FOSS4G fr
Paris 2018

LA BLOCKCHAIN APPLIQUEE A L'INFORMATION GEOGRAPHIQUE

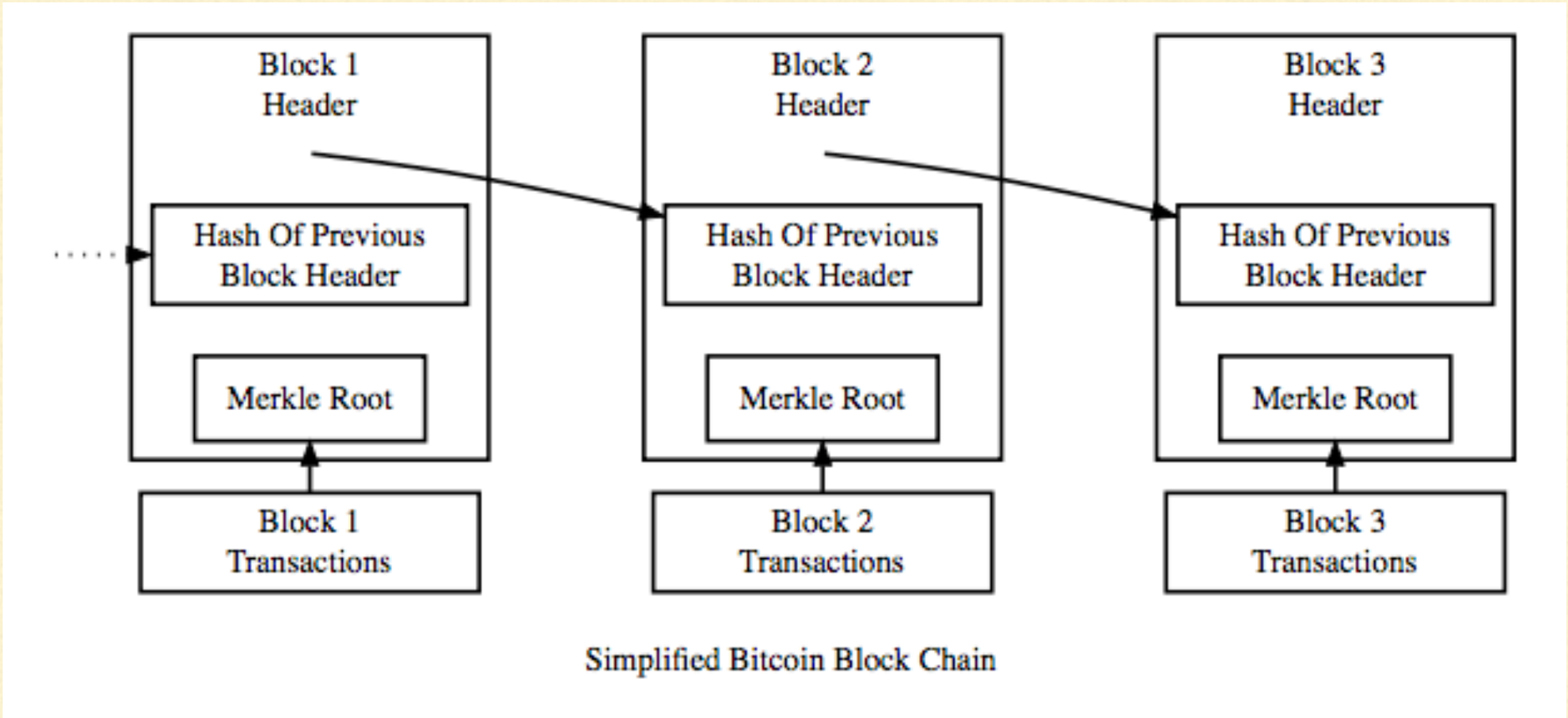
Présenté par Guillaume SUEUR
16 mai 2018

Neogeo technologies

La blockchain est une technologie informatique issue de la cryptographie permettant de tenir des **registres ouverts, décentralisés et partagés**.

