Réseaux fédérés avec ActivityPub et Webfinger

Initiation aux protocoles et retour d'expérience de Funkwhale

À propos de moi

- Développeur Python depuis ~5ans
- Travaille @PeopleDoc
- Mainteneur de Funkwhale

Contact :

- @eliotberriot@mastodon.eliotberriot.com
- contact@eliotberriot.com
- https://eliotberriot.com

Funkwhat?

Funkwhale

- Serveur de musique sous licence AGPL
- Fortement influencé par Grooveshark

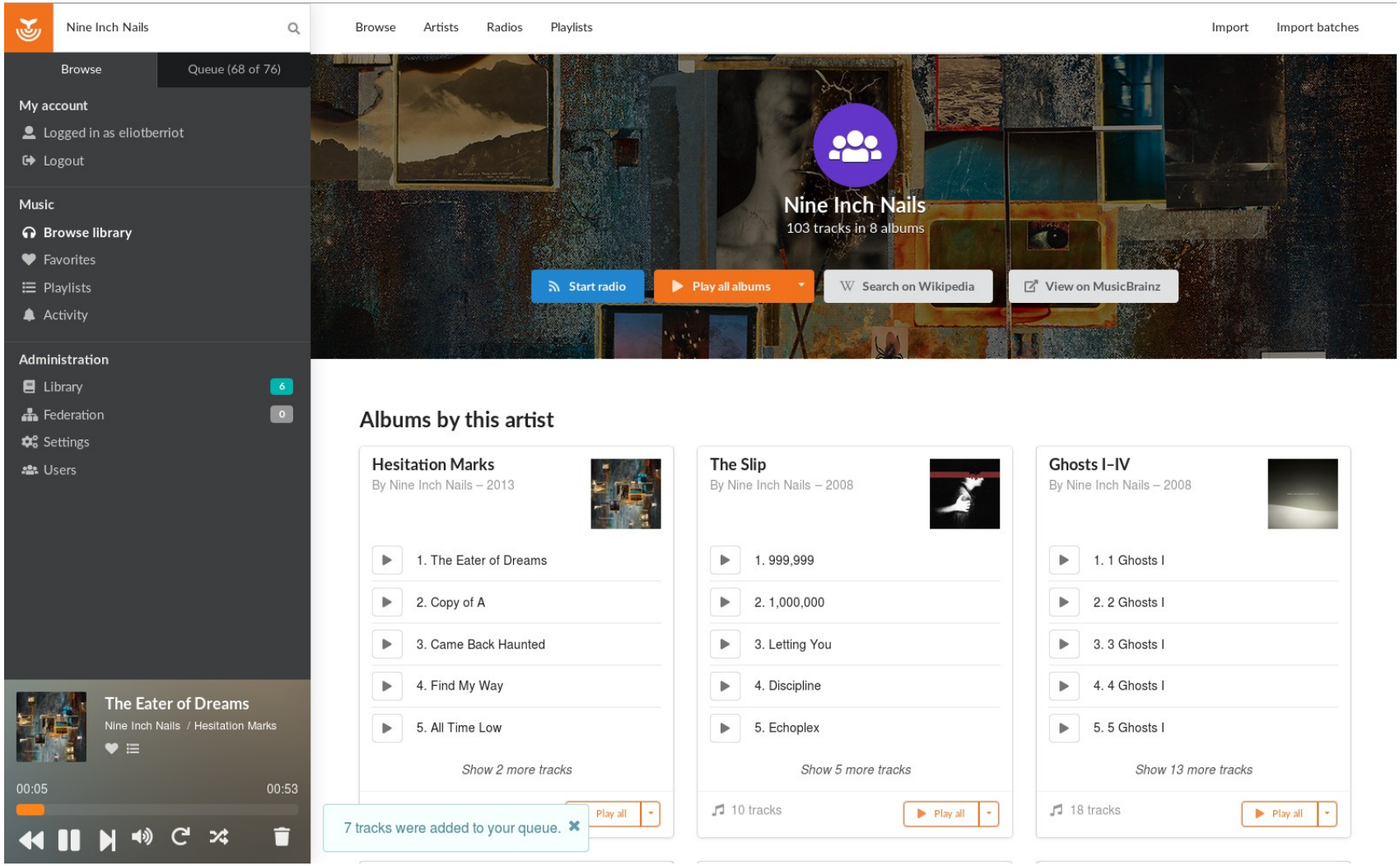
https://funkwhale.audio/

https://docs.funkwhale.audio

https://demo.funkwhale.audio

https://code.eliotberriot.com/funkwhale/funkwhale

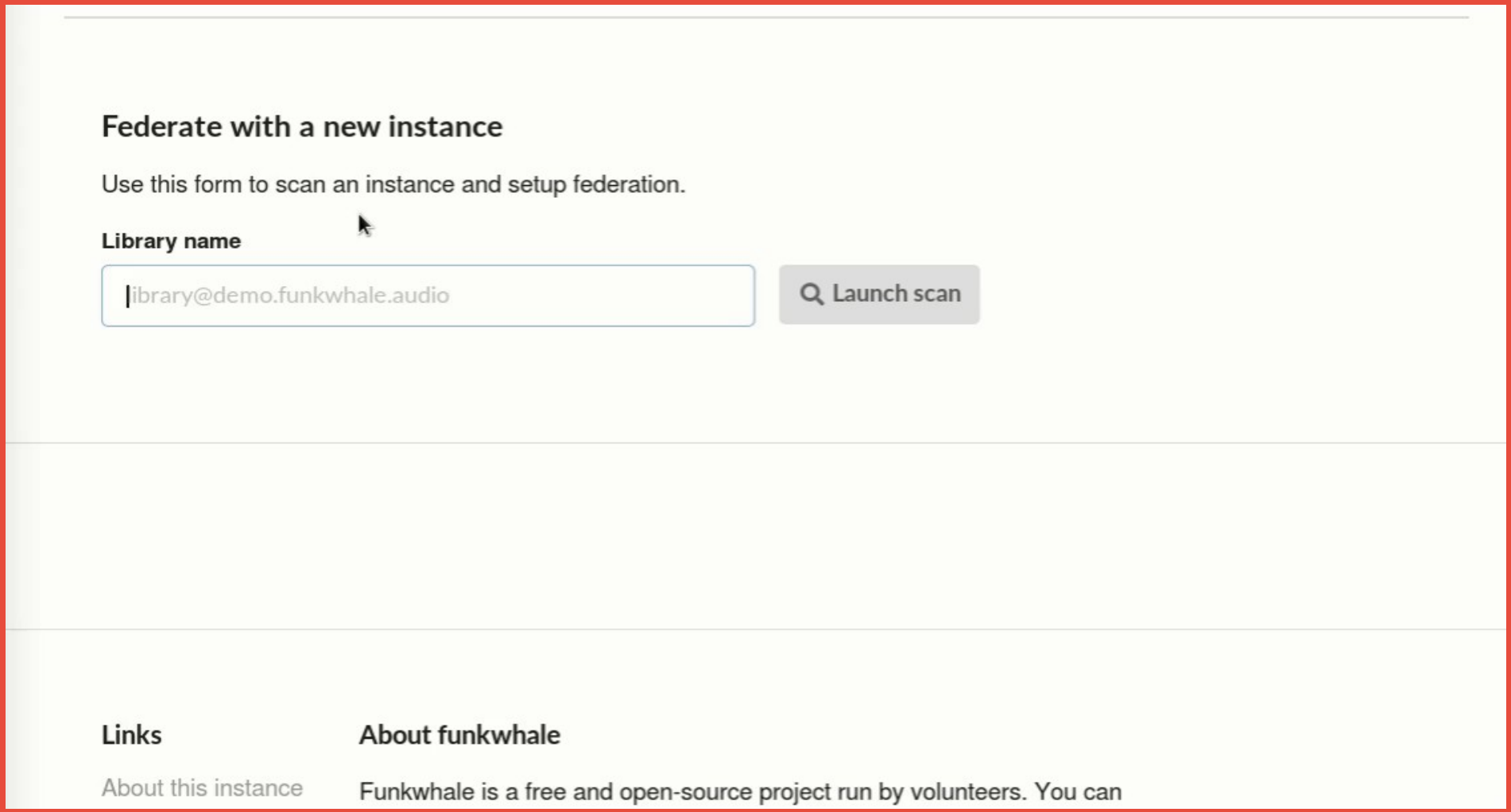
Funkwhale


Nine Inch Nails
Browse
Queue (68 of 76)
My account
Logged in as eliotberriot
Logout
Music
Browse library
Favorites
Playlists
Activity
Administration
Library 6
Federation 0
Settings
Users
The Eater of Dreams
Nine Inch Nails / Hesitation Marks
00:05
00:53
Browse
Artists
Radios
Playlists
Import
Import batches
Nine Inch Nails
103 tracks in 8 albums
Start radio
Play all albums
Search on Wikipedia
View on MusicBrainz
Albums by this artist
Hesitation Marks
By Nine Inch Nails – 2013
1. The Eater of Dreams
2. Copy of A
3. Came Back Haunted
4. Find My Way
5. All Time Low
Show 2 more tracks
Play all
The Slip
By Nine Inch Nails – 2008
1. 999,999
2. 1,000,000
3. Letting You
4. Discipline
5. Echoplex
Show 5 more tracks
10 tracks
Play all
Ghosts I–IV
By Nine Inch Nails – 2008
1. 1 Ghosts I
2. 2 Ghosts I
3. 3 Ghosts I
4. 4 Ghosts I
5. 5 Ghosts I
Show 13 more tracks
18 tracks
Play all
7 tracks were added to your queue.

