



UNIVERSITÀ DI PISA

Corso di Laurea in Informatica

2021/2022

Tutore Accademico:  
Laura Ricci

Tutore Aziendale:  
Simone Gianfranceschi

**Algorand**



# **Integrazione della tecnologia Blockchain in un sistema di certificazione della posizione**

Candidato:  
Lorenzo Angeli

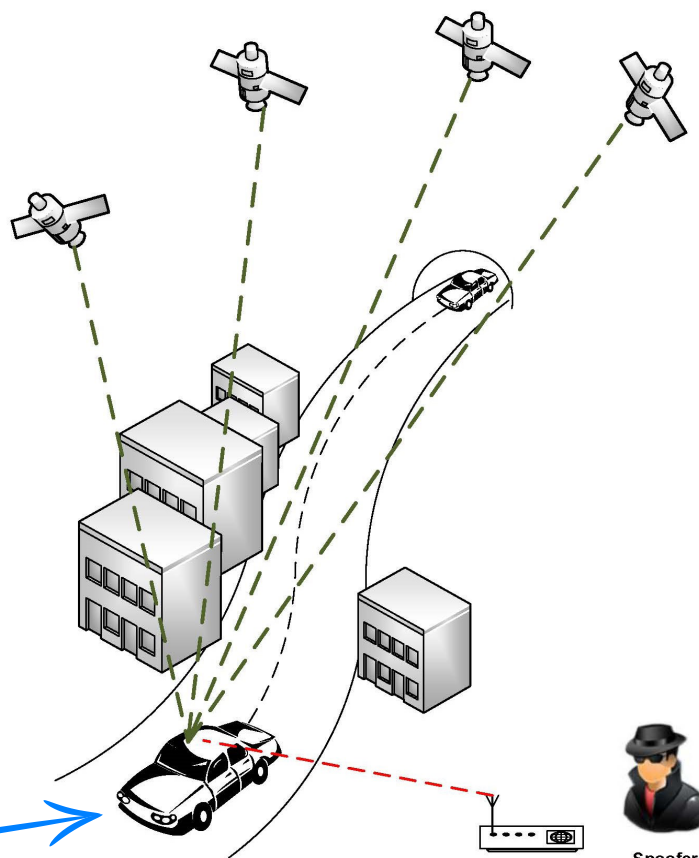
# Scopo del progetto

Intecs S.p.A. ha sviluppato un sistema di certificazione della posizione. Lo scopo del tirocinio è stato quello di integrare la tecnologia Blockchain di Algorand all'interno del sistema di certificazione della posizione già esistente per rendere il sistema più sicuro.



# Perchè certificare la posizione?

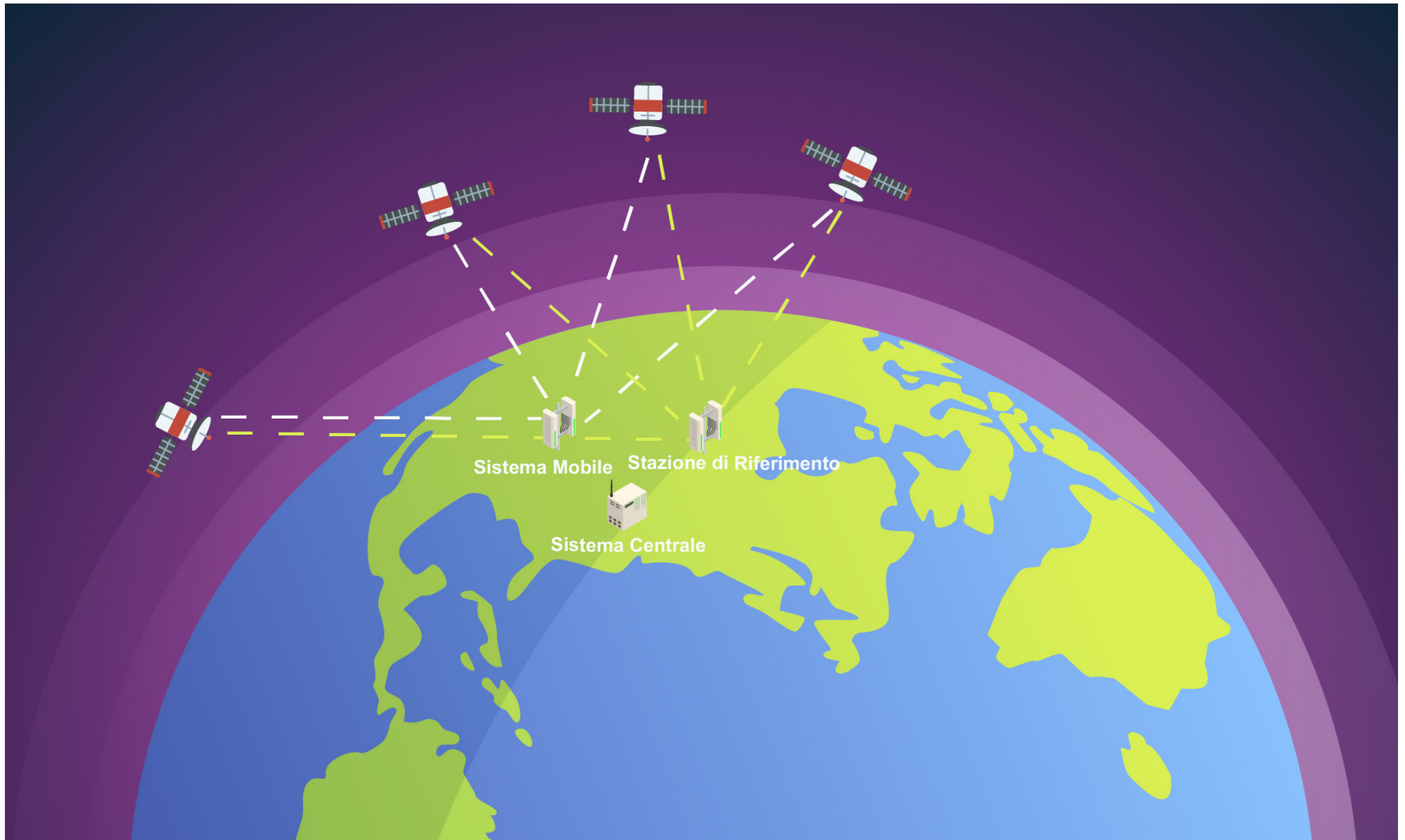
I sistemi che utilizzano strumenti di tipo GNSS (Global Navigation Satellite System) per rilevare la propria posizione, possono essere violati con una tecnica che prende il nome di **spoofing**.



