotify にログインしてください。

FACEBOOK で続ける

APPLE で続ける

GOOGLE で続ける

または

Eメールアドレスまたはユーザ名

Eメールアドレスまたはユーザ名

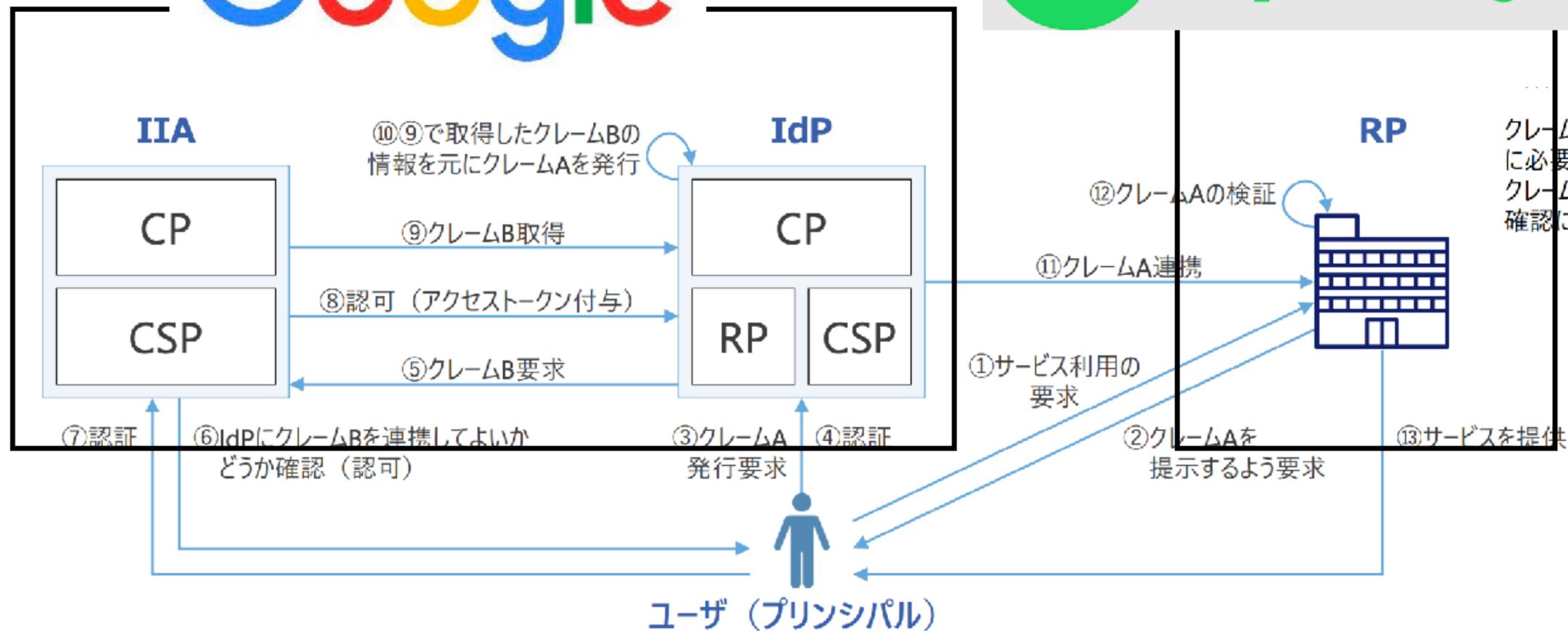
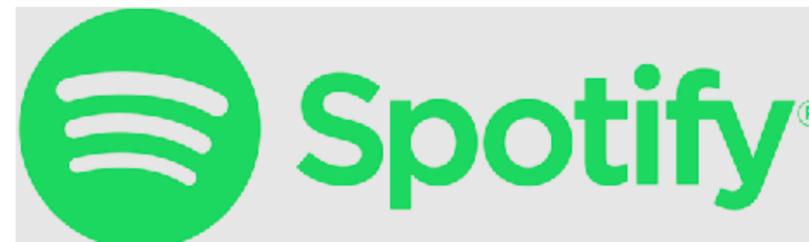
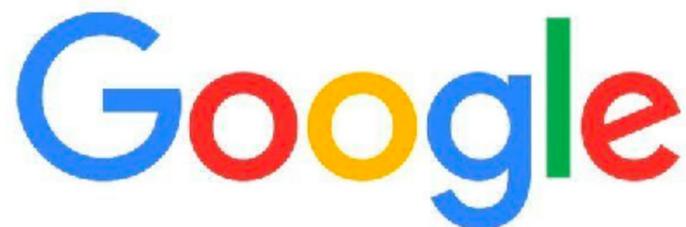
パスワードを設定してください。

パスワードを設定してください。

パスワードをお忘れですか?