La fédération dans Funkwhale

- Échange de musique entre instances
- Restrictif par défaut
- Pas encore de fédération des activités utilisateurs

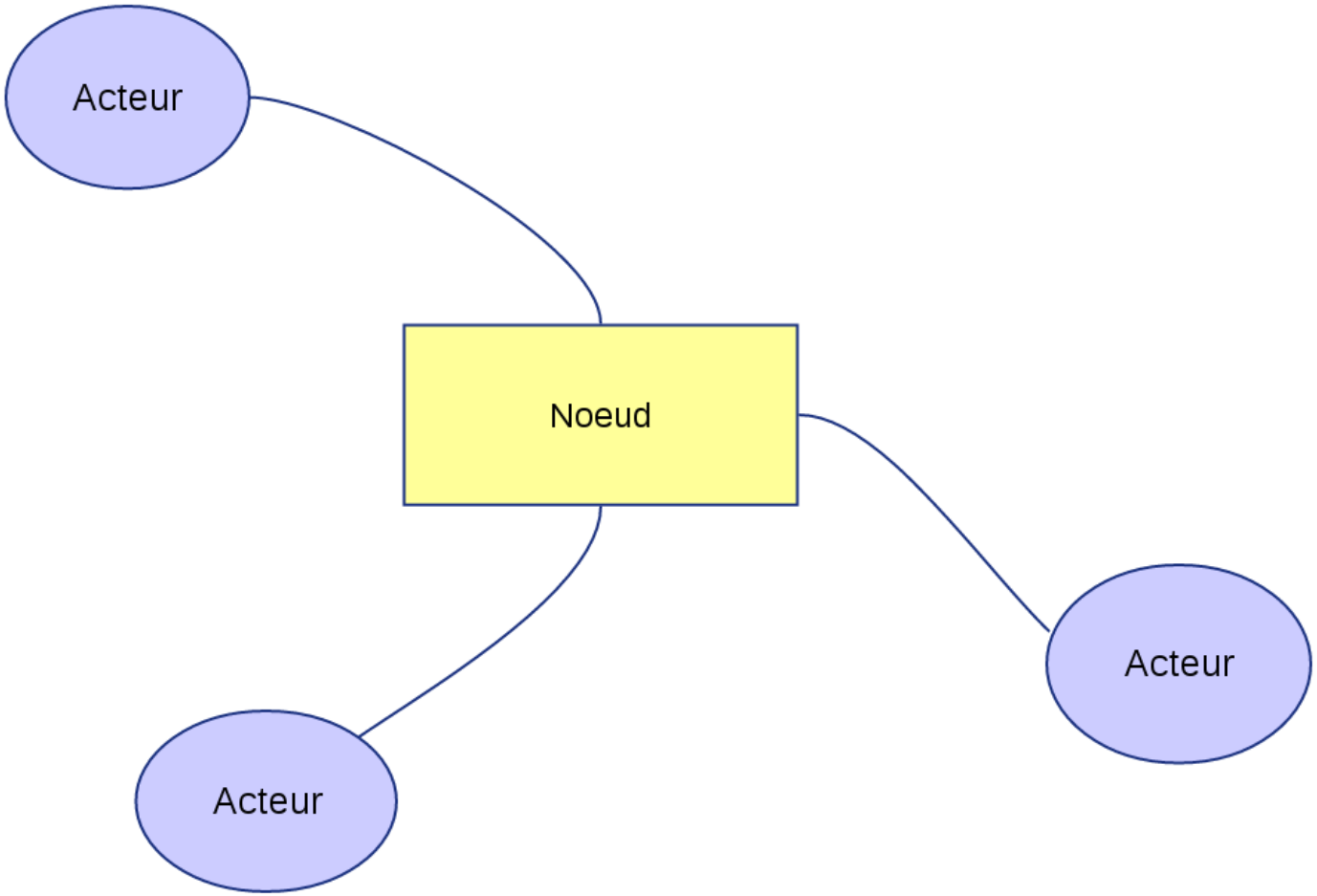
Fédération : une vidéo qui dénonce



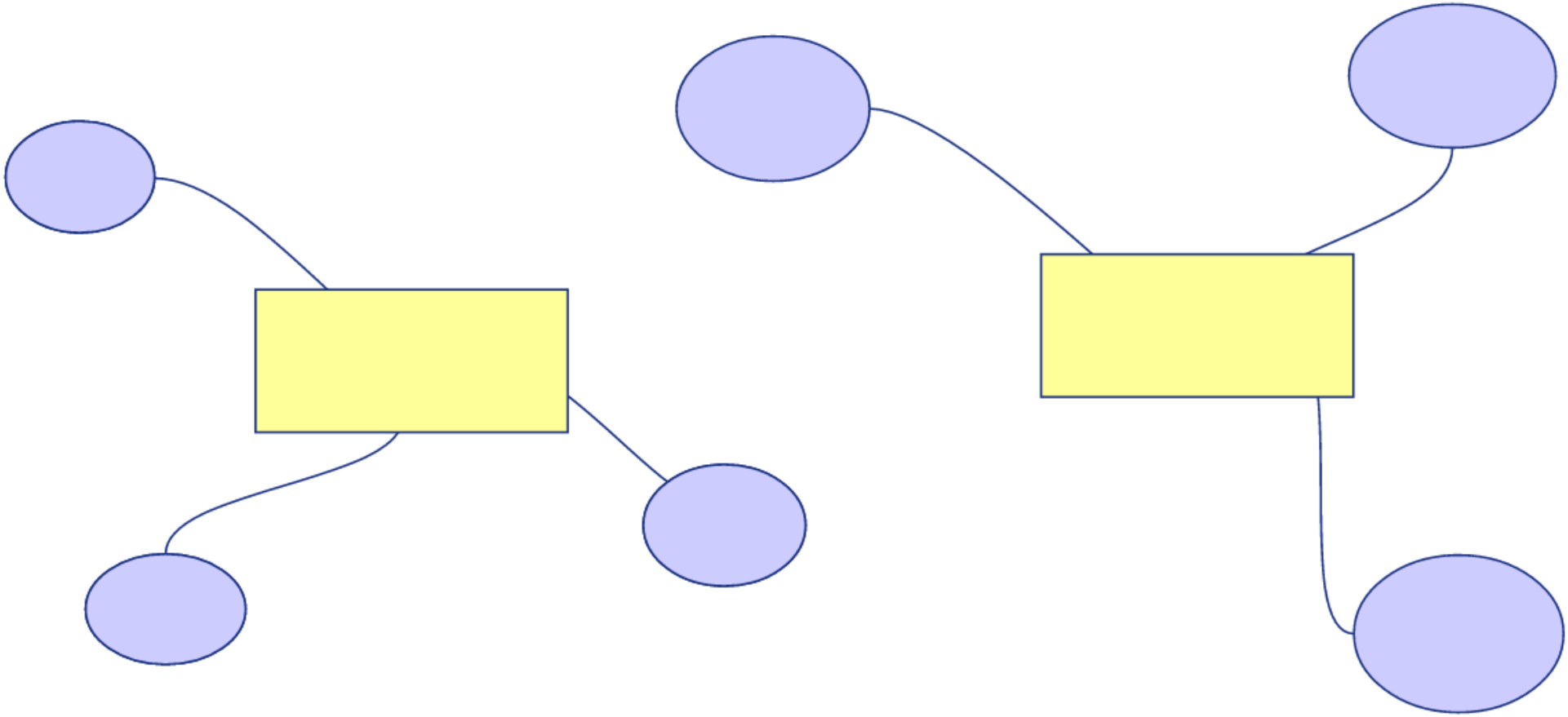
Réseaux centralisés et fédérés

Afin de comprendre de quoi on parle...