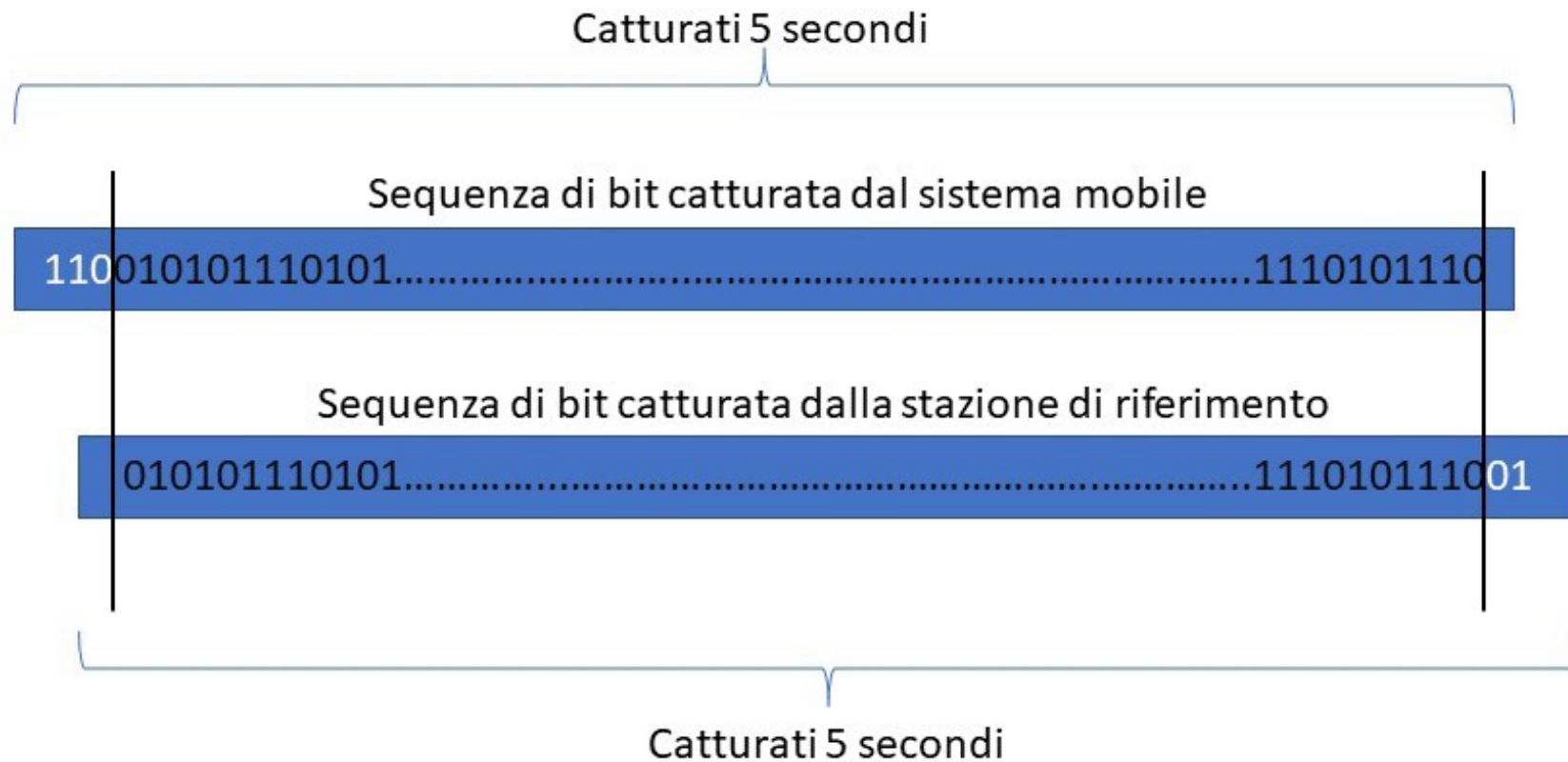
Ricevitore installato sull'auto

Spoofers

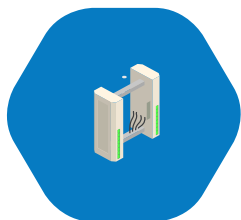
# Sistema di certificazione della posizione