ログイン情報を記憶する

ログイン



IMSモデル

集中型・フェデレーションモデル

- **集中型:**

RPがIdPを運営している。ユーザはサービス毎にアイデンティティ管理を行う

- ▶ フッターなWebサービスはこれ

- **フェデレーションモデル:**

RPとIdPは別エンティティ。ユーザはRPのサービスにアクセスする際、IdPのアイデンティティ情報を用いる

- ▶ 「Facebookでログイン」「Googleでログイン」などはこの類

自己主権型アイデンティティ

既存のIMSモデルのリスク

- 既存のIMSモデルにはリスクがある
 - ▶ IdPがノリで垢Banしてきたら、多数のRPサービスが使えなくなってしまう!?
 - ▶ IdPがお亡くなりになったらどうするの!?
 - ▶ IdPがアイデンティティ情報を改ざんしてきたら!?
 - ▶ RPにアクセスした履歴がIdPにバレる!?
- そこで**自己主権型アイデンティティ (Self Sovereign Identity)**



自己主権型アイデンティティ

概要

- アイデンティティの管理主体が介在することなく、個人が自分自身のアイデンティティをコントロールできるようにすることを目指す「思想」
- ユーザが自分の属性情報に関するコントロール権を確保
- 信頼できる組織から発行された本人の属性情報を取得し、ユーザの許可した範囲でRPに連携する