Un réseau centralisé dans son habitat naturel



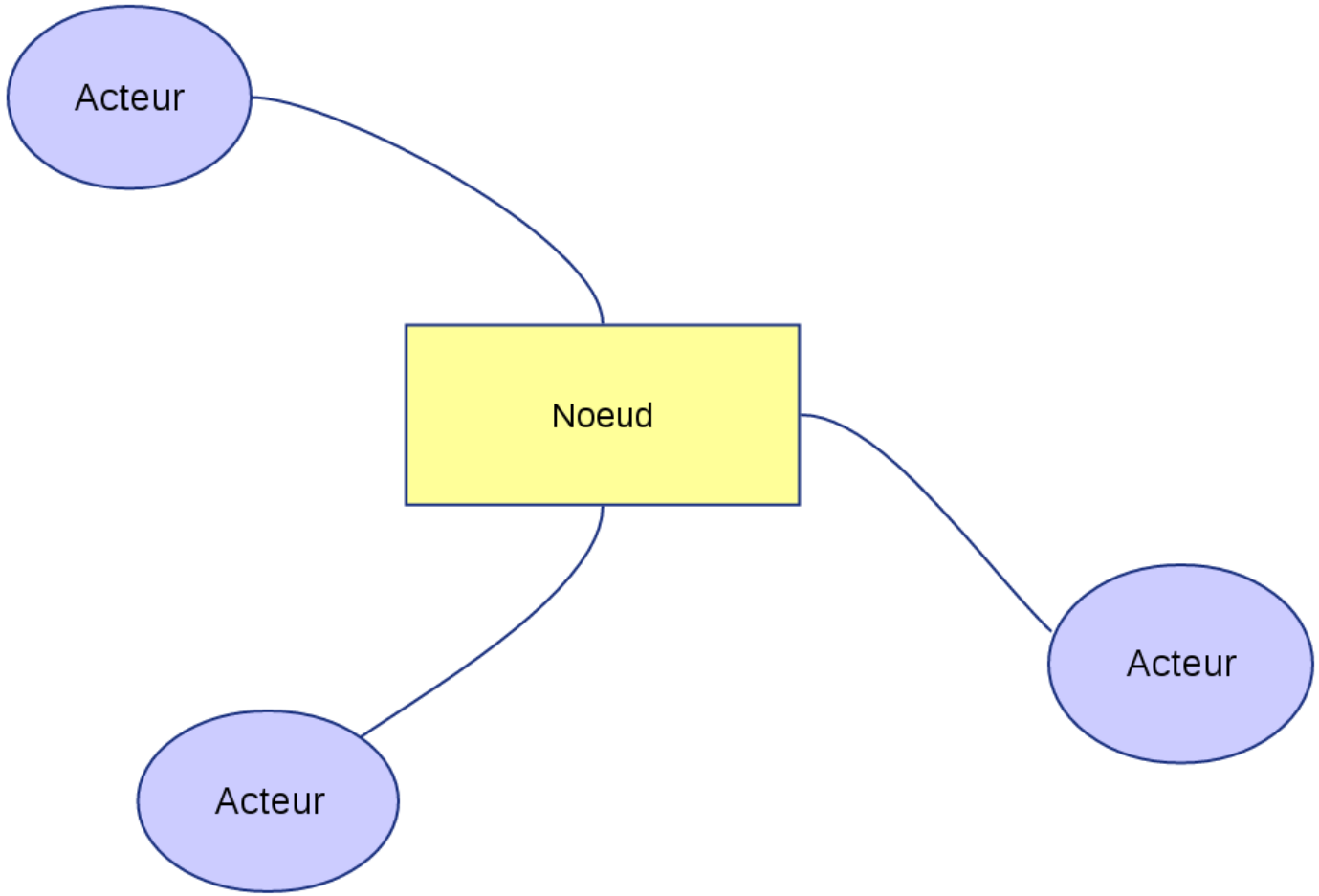
Deux réseaux centralisés dans leur habitat naturel



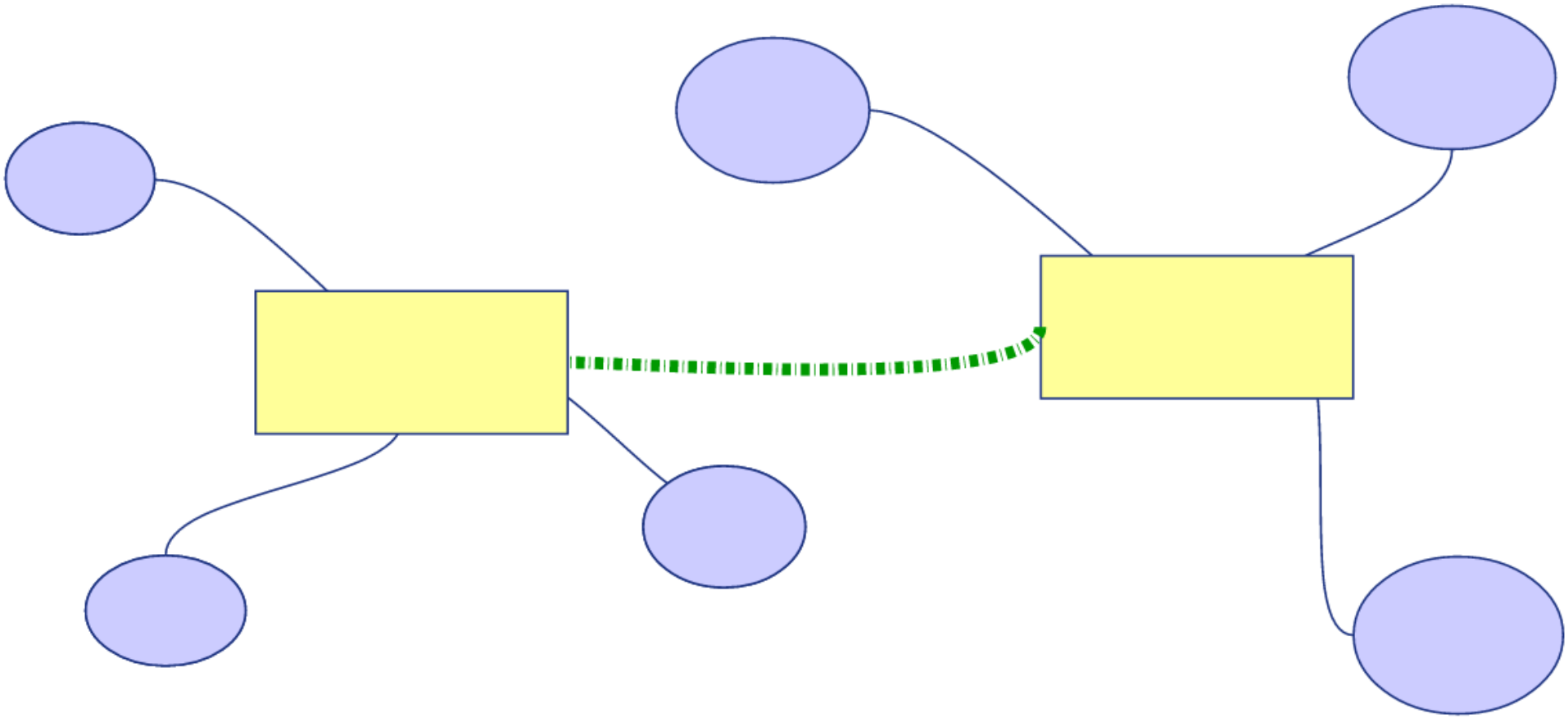
Conclusions

- Ça ne se parle pas beaucoup
- Tout fonctionne grâce à un point central

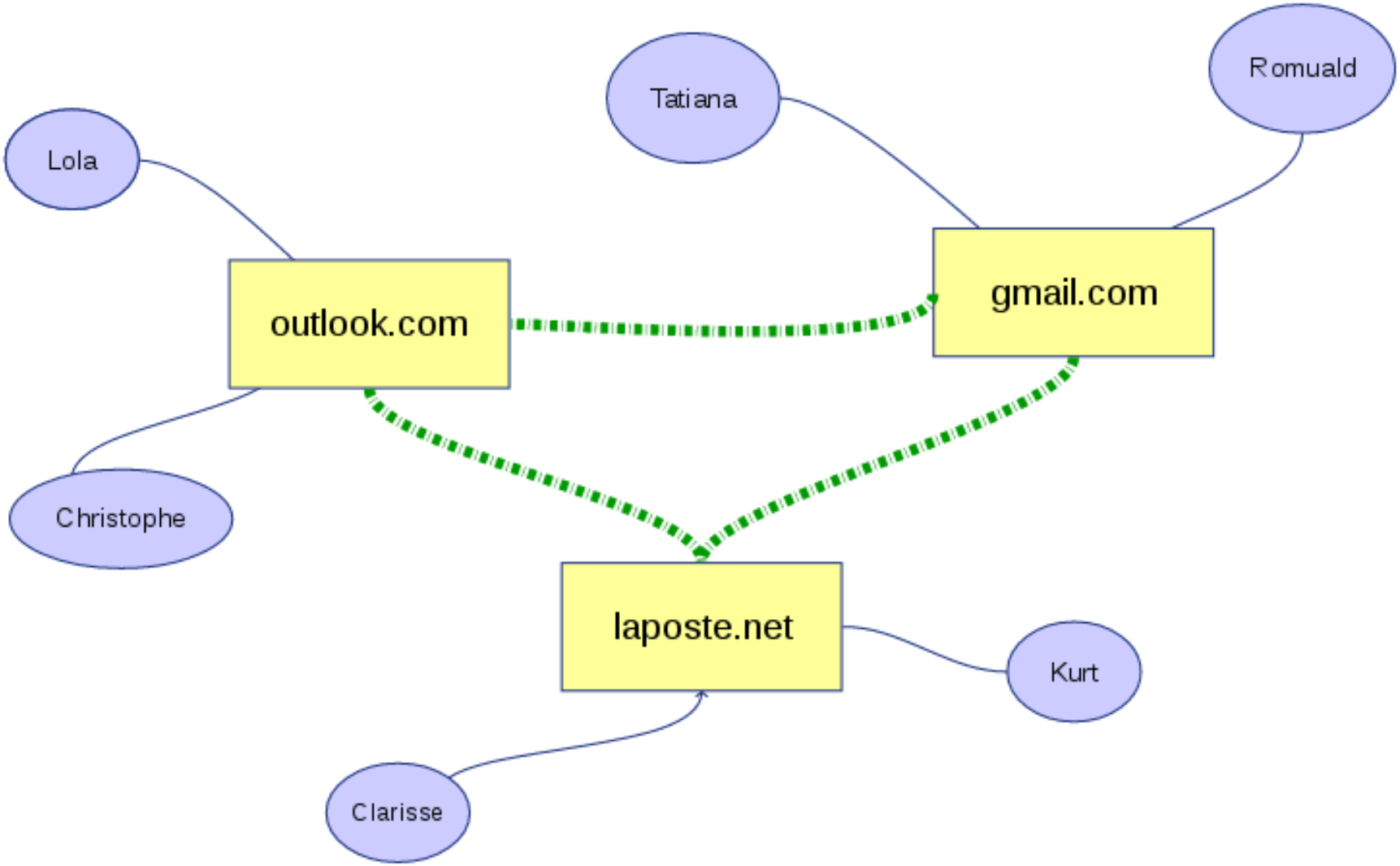
Un noeud d’un réseau fédéré dans son habitat naturel