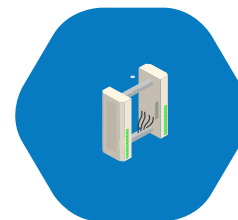
# Allineamento degli snapshot



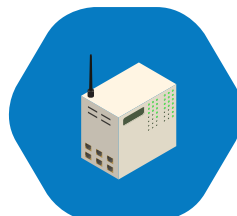
# Componenti del progetto sviluppato



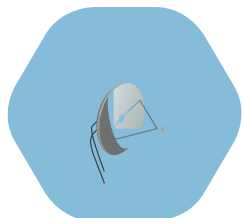
Sistema Mobile



Stazione di Riferimento



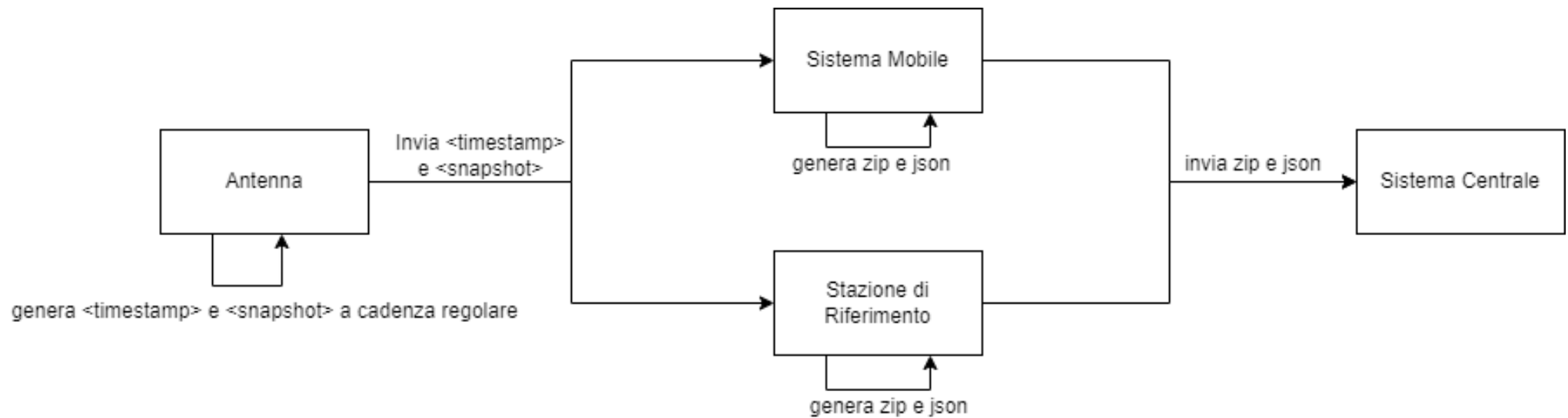
Sistema Centrale



Antenna

L'Antenna simula la ricezione degli snapshot dai satelliti.

# Cos'è la componente Antenna?



# Introduciamo la Blockchain

BLOCK GENESIS



BLOCK 1



BLOCK 2



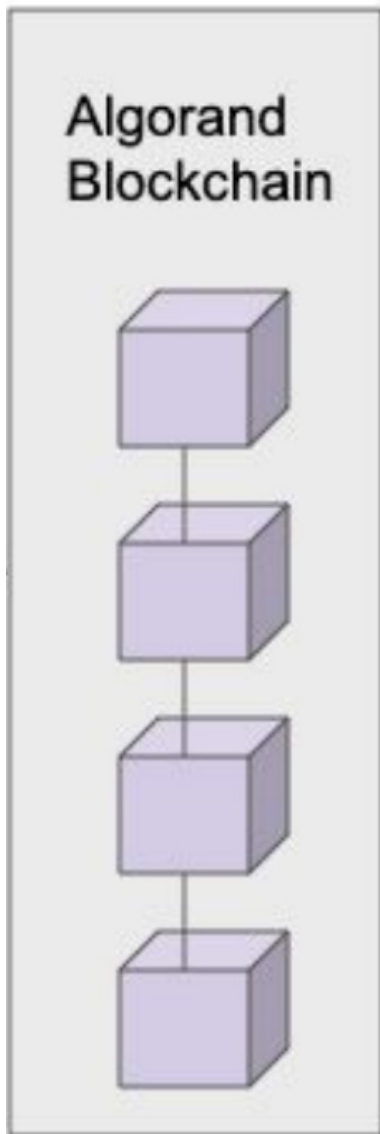
|           |                   |
|-----------|-------------------|
| Timestamp | 1495941516.704196 |
| Data      | 'A found 1 euro'  |
| Prev_hash |                   |
| Hash      | 'a75a9227f...'    |

|           |                       |
|-----------|-----------------------|
| Timestamp | 1495941516.704201     |
| Data      | 'A gives 1 euro to B' |
| Prev_hash | 'a75a9227f...'        |
| Hash      | 'ca911be27...'        |

|           |                       |
|-----------|-----------------------|
| Timestamp | 1495941516.704277     |
| Data      | 'B gives 1 euro to C' |
| Prev_hash | 'ca911be27...'        |
| Hash      | 'be356hj09...'        |

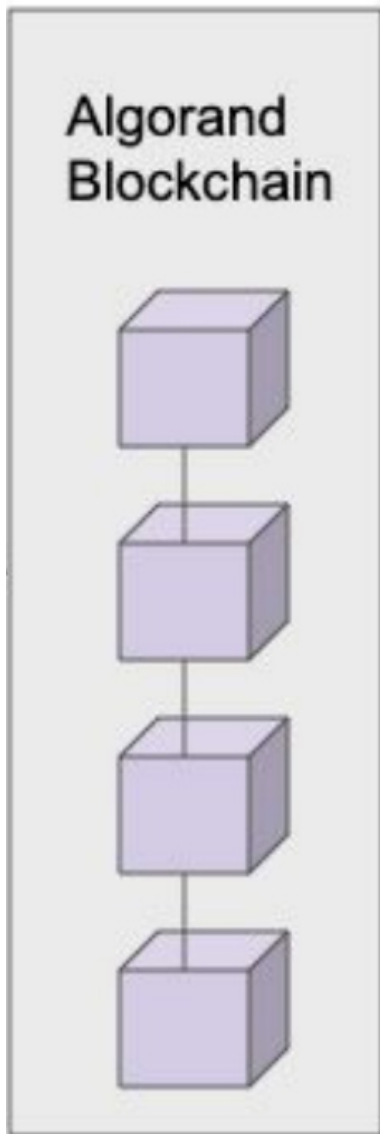


# Integriamo la Blockchain



Abbiamo introdotto uno **Smart Contract** al fine di rendere:

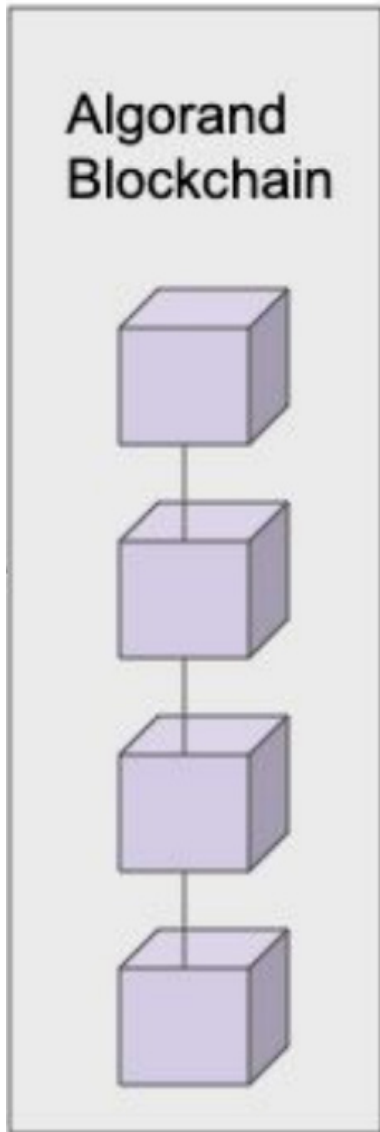
# Integriamo la Blockchain



Abbiamo introdotto uno **Smart Contract** al fine di rendere:

- alcune operazioni sicure

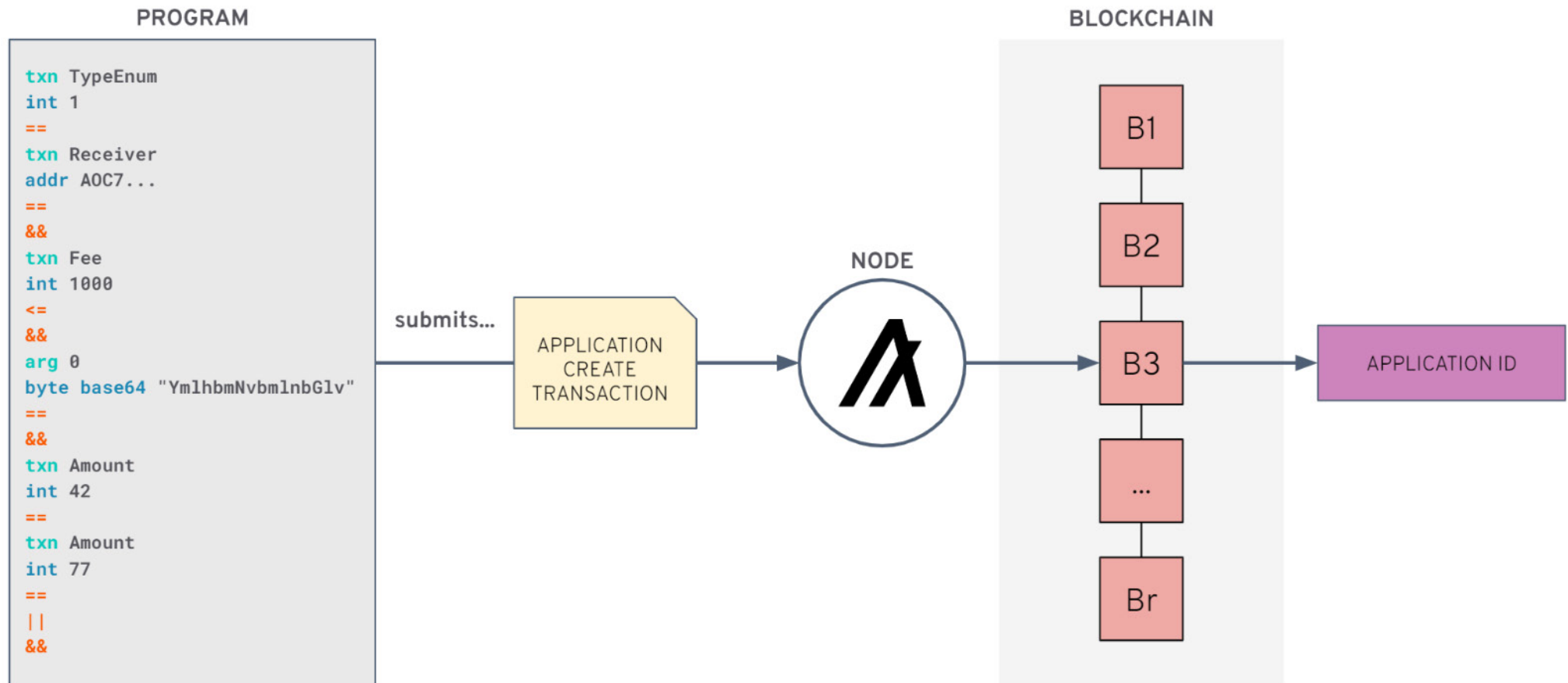
# Integriamo la Blockchain



Abbiamo introdotto uno **Smart Contract** al fine di rendere:

- alcune operazioni sicure
- i dati di posizione immutabili

# Distribuzione dello Smart Contract sulla Blockchain



# Sviluppare Smart Contract è più semplice con PyTEAL

La **libreria PyTEAL** è disponibile solamente per il linguaggio di programmazione Python.

## TEAL Source Code

```
txn Receiver
addr AOC7...
==
txn Amount
int 1000
<=
&&
```

COMPILE...

AVM bytecode

## PyTEAL Source Code

```
And(
    Txn.Receiver == Addr(AOC7...),
    Txn.Amount <= Int(1000),
)
```

COMPILE...