自己主権型アイデンティティ

Kim Cameronによって提示されたSSI 7原則

Kim Cameron

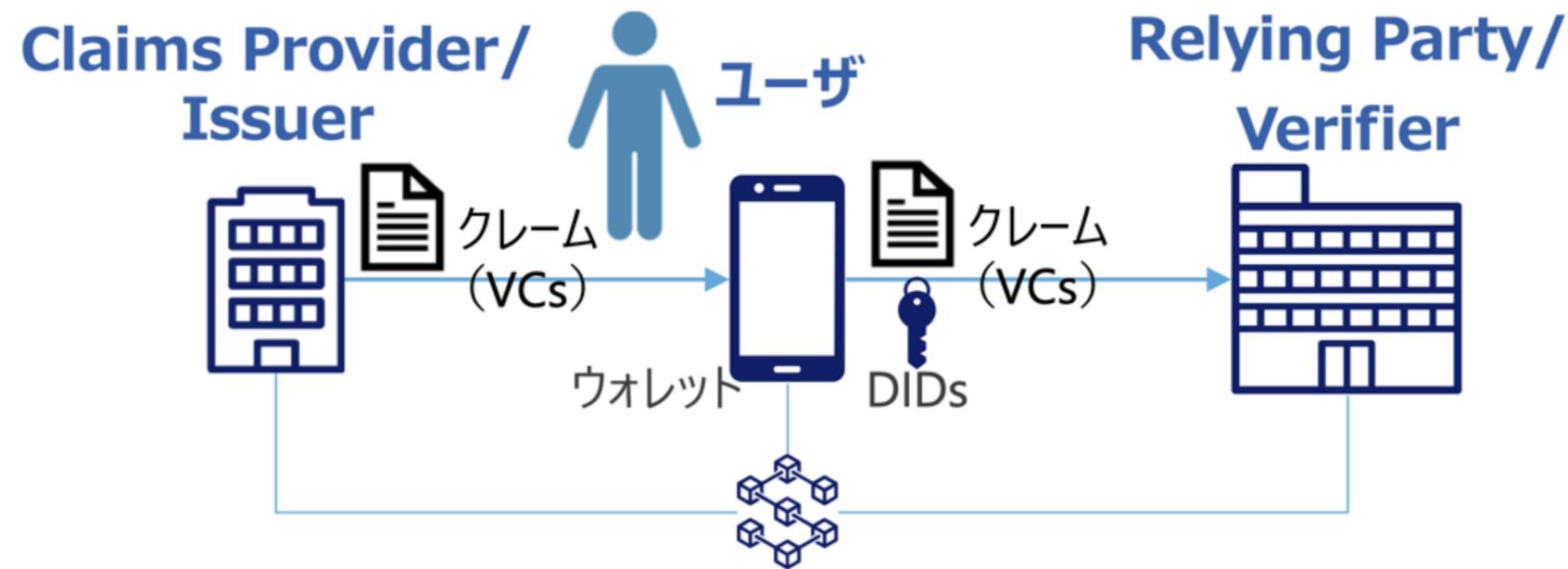


- 1. ユーザによる制御と同意**
アイデンティティ・システムは、ユーザの同意がなければユーザを識別する情報を開示すべきではない
- 2. 限定された用途で最低限の公開**
最も安定し、長期にわたって使用できるソリューションとは、開示するアイデンティティ情報を最小限にし、情報へのアクセスを適切に制限するソリューションである
- 3. 正当な関係者のみへの情報開示**
アイデンティティ・システムは、特定の状況において識別情報を必要とし、かつ入手できる正当な権利を持つ関係者のみに対して情報を開示するように設計されなければならない
- 4. 方向づけられたアイデンティティ**
アイデンティティ・システムは、公に使用する「全方位的」な識別子とプライベートで使用する「特定の方向性」を持った識別子の両方をサポートしなければならない。このことにより公共性を維持しながら不必要に関連付けの公開を防止できる
- 5. 「アイデンティティハブ」への統合**
ユーザは、プロバイダ間で一貫した方法で自分自身を表現し、アイデンティティを使用することができ、同時にコンテキスト間でアイデンティティを分離することができる
- 6. 長期のアイデンティティの安定性に向けたDIDの統合**
個人データを事業者に依存しない形で保管したうえで、アイデンティティ事業者を存続させ、サービスとの関係を維持する
- 7. 人間の統合**
アイデンティティ・システムは、利用者たるユーザを分散システムの1つのコンポーネントとして定義しなければならない。明確なマンマシン・インターフェイスを策定してユーザを分散システムに統合し、アイデンティティを保護しなければならない

「ブロックチェーン技術等を用いたデジタルアイデンティティの活用に関する研究」(参考文献2) より

DID/VC

- 分散型アイデンティティ (Decentralized Identity: DID): ユーザのデジタルアイデンティティが特定のIdPに依存しないよう、依存度を下げる