Deux noeuds d’un réseau fédéré dans leur habitat naturel



Exemple de réseau fédéré



Conclusions

- Ça se parle beaucoup plus
- Le réseau a plusieurs centres
- La fédération peut-être une question d'échelle

Comparaison

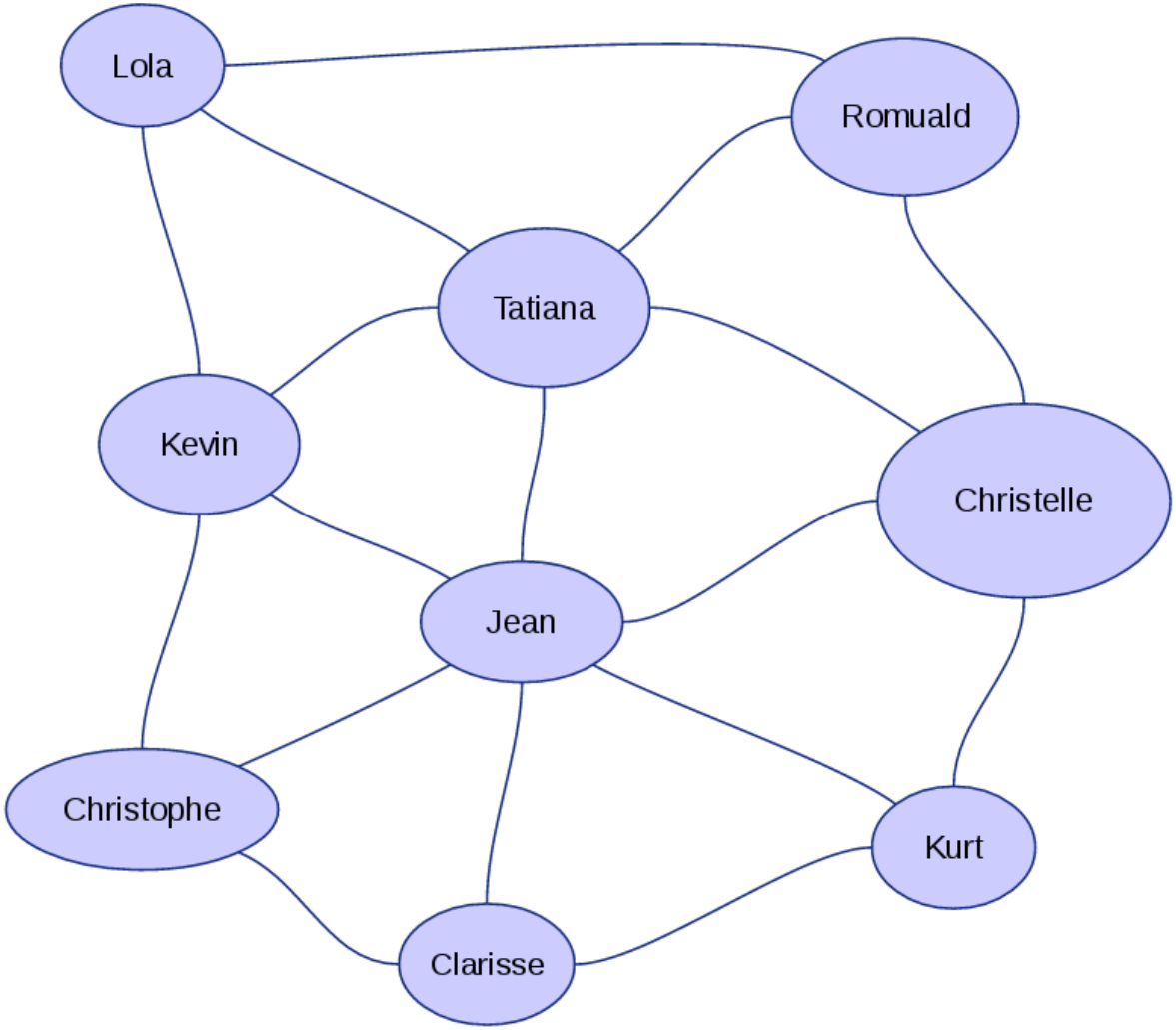
Centralisé

- Un seul noeud
- Un acteur contrôle le fonctionnement du réseau
- Plus fragile
- Moins complexe

Fédéré

- Noeuds à volonté
- Aucun noeud n'est indispensable
- Plus résilient
- Plus complexe

Bonus : les réseaux pair-à-pair



Comment construire une fédération ?

De la nécessité d’un language commun

Les protocoles

- Indispensable à la fédération
- Détermine le fonctionnement du réseau
  - Performances
  - Fonctionnalités
  - Limites
  - Extensibilité
- Un protocole, c'est comme une langue humaine

Role d’un protocole

- Découverte des acteurs et noeuds
- Transmission des messages
- Authentification des acteurs
- Structure des messages
- Vocabulaire
- Autres règles eventuelles

Un bon protocole...

C’est comme une langue, c’est :

- Facile à comprendre
- Facile à parler
- Capable de véhiculer ce que l’on cherche à dire
- Capable d’évoluer
- Parlé par beaucoup de monde !

Aparté sur Funkwhale

- Intégrer/construire une fédération fait peur
- Apprendre à parler une nouvelle langue
- Mis de côté en attendant un signe du destin

Arrive ActivityPub

- Protocole standardisé officiellement par le W3C
- Implémenté par Mastodon
- Semble adapté à de nombreux usages

https://www.w3.org/TR/activitypub/

Un protocole, vraiment ?

ActivityPub réutilise des standards existants :

- ActivityStreams
- JSON-LD
- Webfinger
- HTTP Signatures
- HTTP

Structure et contenu des messages

C'est bien gentil tout ça, mais on s'envoie quoi ?