## TEAL Source Code

```
txn Receiver
addr AOC7...
==
txn Amount
int 1000
<=
&&
```

COMPILE...

AVM bytecode

# Archiviazione dei dati sullo Smart Contract

Si possono anche memorizzare valori sullo smart contract, esistono due **tipi di dato base** da poter salvare:

# Archiviazione dei dati sullo Smart Contract

Si possono anche memorizzare valori sullo smart contract, esistono due **tipi di dato base** da poter salvare:

- uint64

# Archiviazione dei dati sullo Smart Contract

Si possono anche memorizzare valori sullo smart contract, esistono due **tipi di dato base** da poter salvare:

- uint64
- stringhe di byte



# Archiviazione dei dati sullo Smart Contract

Si possono anche memorizzare valori sullo smart contract, esistono due **tipi di dato base** da poter salvare:

- uint64
- stringhe di byte

L'**archiviazione** delle variabili può essere:

# Archiviazione dei dati sullo Smart Contract

Si possono anche memorizzare valori sullo smart contract, esistono due **tipi di dato base** da poter salvare:

- uint64
- stringhe di byte

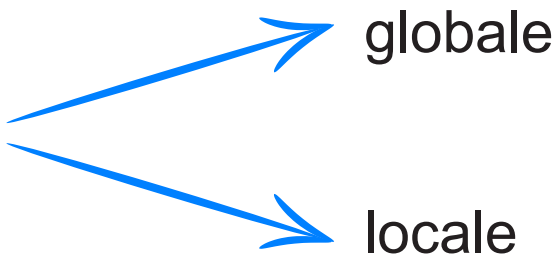
L'**archiviazione** delle variabili può essere:  globale

# Archiviazione dei dati sullo Smart Contract

Si possono anche memorizzare valori sullo smart contract, esistono due **tipi di dato base** da poter salvare:

- uint64
- stringhe di byte

L'**archiviazione** delle variabili può essere:



- globale
- locale

# Specifiche dello Smart Contract

## Smart Contract

### VARIABILI GLOBALI

Address Sistema Centrale

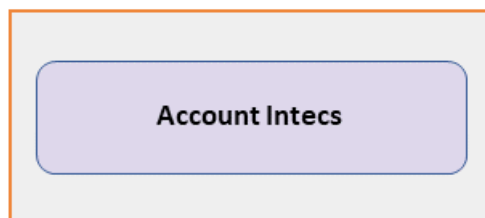
### VARIABILI LOCALI

hash\_snapshot

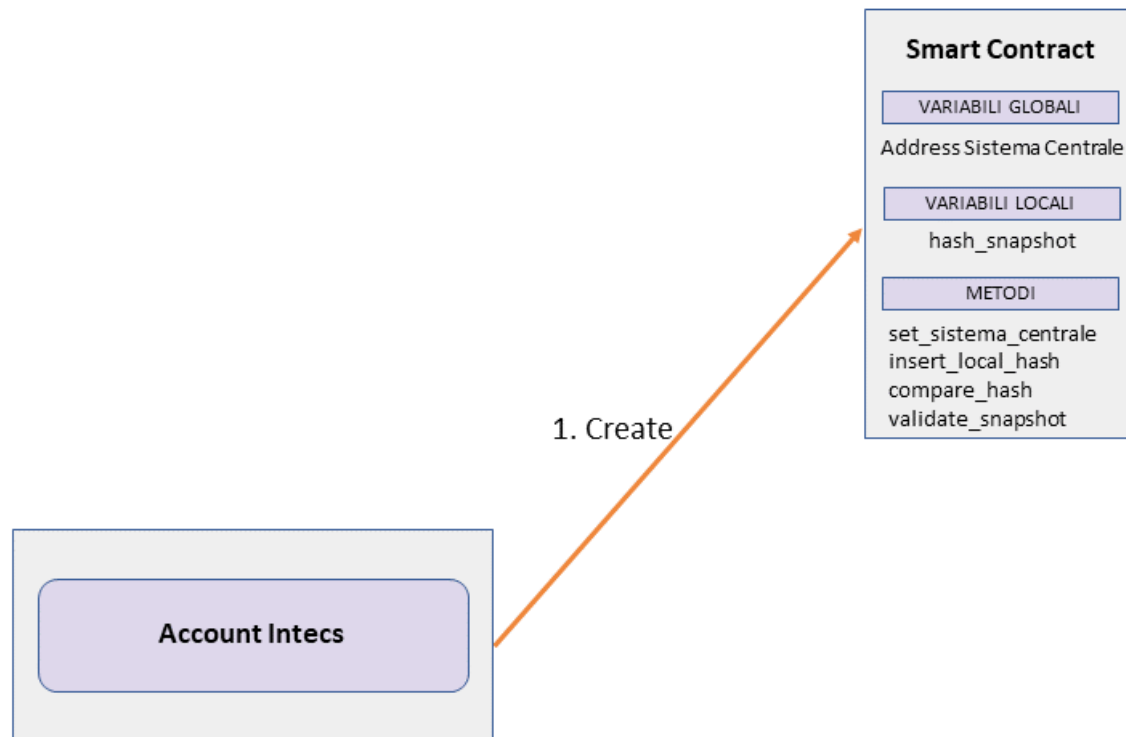
### METODI

- set\_sistema\_centrale: Intecs
- insert\_local\_hash: Sistema Mobile
- compare\_hash: Sistema Centrale
- validate\_snapshot: Sistema Centrale