C’est une base de données **distribuée,**
répartie entre chacun des utilisateurs.

Les blockchains remplacent l'autorité de l'officier par **l'infalsifiabilité** de leur structure



Simplified Bitcoin Block Chain

et l'utilisation d'une **preuve** contrôlable par chacun :

- Preuve de travail (Proof of work) : minage, très coûteux en énergie.
- Preuve d'enjeu (Proof of Stake) : cautionnement
- Preuve d'enjeu déléguée (Delegated Proof of Stake)
- Preuve de permission (Permissioned Blockchain)

Exemple de mise en oeuvre

Exemple d'une petite implémentation de blockchain en python / Flask

```
# Blockchain initialisation
blockchain = [create_genesis_block()]
print("The blockchain now contains {0} block(s)".format(len(blockchain)))
previous_block = blockchain[0]
# Local transactions initialisation
this_nodes_transactions = []
```

Initialisation de la chaîne par la création du Genesis Block


```
"blockchain content": [
  {
    index: 0,
    previous: "0",
    data: {
      proof-of-work: 9,
      transactions: [ ],
      title: "Genesis Block"
    },
    hash: "a376c97cae5db436368064622dbe3f512f8f8696f5b4ac22c65912fcf0089f8e",
    timestamp: "2018-05-14T08:58:58.161108"
  }
]
```

Exemple de mise en oeuvre

Exemple d'une petite implémentation
de blockchain en python / Flask

Définition objet des Blocs

```python
import hashlib as hasher


class Block:
  def __init__(self, index, timestamp, data, previous_hash):
    self.index = index
    self.timestamp = timestamp
    self.data = data
    self.previous_hash = previous_hash
    self.hash = self.hash_block()

  def __str__(self):
      return "Hi, I'm block {0}".format(self.index)



  def hash_block(self):
    sha = hasher.sha256()
    sha.update((str(self.index) +
       str(self.timestamp) +
           str(self.data) +
           str(self.previous_hash)).encode())
    return sha.hexdigest()
```

Exemple de mise en oeuvre

Exemple d'une petite implémentation
de blockchain en python / Flask

```python
@node.route('/txn', methods=['POST'])
def transaction():
  if request.method == 'POST':
    # On each new POST request,
    # we extract the transaction data
    new_txion = request.get_json()

# Then we add the transaction to our list
this_nodes_transactions.append(new_txion)
res = {
    "result":"success",
    "message":"Transaction submitted at rank {0}".format(len(this_nodes_transactions))
}
return(json.dumps(res))
```

Gestion des transactions

Exemple de mise en oeuvre

Minage 1 : vérification du bloc précédent et génération de la Proof work

```python
@node.route('/mine', methods = ['GET'])
def mine():
# Test exisente of transaction in the transaction heap
  if len(this_nodes_transactions) == 0:
      return json.dumps({
          "error": "No transaction to proceed"
      })

  # test previous block proof-of-work
  last_block = blockchain[len(blockchain) - 1]
  last_proof = last_block.data.get('proof-of-work')
  if not validate_proof(last_proof):
      return json.dumps({
          "error": "Critical error : last block is invalid. "
      })
  # generate a proof-of-work for current block
  new_proof = proof_of_work(last_proof)
```

```python
def proof_of_work(last_proof):
  incrementor = last_proof +1
  while not (incrementor % 9 == 0 and incrementor % last_proof == 0):
    incrementor += 1
  return incrementor

def validate_proof(proof):
    previous_block = blockchain[len(blockchain) - 2]
    previous_proof = previous_block.data.get('proof-of-work')
    return proof % 9 == 0 and proof % previous_proof == 0
```

Exemple de mise en oeuvre

Minage II : Intégration des transactions à un nouveau bloc et du bloc à la chaîne

```
new_block_data = {
  "proof-of-work": new_proof,
  "transactions": list(this_nodes_transactions)
}
new_block_index = last_block.index + 1
new_block_timestamp = this_timestamp = date.datetime.n
last_block_hash = last_block.hash
# Empty transaction list
this_nodes_transactions[:] = []
# Now create the
# new block!
mined_block = Block(
  new_block_index,
  new_block_timestamp,
  new_block_data,
  last_block_hash
)
blockchain.append(mined_block)
```