画像は「ブロックチェーン技術等を用いたデジタルアイデンティティの活用に関する研究」(参考文献2) より

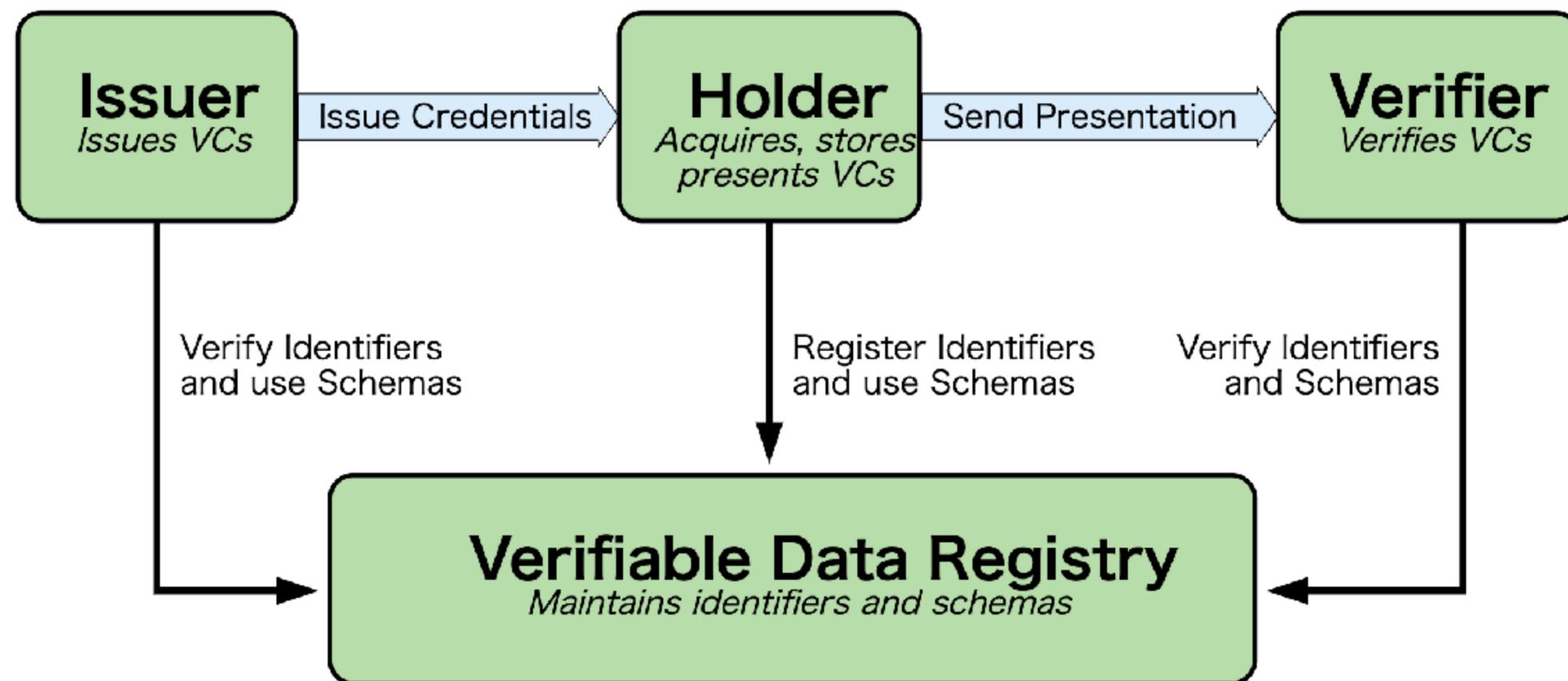
DID/VC

W3Cが標準化しようとしているDID技術

- Decentralized Identifiers: DIDs
識別子のほう
- Verifiable Credentials: VCs
クレデンシャル(証明書)のデータモデル