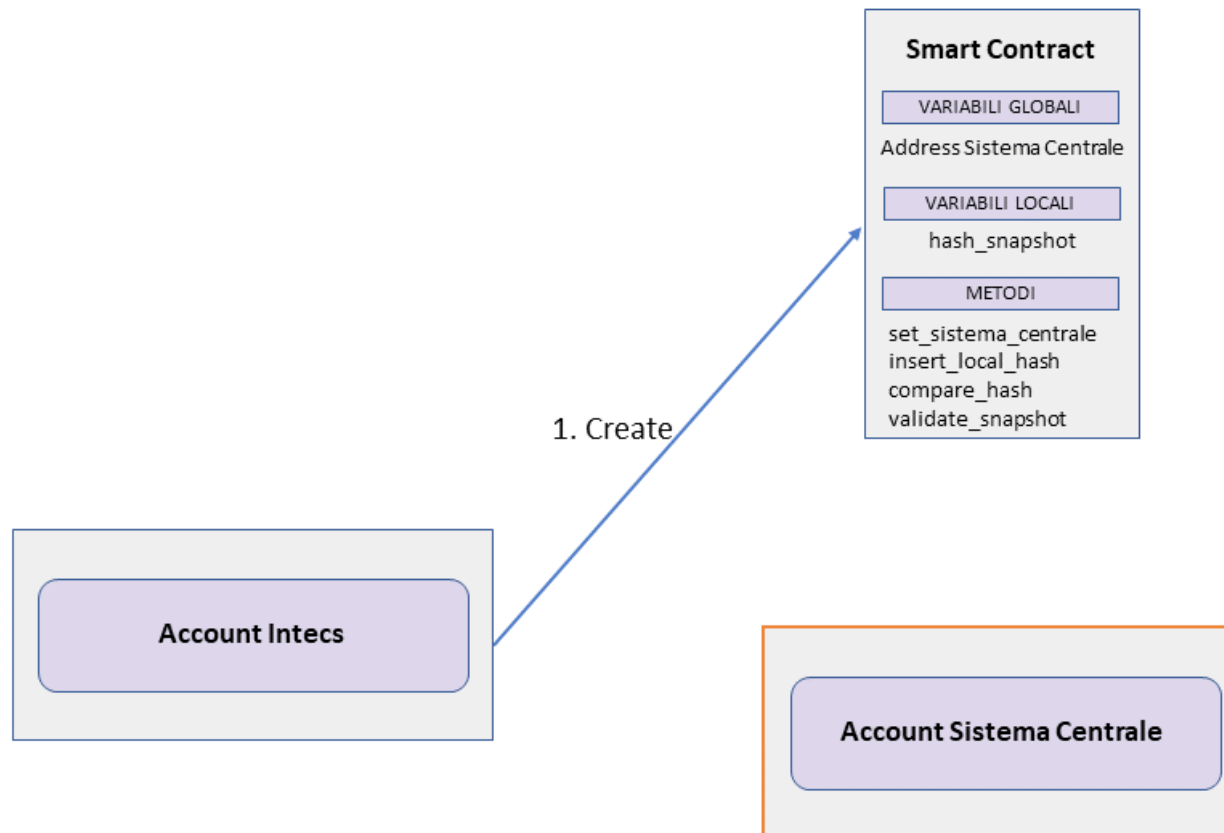
# Creazione dello Smart Contract



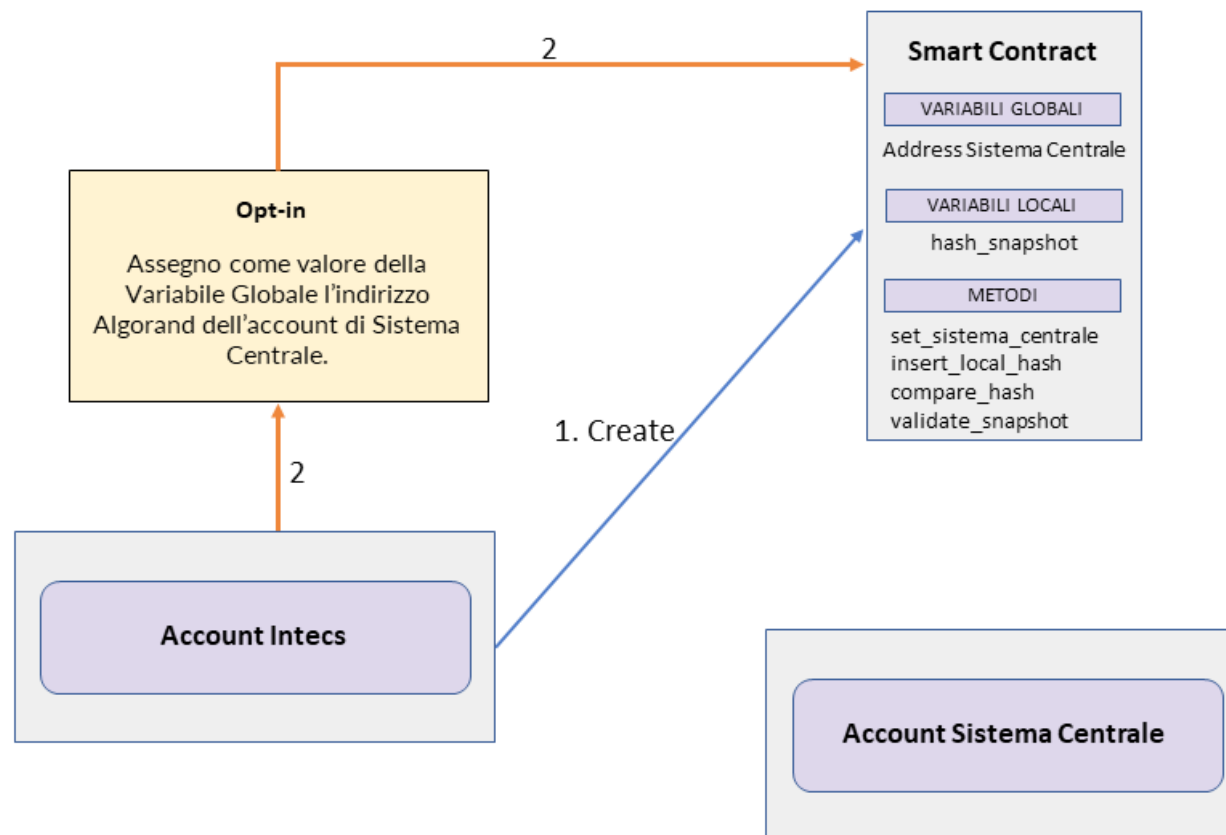