Exemple de mise en oeuvre

Minage 11 : Affichage des blocks (exemple avec transactions financières)


```
{
    index: 3,
    previous: "44b0c1a22ac98b77edbf9a533592ae0c4153182becd2f7e7856845b336f23339",
  data: {
   transactions: [
     {
        to: "ben",
        amount: 4,
        from: "john"
     },
     {
        to: "boris",
        amount: 0,
        from: "ben"
     },
     {
        to: "jenny",
        amount: 6,
        from: "ingmar"
     },
     {
        to: "elise",
        amount: 1,
        from: "network"
     }
    ],
    proof-of-work: 72
  },
  hash: "e28fe8a46727ae2bfc5e5fd1730135bd875923835fb03e60b5e7c21db7e5bbd8",
  timestamp: "2018-05-14T09:00:10.707547"
},
```

```
{
  index: 7,
  previous: "548e1c80027b230e3e5a65d41855203e6871daa619d114ee137a7a8474aa6f03",
  data: {
    transactions: [
      {
        lat: 47.06382784624888,
        timestamp: "2018-05-14T09:04:05.826642",
        lon: 4.334819928405799,
        device_id: 10
      },
      {
        to: "simone",
        amount: 1,
        from: "network"
      }
    ],
    proof-of-work: 1152
  },
  hash: "75e13de03c25dfcf152a8a8ce24748414a866aabcb5d0fc2720f6353335eff10",
  timestamp: "2018-05-14T09:04:05.826808"
},
{
  index: 8,
  previous: "75e13de03c25dfcf152a8a8ce24748414a866aabcb5d0fc2720f6353335eff10",
  data: {
    transactions: [
      {
        lat: 46.98561565083391,
        timestamp: "2018-05-14T09:04:05.826915",
        lon: 1.0550093955824957,
        device_id: 5
      },
      {
        lat: 46.61269577251922,
        timestamp: "2018-05-14T09:04:05.826935",
        lon: -0.7219620809971137,
        device_id: 1
      },
      {
        to: "tatiana",
        amount: 1,
        from: "network"
      }
    ],
    proof-of-work: 2304
  },
  hash: "b6d9261cfa8e82a0bcb3c811130be611473444cd6519e10320c0e310fb8cee73",
  timestamp: "2018-05-14T09:04:05.827079"
},
```

```
{
  index: 1,
  previous: "2000fe7311195c59c3811b33e76185f50e7ac83fa63d073cdac528001665021d",
  data: {
    transactions: [
      {
        dataset_url: "https://www.myopendata.fr/datasets/e8158e7f-8bc7-4c69-a99b-fd61026718fd",
        author: "some.author@gmail.com",
        timestamp: "2018-05-14T09:34:58.408790",
        dataset_hash: "ecc477e0c56fb31ef6d57743c50c5dfdd7c47d2c4630f488f8507e8716731be9"
      },
      {
        dataset_url: "https://www.myopendata.fr/datasets/f042d2f4-0d0a-4030-992b-b4c25b2a3fef",
        author: "some.author@gmail.com",
        timestamp: "2018-05-14T09:34:58.408857",
        dataset_hash: "487a3d2a599ab8d3ce3b0622931dbba9f07e57d79b154cb925166c2f3c35d106"
      },
      {
        dataset_url: "https://www.myopendata.fr/datasets/89fc98bf-9566-428d-9fc4-d294830e791c",
        author: "some.author@gmail.com",
        timestamp: "2018-05-14T09:34:58.408914",
        dataset_hash: "b6fa7210e7b80ea05d1eb1146f8710161d4583aedea16596fc80e31f65abe840"
      },
      {
        dataset_url: "https://www.myopendata.fr/datasets/9e73da0e-7404-4ca9-b2cb-ba9b83015b68",
        author: "some.author@gmail.com",
        timestamp: "2018-05-14T09:34:58.408973",
        dataset_hash: "ace9005cff1abee9a30b3aaeaa3e2ce2d58c23f6f3a80ac0994cc204b3c11b37"
      },
      {
        dataset_url: "https://www.myopendata.fr/datasets/65a34372-01cf-4e83-a4b0-ce9c74d29092",
        author: "some.author@gmail.com",
        timestamp: "2018-05-14T09:34:58.409025",
        dataset_hash: "c44731bcd971d2ced7747116400932a1f34512e8d5a2c82c54dde8a45dd2ffe6"
      },
      {
        dataset_url: "https://www.myopendata.fr/datasets/e742376c-e7a9-4ac4-bbb0-55ada6441d9b",
        author: "some.author@gmail.com",
        timestamp: "2018-05-14T09:34:58.409077",
        dataset_hash: "6ffb94fdbf018bb19a2f0d5eee25b09a9d172aa7ba3245bf1c300a97c8c4f360"
      },
      {
        dataset_url: "https://www.myopendata.fr/datasets/9e43e2c1-3b79-42f5-aae8-09192691e4d7",
        author: "some.author@gmail.com",
        timestamp: "2018-05-14T09:34:58.409155",
        dataset_hash: "5479be40c104d8baeeddc9cbd8498eea760e05fdd2c1d1b57631fab166d94b7d"
      },
      {
        to: "john",
        amount: 1,
        from: "network"
      }
    ],
    proof-of-work: 18
  },
  hash: "8f5d69d319a7e5bfb89f8ad0ed4a33b3abdfec2c4df33575e0fc2031fa33ba57",
  timestamp: "2018-05-14T09:34:58.409203"
},
```