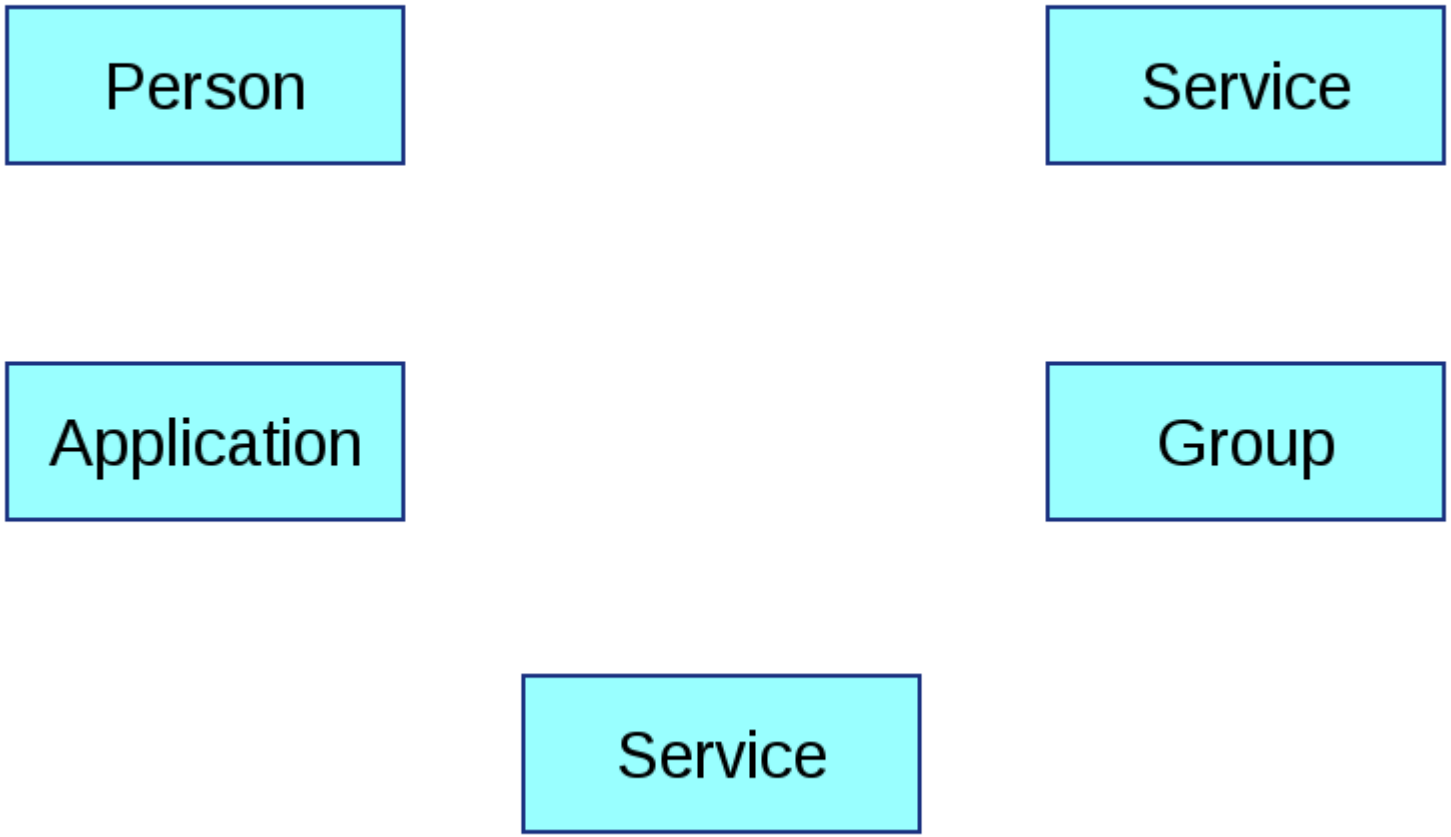
Acteurs, activités et objets

- Le vocabulaire de base est décrit dans Activity Streams
- Les acteurs réalisent des activités associées à des objets

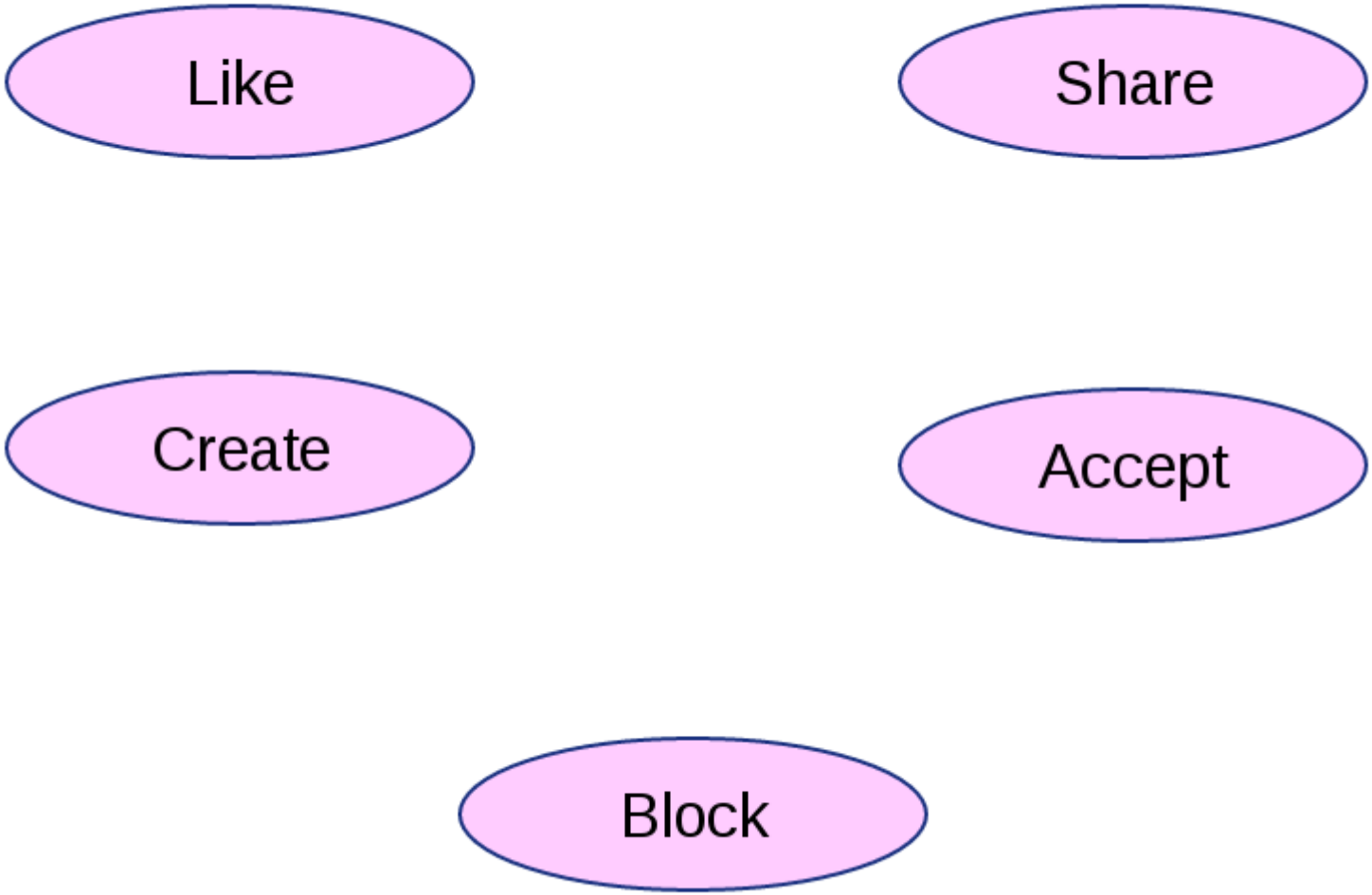
https://www.w3.org/TR/activitystreams-core/

https://www.w3.org/TR/activitystreams-vocabulary

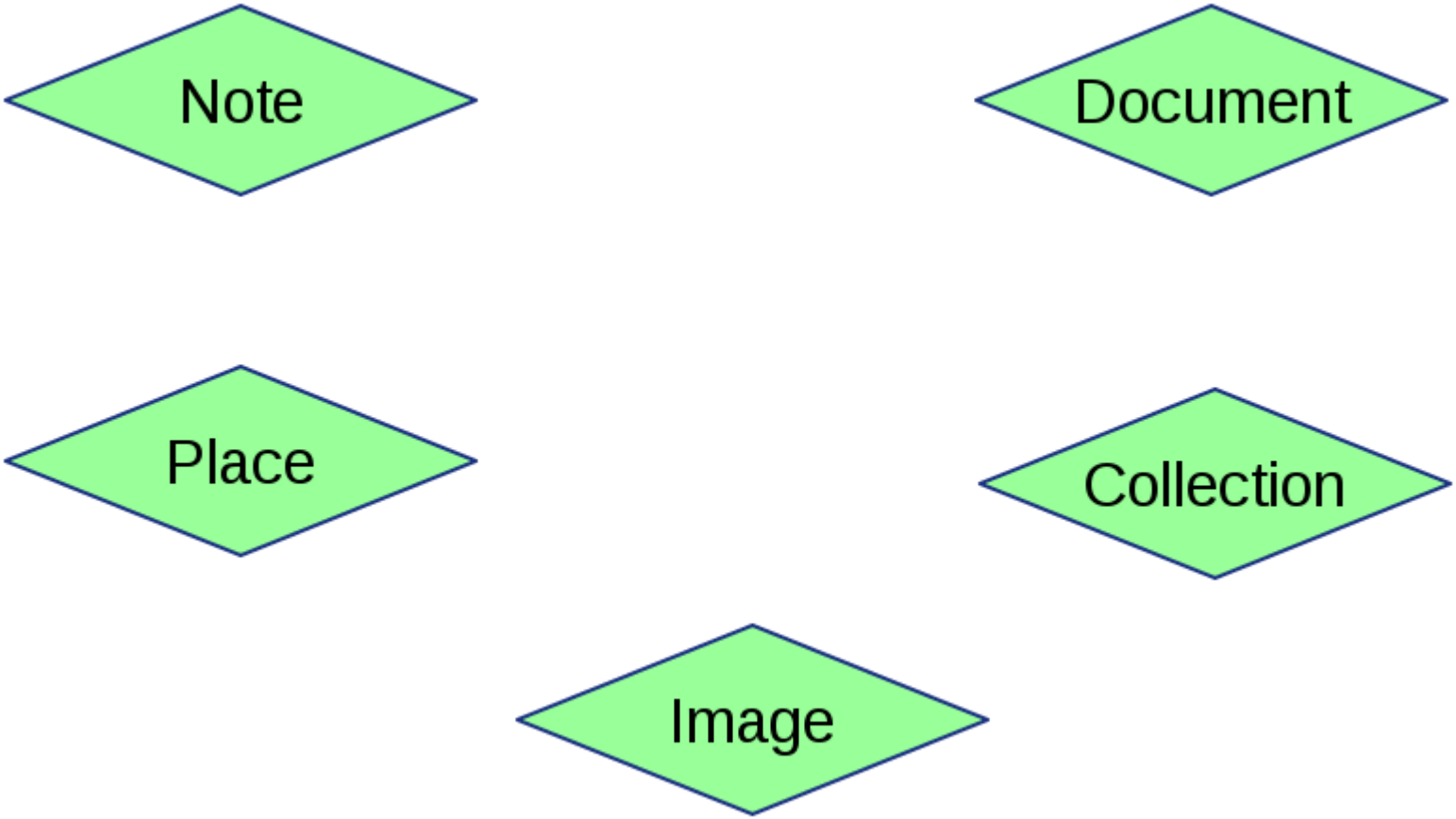
Acteurs



Activités (liste non-exhaustive)