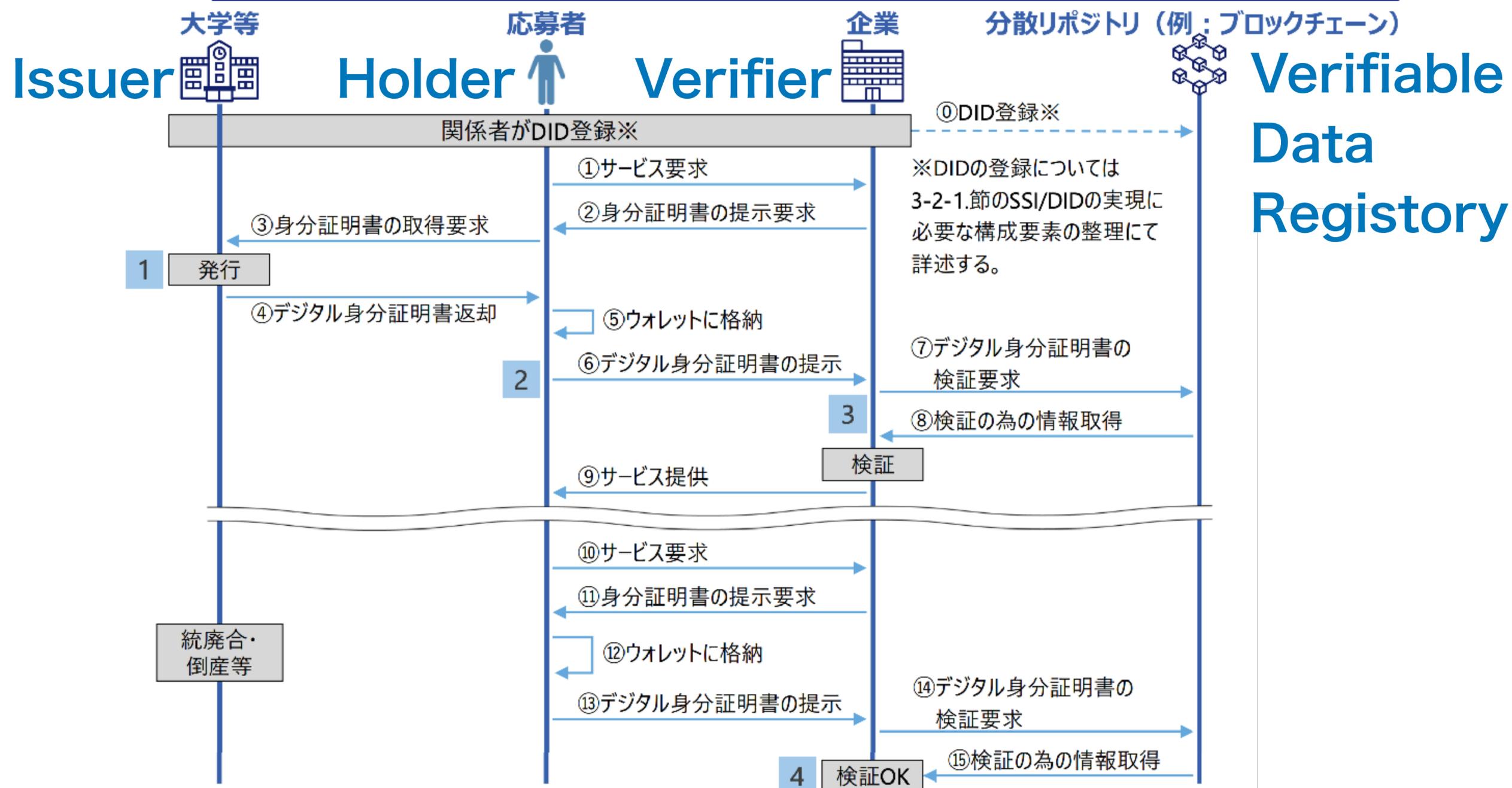
DID/VC

- DID/VCのモデル



DID/VC

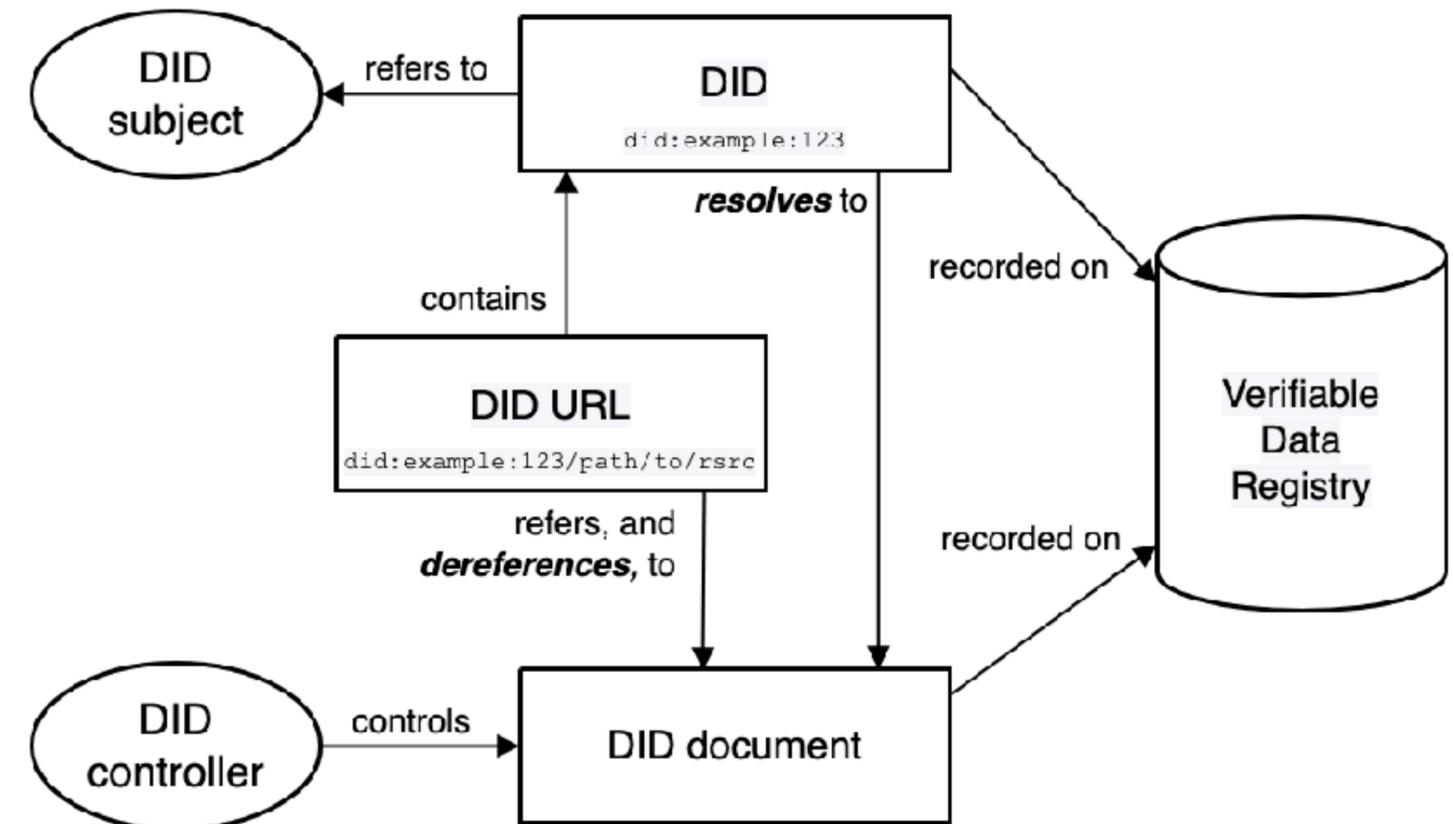
大学生が、企業に学歴証明を行う例



DID

DIDのアーキテクチャ

- DID: 識別子そのもの
- DID Subject: エンティティのこと
- DID Document:
DID Subjectについてのデータの集合。
Subjectの公開鍵などを含む
- DID Controller:
DIDドキュメントに対して変更を加える
ことのできるエンティティ