Cadastre Alsace - Moselle

GÉOGRANDEST

UNION EUROPÉENNE

CARTE | PROJET | RESSOURCES | AIDE

(0) | GUILLAUME SUEUR

Recherche

Commune

...

Préfixe / Section / Parcelle

Numéro de Document d'Arpentage

Adresse

Type de document

Date d'application

Document (PDF)

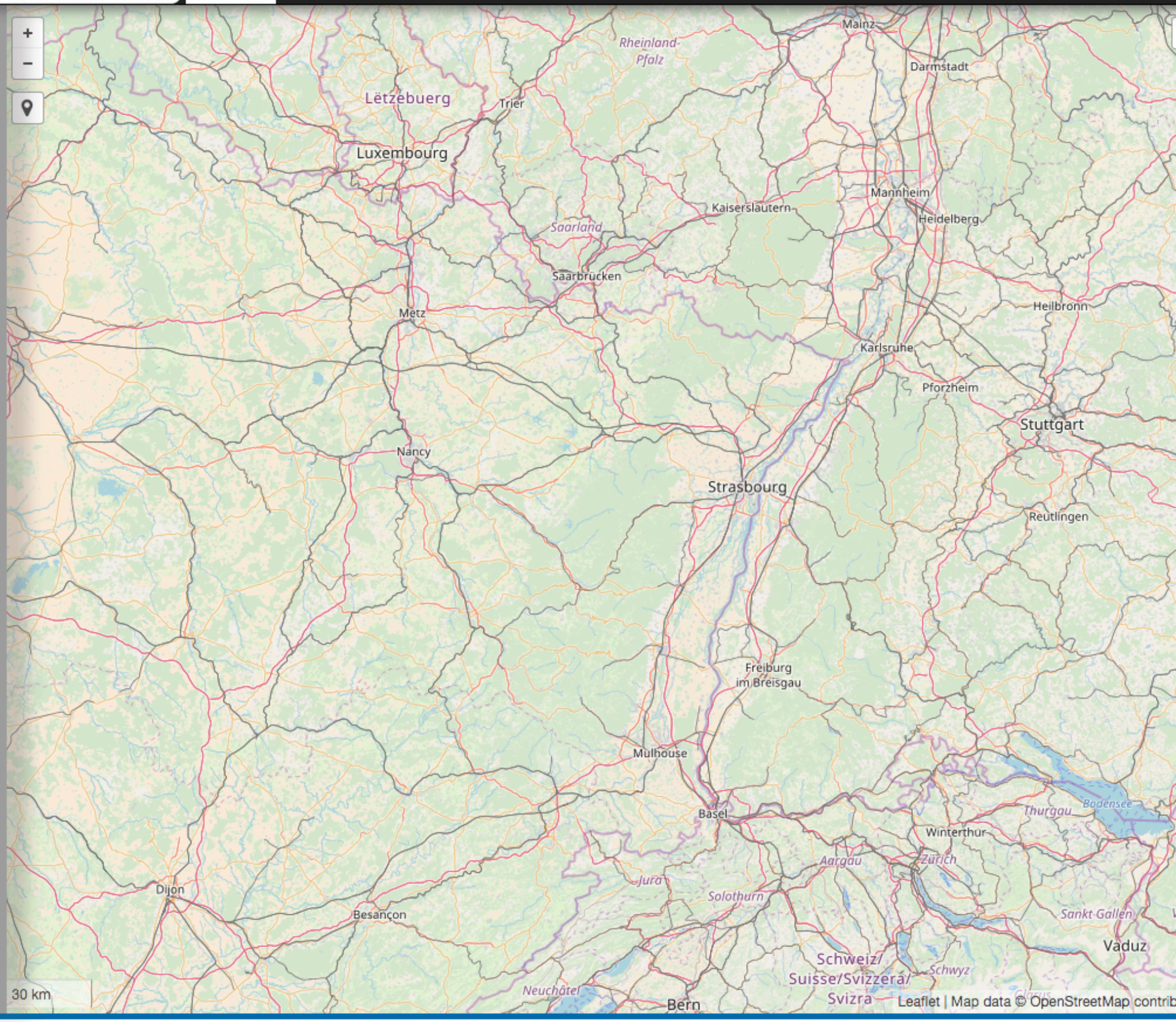
Choisir un document PDF

ou

le glisser ici

20180326-102629.145911_total_2.pdf

Valider



PARTENAIRES

Grand Est – Alsace Champagne-Ardenne Lorraine

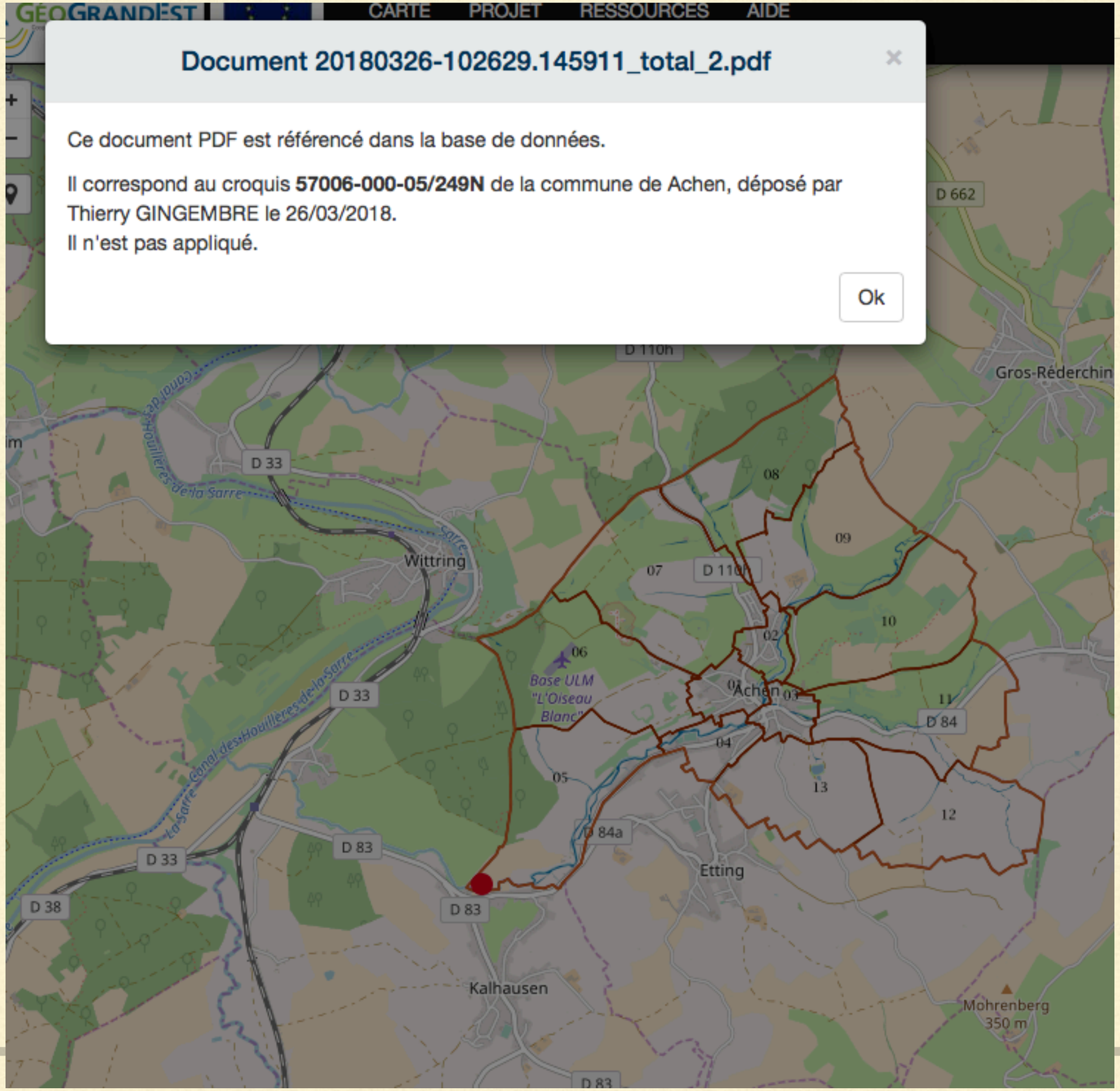
Conseil départemental Bas-Rhin

Conseil départemental Haut-Rhin

Département Moselle – Réussir ensemble !

Conseil régional Strasbourg

Finances publiques


GEOGRANDEST
CARTE
PROJET
RESSOURCES
AIDE
Document 20180326-102629.145911_total_2.pdf
Ce document PDF est référencé dans la base de données.