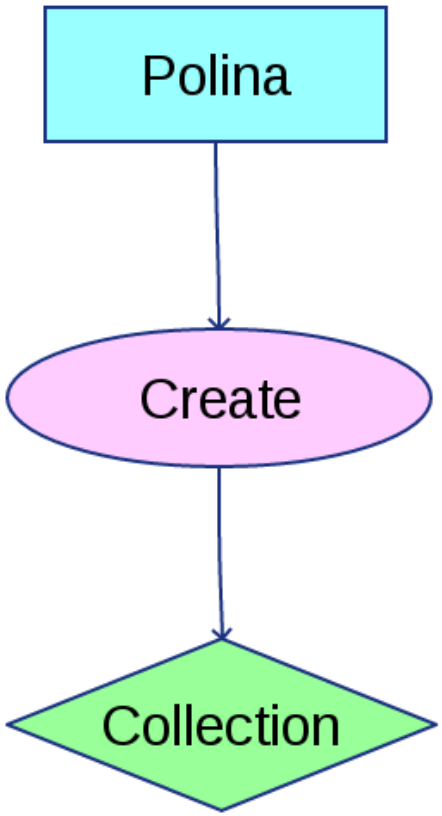


Objets (liste non-exhaustive)

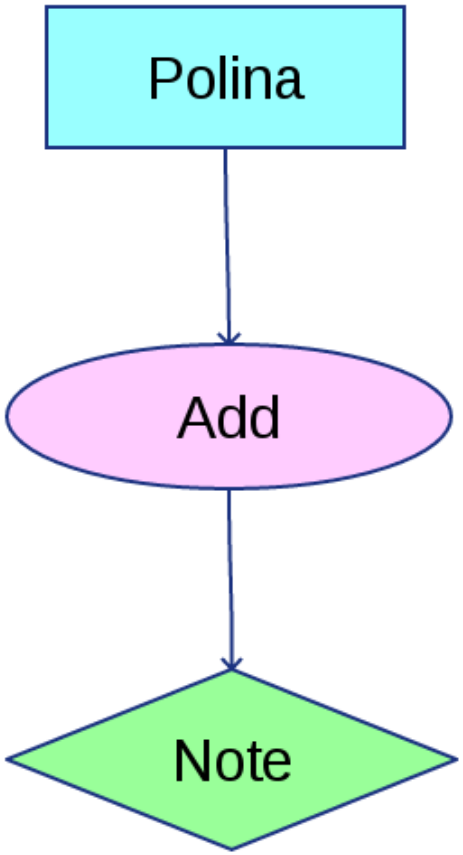


On mélange tout…

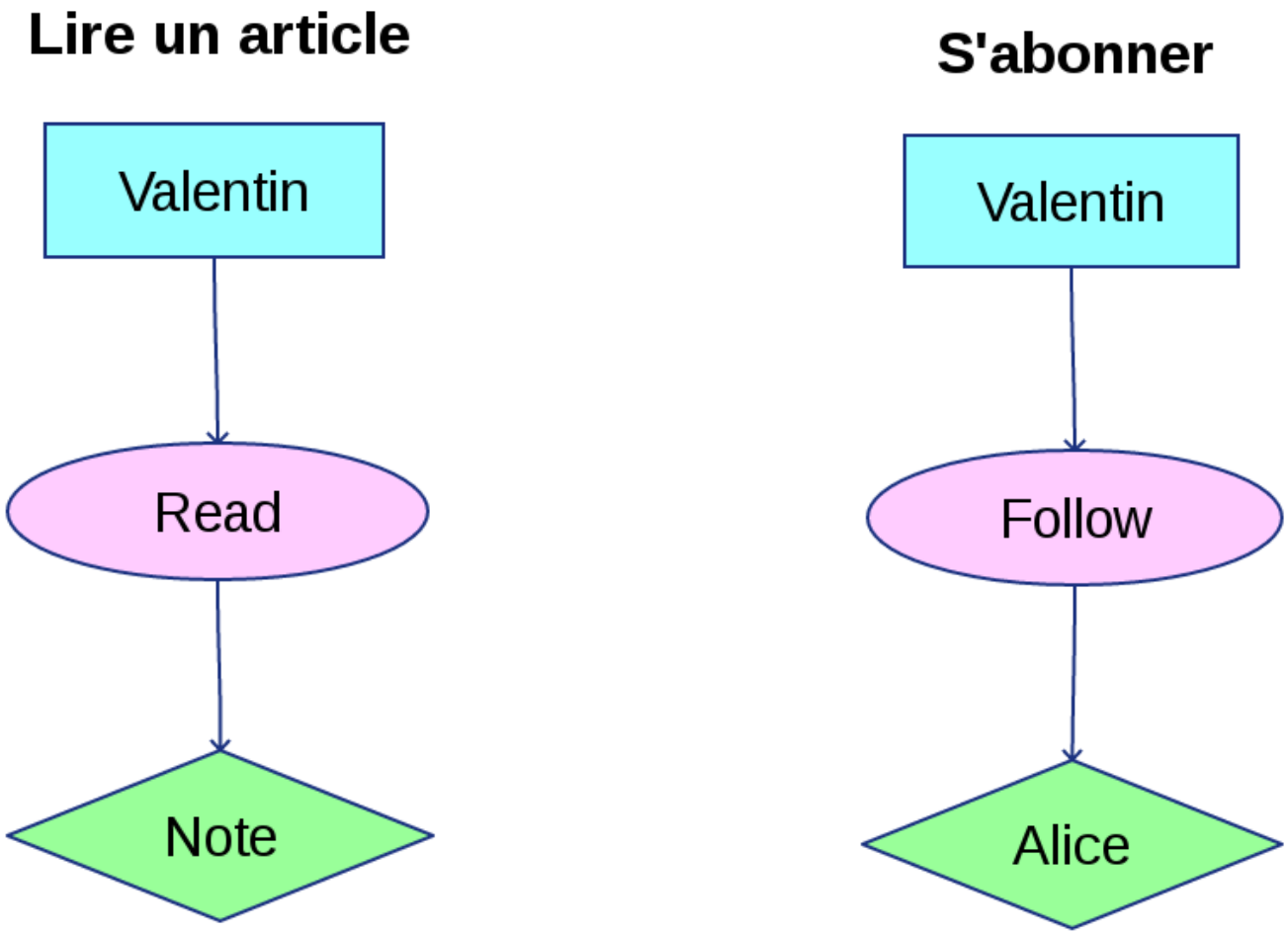
Modéliser un blog



Publier un article



On mélange tout… (2)



Conclusions

- On peut modéliser énormément de choses en combinant les objets et les activités
- On peut grouper les objets grâce aux collections
- Conceptuellement simple

Représentation des messages

Le modèle abstrait ne suffit pas, il faut une syntaxe pour pouvoir l'écrire.

JSON-LD

Un format de données lisible et extensible pour représenter des données liées.

https://json-ld.org/spec/latest/json-ld/

JSON ?

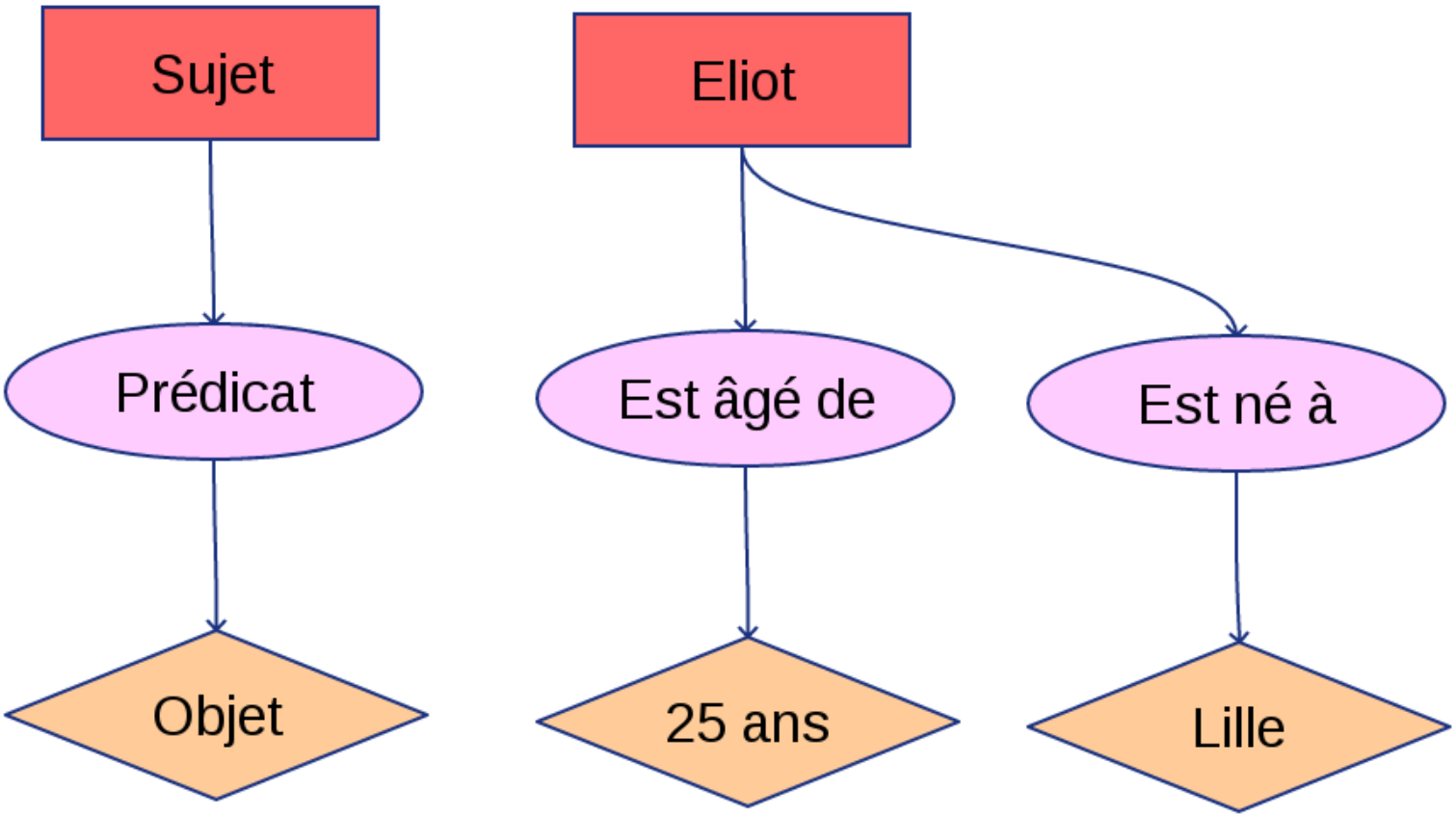
```json
{
    "nom": "Eliot",
    "age": 25,
    "aime": ["manger", "dormir"],
    "content": true,
    "projets": [
        {
            "nom": "Funkwhale",
            "url": "https://funkwhale.audio"
        },
        {
            "nom": "MNM",
            "url": "https://mnm.social"
        }
    ]
}
```