# Creazione dello Smart Contract



# Creazione dello Smart Contract

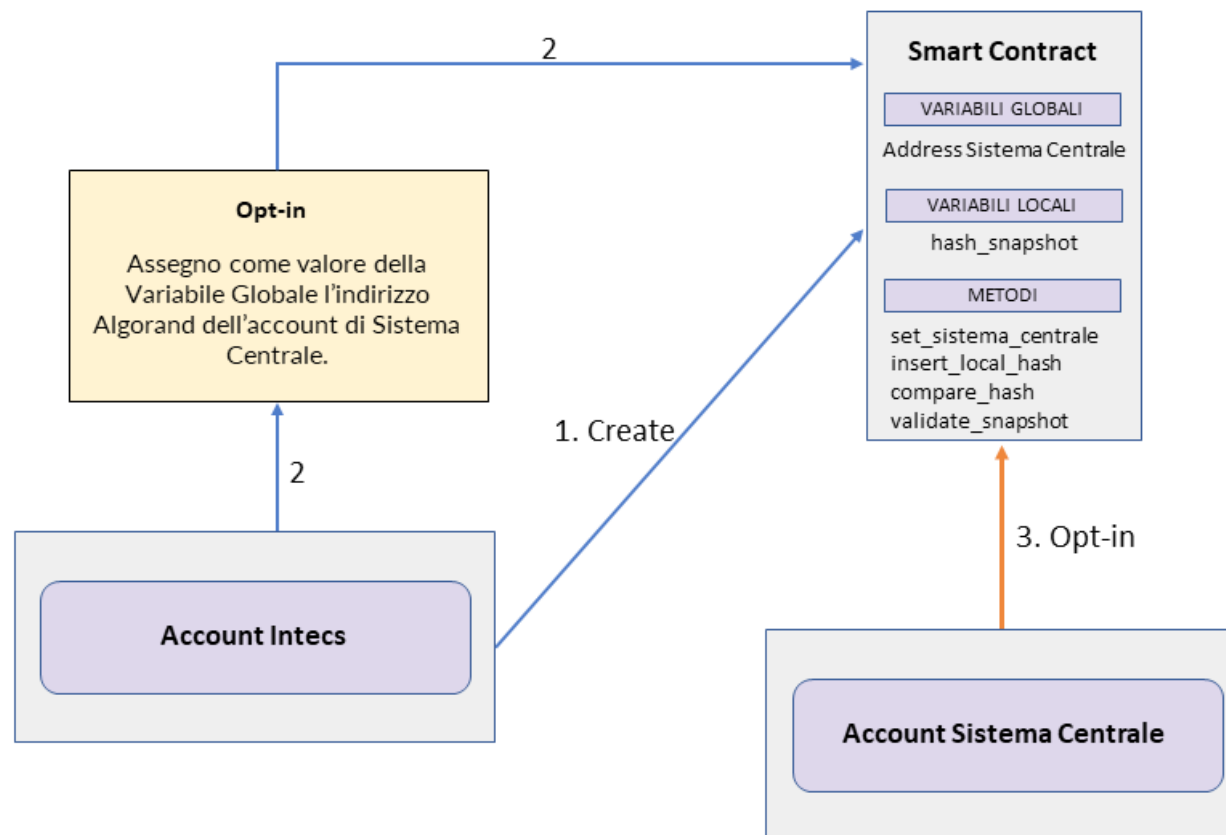


# Creazione dello Smart