DIDのシンタックス

- DIDのシンタックス: 

The diagram shows the DID `did:example:123456789abcdefghi` with annotations: `did` is labeled "Scheme", `example` is labeled "DID Method", and `123456789abcdefghi` is labeled "DID Method-Specific Identifier".
- DID URLのシンタックス: `did path ["?" query] ["#" fragment]`
 - ▶ Path: Pathのセマンティクスはmethodごと定義
 - ▶ Query: いくつか定義されたクエリを使用できます(本資料では触れず)
<https://www.w3.org/TR/did-core/#did-parameters>
 - ▶ Fragment: DID Documentの中の要素を指し示したりできる

DID Documentのデータモデル

DID Documentの例

- id: DID
- 他にも様々なフィールドを書くことができる
 - ▶ authentication/
assertionMethod/
verificationMethodフィールド
に公開鍵を置いておける

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ]
  "id": "did:example:123456789abcdefghi",
  "authentication": [{
    // used to authenticate as did:...fghi
    "id": "did:example:123456789abcdefghi#keys-1",
    "type": "Ed25519VerificationKey2020",
    "controller": "did:example:123456789abcdefghi",
    "publicKeyMultibase": "zH3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
  }]
}
```

did:web

did:web

- いくつもあるDID Methodの一つ
- DNS, World Wide Webを使う

did:web

DIDフォーマット

- did:web:domain-name[:port][:path]
- 例:
 - ▶ did:web:kekeho.net
 - ▶ did:web:kekeho.net:hoge
 - ▶ did:web:kekeho.net%3A3000:fuga:piyo

did:web

Register

1. (ドメイン名レジストラにドメイン名の使用を申請する)
 - 例: Google Domainでkekeho.netを取得する
2. (DNSにWebサーバーのIPアドレスを登録)
 - 例: AWS EC2でサーバーを建て、そのIPアドレスをkekeho.netに紐付け
3. DIDドキュメントを作成し、以下のパスで公開
 - `did:web:kekeho.net` → <https://kekeho.net/.well-known/did.json>
 - `did:web:kekeho.net:hoge` → <https://kekeho.net/hoge/did.json>

did:web

Resolve

1. HTTP GETでdid.jsonにアクセス

• 例:

▶ did:web:kekeho.net → GET <https://kekeho.net/.well-known/did.json>

did:web

Update

1. did.jsonを更新

did:web

Revoke

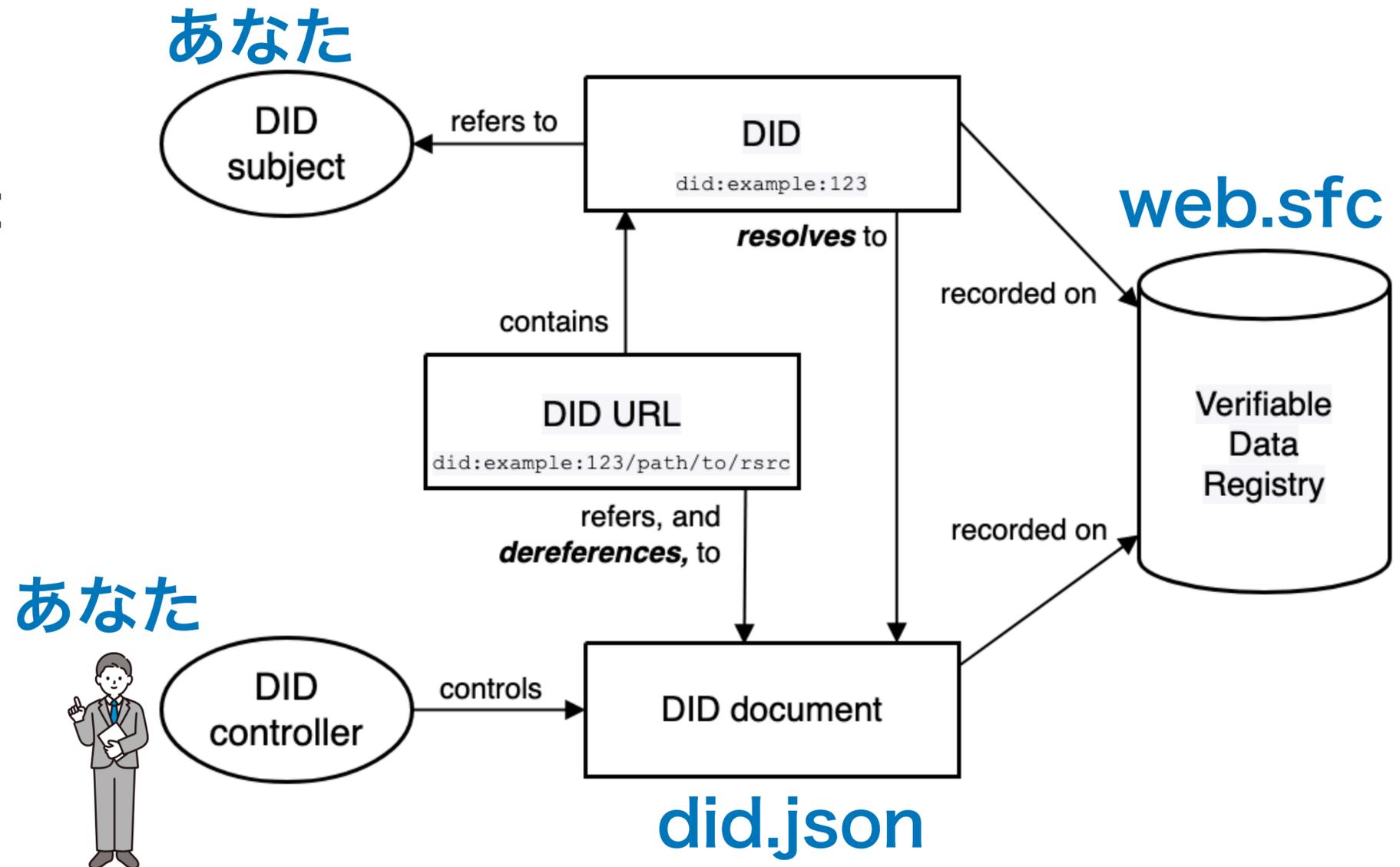
1. did.jsonを削除(あるいは何らかの方法で非公開に)

チュートリアル1

did:webで実際にDIDを発行してみよう

概要

- web.sfc.keio.ac.jpでDIDを発行
- 例: did:web:web.sfc.keio.ac.jp:
%7Et21440ht
- Register・Resolveをやります



Register

- 秘密鍵と公開鍵のペアを作成
 - ▶ `openssl genpkey -algorithm ed25519 -out private.pem`
 - ▶ `openssl pkey -in private.pem -pubout > public.pem`

Register

- did.jsonを書く