Linked data ?

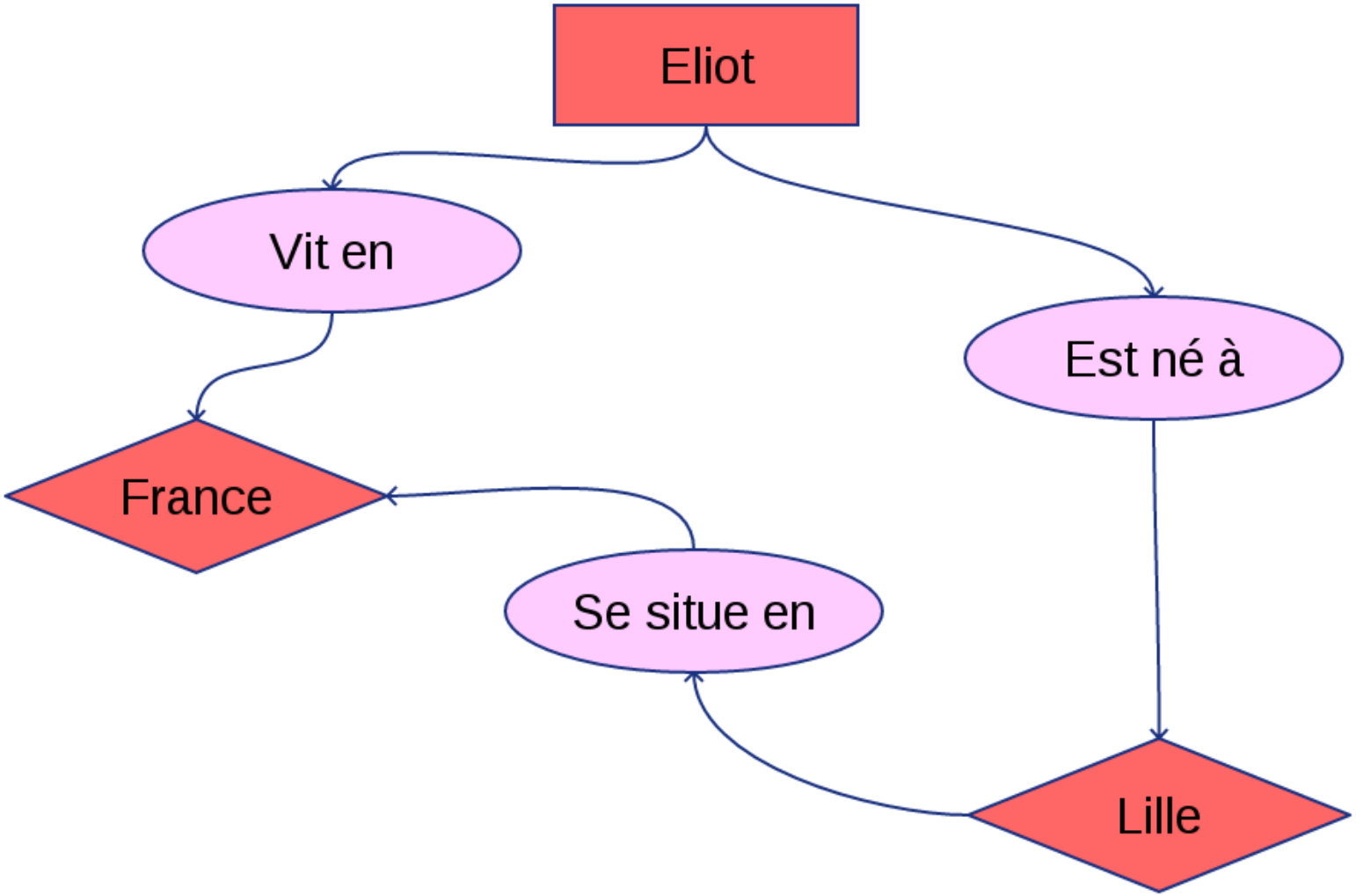
- Rendre les données compréhensibles aux machines
- Associer des données situées dans des applications distinctes
- Partager les mêmes vocabulaires

Linked data ? (2)

Un changement de paradigme : sujet, prédicat, objet



Linked data ? (3)



Linked data ? (4)

Pour se comprendre, il faut des vocabulaires communs :

- ActivityStreams pour les activité
- Dublin Core pour les documents
- FOAF pour les personnes
- Etc.

Linked data ? (5)

JSON-LD utilise les vocabulaires comme clés :

```
{
    "http://xmlns.com/foaf/0.1/name": "Eliot",
    "http://xmlns.com/foaf/0.1/age": 25,
    "http://xmlns.com/foaf/0.1/currentProject": {
        "http://xmlns.com/foaf/0.1/homepage": "https://funkwhale.audio"
    }
}
```

Linked data ? (6)

En plus lisible :

```
{
    "@context": [
        "http://xmlns.com/foaf/0.1/"
    ],
    "name": "Eliot",
    "age": 25,
    "currentProject": {
        "homepage": "https://funkwhale.audio"
    }
}
```

Conclusions

- Un format (relativement simple)
- Extensible
- Permet d'utiliser des vocabulaires existants

La découverte

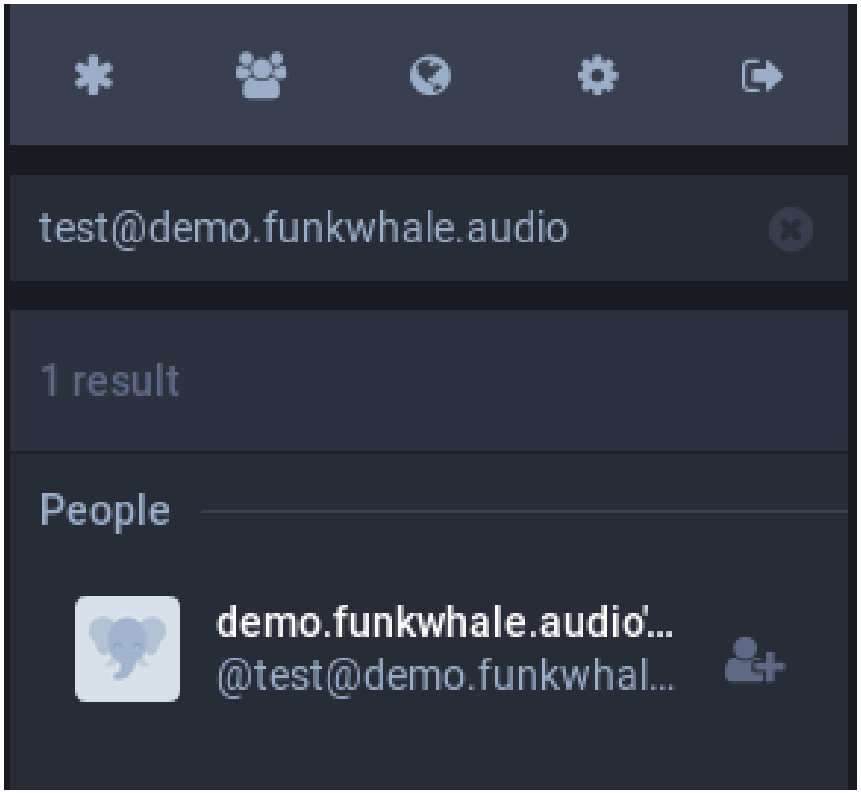
Webfinger :