Il correspond au croquis 57006-000-05/249N de la commune de Achen, déposé par Thierry GINGEMBRE le 26/03/2018.
Il n'est pas appliqué.
Ok
D 662
D 110h
Gros-Réderchin
D 33
Wittring
Base ULM "L'Oiseau Blanc"
Achen
D 84
D 84a
D 83
D 38
Etting
Kalhausen
Mohrenberg
350 m
01
02
03
04
05
06
07
08
09
10
11
12
13

Document information

6caab7acf85c7c2a44a21a422241551f4ece410ef540bc874e105bd20aefe098

Registered in our servers since: **2018-04-10 08:23:23**

Want this document certified by a decentralized proof of existence?

We can embed the document's digest in the blockchain for you!

You'll need to pay **0.5 mBTC** to do so, to cover our costs. Please pay to the following address:

Please send **0.5 mBTC** or more to:
1ATDpJfd39Y18ozKNWuxi4BywGFQ6Y2AH9

Waiting for payment... the page will refresh automatically when a payment is received.

Blockchain et information géographique ?

En tant que registre, on peut y **stocker** la trace indélébile de transactions. Dans le domaine de l'information géographique, que vont pouvoir représenter ces transactions ?

- Des positions géographiques (flotte de véhicule, observations d'animaux, zonages protégés…) qui méritent d'être partagées et certifiées.
- Des mises à jours de positions géographiques, dans le contexte d'un référentiel ouvert (ala OpenStreetMap)
- Des fichiers (OpenData, photos, PDF…) référencés par leur empreinte informatique (hash)
- Des données de capteurs (bruit, pollution, fréquentation…)

A chacun de ces exemples l'utilisation de la blockchain apporte distributivité et inaltérabilité.

Un projet, une idée ?
N’hésitez pas à venir en discuter
sur notre stand !