```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:web:web.sfc.keio.ac.jp:%7Et21440ht",
  "verificationMethod": [
    {
      "id": "did:web:web.sfc.keio.ac.jp:%7Et21440ht#key1",
      "type": "Ed25519VerificationKey2020",
      "controller": "did:web:web.sfc.keio.ac.jp:%7Et21440ht",
      "publicKeyMultibase": "mMCowBQYDK2VwAyEAww6wLKL8Sn2iMRtYljghfftfb3Ye2bsLBaLPkQGGtvs="
    }
  ]
}
```

Resolve

- Resolverとしてcurl + 適当なjsonパーサを使ってもいいですが…
- Universal Resolver
(<https://github.com/decentralized-identity/universal-resolver>)
 - ▶ 様々なDID Methodに対応したResolver
 - ▶ 手元でUniversal Resolverを動かさなくてもお手軽に使えるWebサービスを提供してくれている
<https://dev.uniresolver.io/>

Resolve

 **DIF Universal Resolver** Configuration 

Supported methods: did:ace did:ala did:bba did:bid did:btcr did:ccp did:cheqd did:com did:dns did:dock did:dyne did:ebsi did:elem did:emtrust did:ens did:eosio did:ethr did:ev did:evan did:everscale did:factom did:gatc did:github did:hcr did:icon did:iid did:indy did:io did:ion did:iscc did:jolo did:jwk did:key did:kilt did:kscirc did:lit did:meta did:moncon did:mydata did:nacl did:ont did:orb did:oyd did:pkh did:schema did:sol did:sov did:stack did:tz did:unisot did:v1 did:vaa did:web did:work [Contribute a driver?](#)

did-url Resolve Clear Examples

Copy link to result

RESULT DID DOCUMENT RESOLUTION METADATA DOCUMENT METADATA

Parser

did	method	method-specific-id	path-abempty	query	fragment
did:web:web.sfc.keio.ac.jp:%7Et21440ht	web	web.sfc.keio.ac.jp:%7Et21440ht			

Services
(none)

Verification Methods

 **Ed25519VerificationKey2020**
did:web:web.sfc.keio.ac.jp:%7Et21440ht#key1
mMCowBQYDK2VwAyEAww6wLKL8Sn2iMRtYljghfftb3Ye2bsLbaLPkQGGtvs=

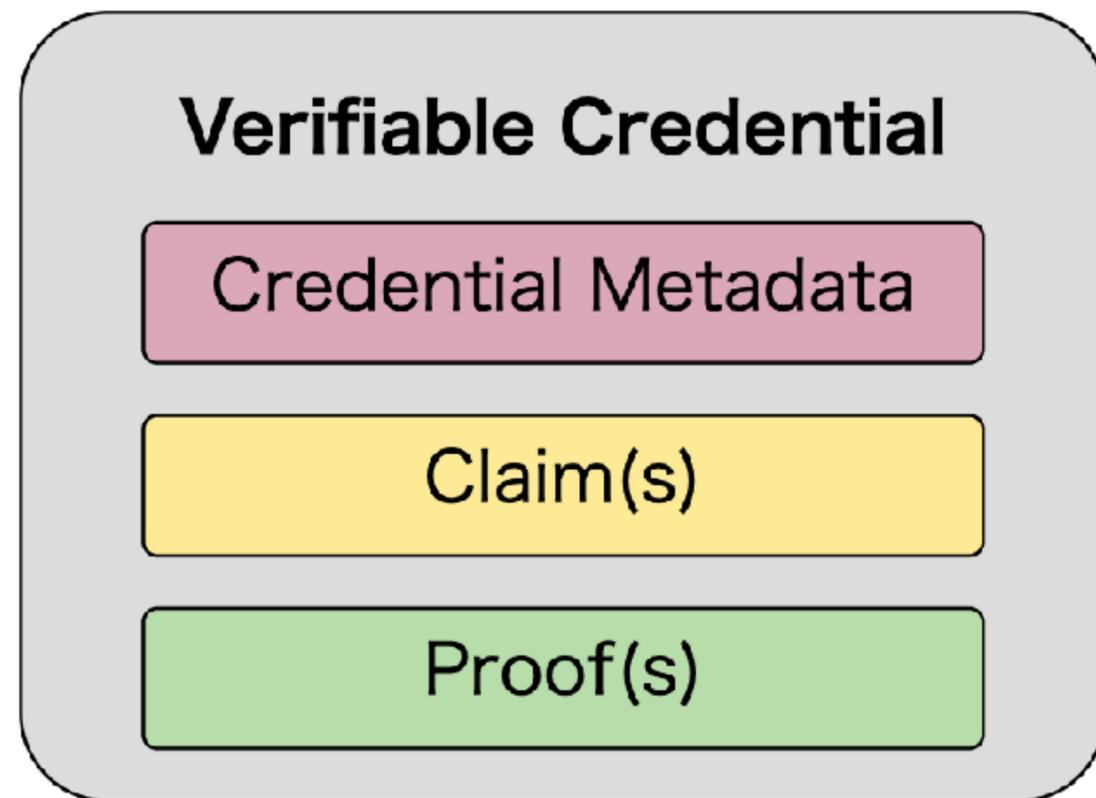


Verifiable Credential

Verifiable Credential

データモデル

- W3CのVCsでは、Credentialのデータモデルを定義している



Credentialの例

```
{
  "@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": { ... }
}
```

チュートリアル2

Verifiable Credentialを発行してみよう

チュートリアル

- Data RegistryとしてGitHub Gistを使うデモがあるので、動かしてみましよう

<https://github.com/digitalbazaar/vc-demo>

参考文献

1. ISO/IEC 24760-1:2019
IT Security and Privacy — A framework for identity management — Part 1: Terminology and concepts
<https://www.iso.org/standard/77582.html>
2. ブロックチェーン技術等を用いたデジタルアイデンティティの活用に関する研究 | NRI
https://www.fsa.go.jp/policy/bgin/ResearchPaper_NRI_ja.pdf
3. Decentralized Identifiers (DIDs) v1.0 | W3C
<https://www.w3.org/TR/did-core/>
4. did:web Method Specification | W3C CCG
<https://w3c-ccg.github.io/did-method-web/>
5. did:webをGitHub Pagesで使う | chike0905の日記
<https://chike0905.hatenablog.com/entry/2022/04/06/134446>