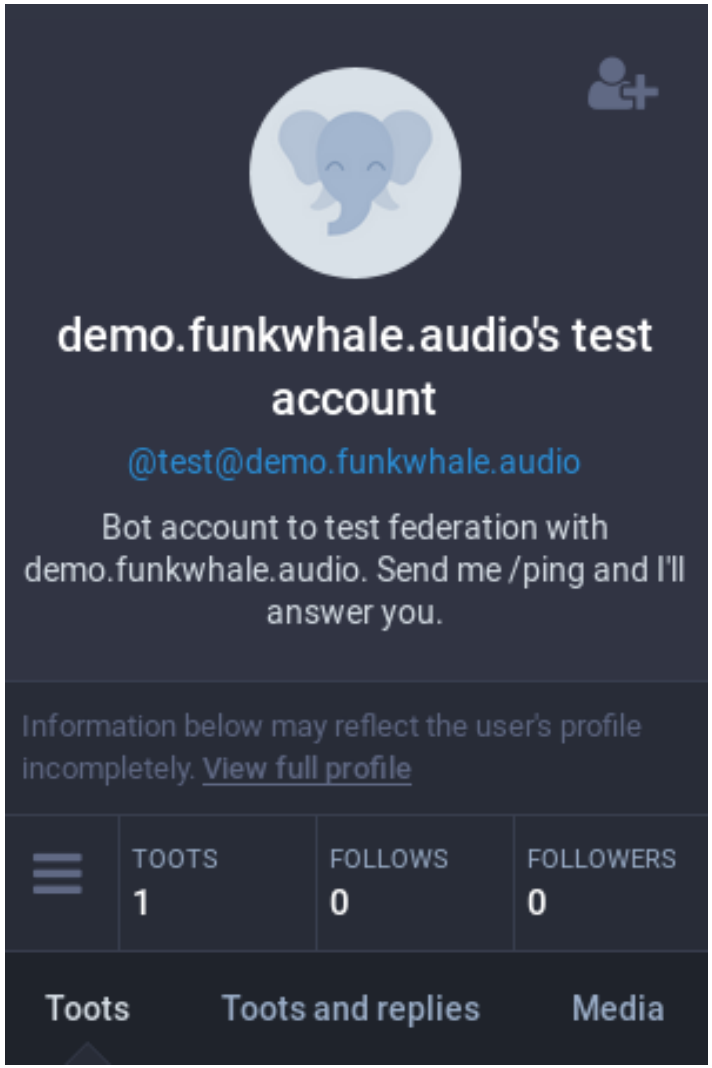
- Inspiré du mail
- user@domaine
- Pluggable avec n'importe quel protocole

https://tools.ietf.org/html/rfc7033

http://domaine/.well-known/webfinger?resource=acct:user@domaine

Utilisation de Webfinger dans Mastodon





Une réponse Webfinger

Acteur : test@demo.funkwhale.audio

URL webfinger : https://demo.funkwhale.audio/.well-known/webfinger?resource=acct:test@demo.funkwhale.audio


```
{
  "subject": "acct:test@demo.funkwhale.audio",
  "links": [
    {
      "rel": "self",
      "href": "https://demo.funkwhale.audio/federation/instance/actors/test",
      "type": "application/activity+json"
    }
  ],
  "aliases": ["https://demo.funkwhale.audio/federation/instance/actors/test"]
}
```

La transmission

- Chaque acteur ActivityPub possède une inbox
- Cette inbox est accessible via une URL
- On envoie un message en HTTP à cette URL

La transmission (2)

Pour trouver l'inbox, on va chercher la représentation de l'acteur


```
{
    "id": "https://demo.funkwhale.audio/federation/instance/actors/test",
    "inbox": "https://demo.funkwhale.audio/federation/instance/actors/test/inbox",
    "preferredUsername": "test",
    "type": "Person",
    "name": "demo.funkwhale.audio's test account",
    "summary": "Bot account to test federation with demo.funkwhale.audio. Send me /ping and I'll answer you.",
}
```

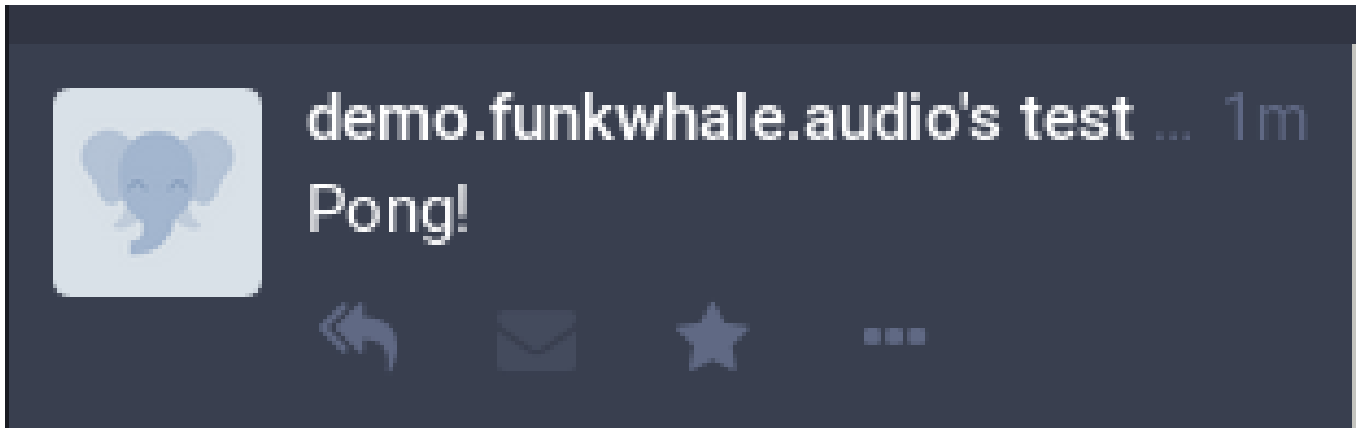
Le message

```
{
    "actor": "https://hello.world/@donna",
    "type": "Create",
    "object": {
        "type": "Note",
        "id": "https://hello.world/@donna/posts/1",
        "content": "/ping"
    }
}
```

La transmission (3)

- On envoie le message sur l'URL d'inbox
- On utilise la méthode POST
- On utilise le header HTTP Content-Type:

  application/activity+json



Et la sécurité, dans tout ça ?

- Comment s'assurer que le message reçu est bien envoyé par l'acteur annoncé ?
- Comment éviter les attaques de type replay ?

HTTP Signatures

- Pas spécifié dans ActivityPub, standard de-facto
- Basé sur la cryptographie asymétrique
- On signe les requêtes envoyées

https://tools.ietf.org/id/draft-cavage-http-signatures-01.html

Rappel sur la cryptographie asymétrique

- Alice possède une clé privée et une clé publique
- Bob signe le message avec la clé publique d'Alice
- Alice déchiffre le message avec sa clé privée

Récupérer la clé publique d'un acteur

- ActivityPub à la rescousse
- On peut l'inclure dans la représentation de l'acteur


```
{
    "id": "https://demo.funkwhale.audio/federation/instance/actors/library",
    "outbox": "https://demo.funkwhale.audio/federation/instance/actors/library/outbox",
    "inbox": "https://demo.funkwhale.audio/federation/instance/actors/library/inbox",
    "preferredUsername": "library",
    "type": "Person",
    "publicKey": {
        "owner": "https://demo.funkwhale.audio/federation/instance/actors/library",
        "publicKeyPem": "-----BEGIN RSA PUBLIC KEY-----\nredacted-----END RSA PUBLIC KEY-----\n",
        "id": "https://demo.funkwhale.audio/federation/instance/actors/library#main-key"
    },
}
```

Validation d’une signature

1. Je reçois un message d’un acteur
2. La requête HTTP inclut un header Signature
3. Je vais chercher la clé publique de l’acteur du message
4. Je vérifie que la signature est correcte

Exemple de signature

keyId="https://my-example.com/actor#main- key",

headers="(request-target) host date",

signature="..."

Et pour les attaques replay ?

- On inclut la date de création de la requête dans un header
- On inclut ce header dans la signature
- Le serveur qui reçoit le message s'assure que la date est suffisamment proche

Si on récapitule

1. On écrit des messages avec JSON-LD
2. On trouve où les envoyer grâce à Webfinger et ActivityPub
3. On les signe avec HTTP Signatures

Bonus : les abonnements

- Envoyer des messages, c'est bien
- Recevoir des messages qui nous intéressent, c'est mieux
- L'objet Follow est spécifié
- L'envoi des messages aux followers est spécifié

Revenons à Funkwhale

- Chaque acteur possède une bibliothèque
- Les acteurs suivent d'autres acteurs
- Un acteur peut refuser un abonné
- Un acteur notifie ses abonnés lors d'un ajout dans sa bibliothèque

C'est tout.

Liens et ressources utiles

Guide: https://blog.joinmastodon.org/2018/06/how-to-implement-a-basic-activitypub-server/

Implémentations :

- https://github.com/Plume-org/Plume
- https://github.com/Chocobozzz/PeerTube
- https://github.com/Natoum30/mightynetwork

Merci pour votre écoute !

Avez-vous des questions ? Des remarques ?

Les slides seront publiées sur https://eliotberriot.com