Contract

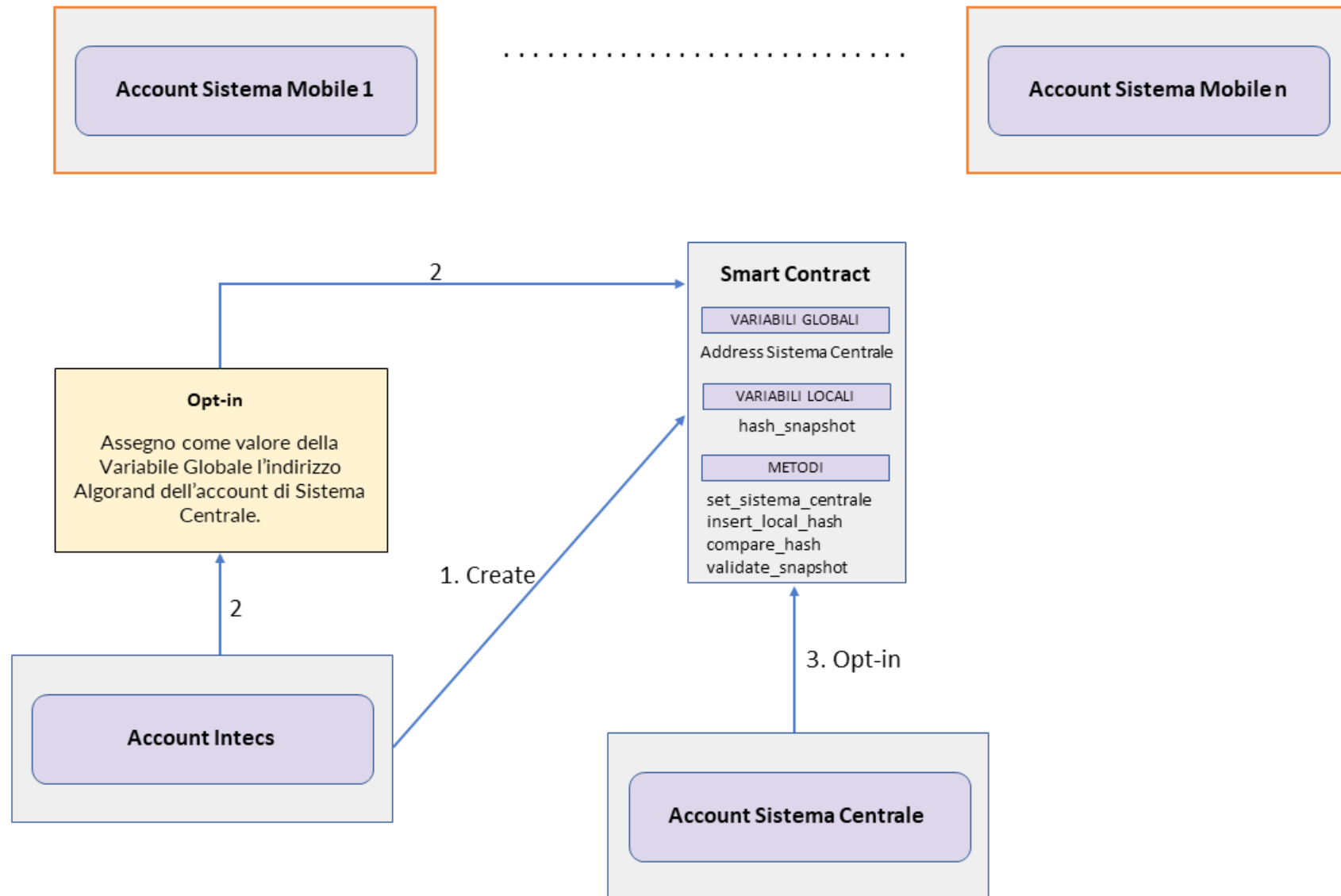




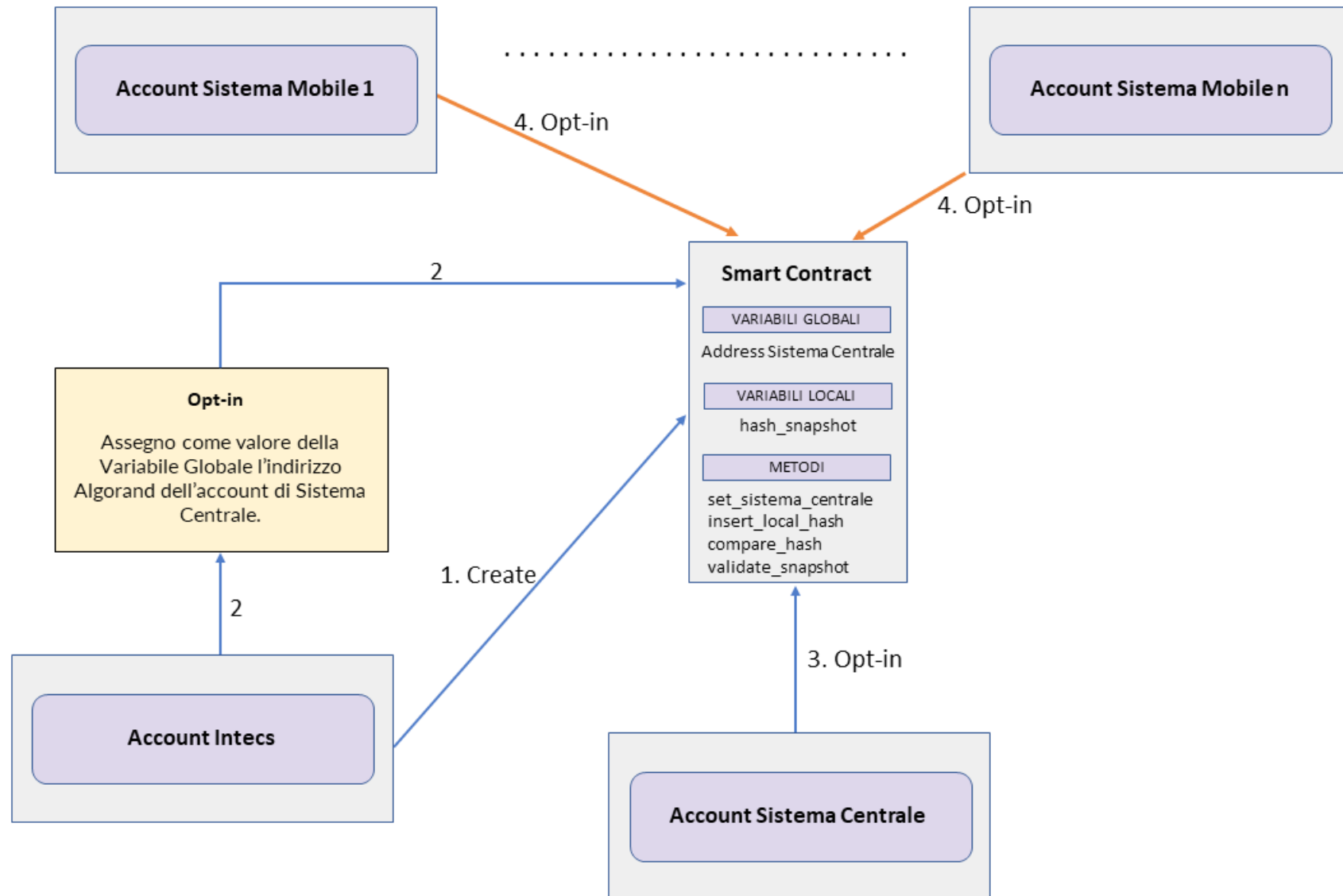
# Creazione dello Smart Contract



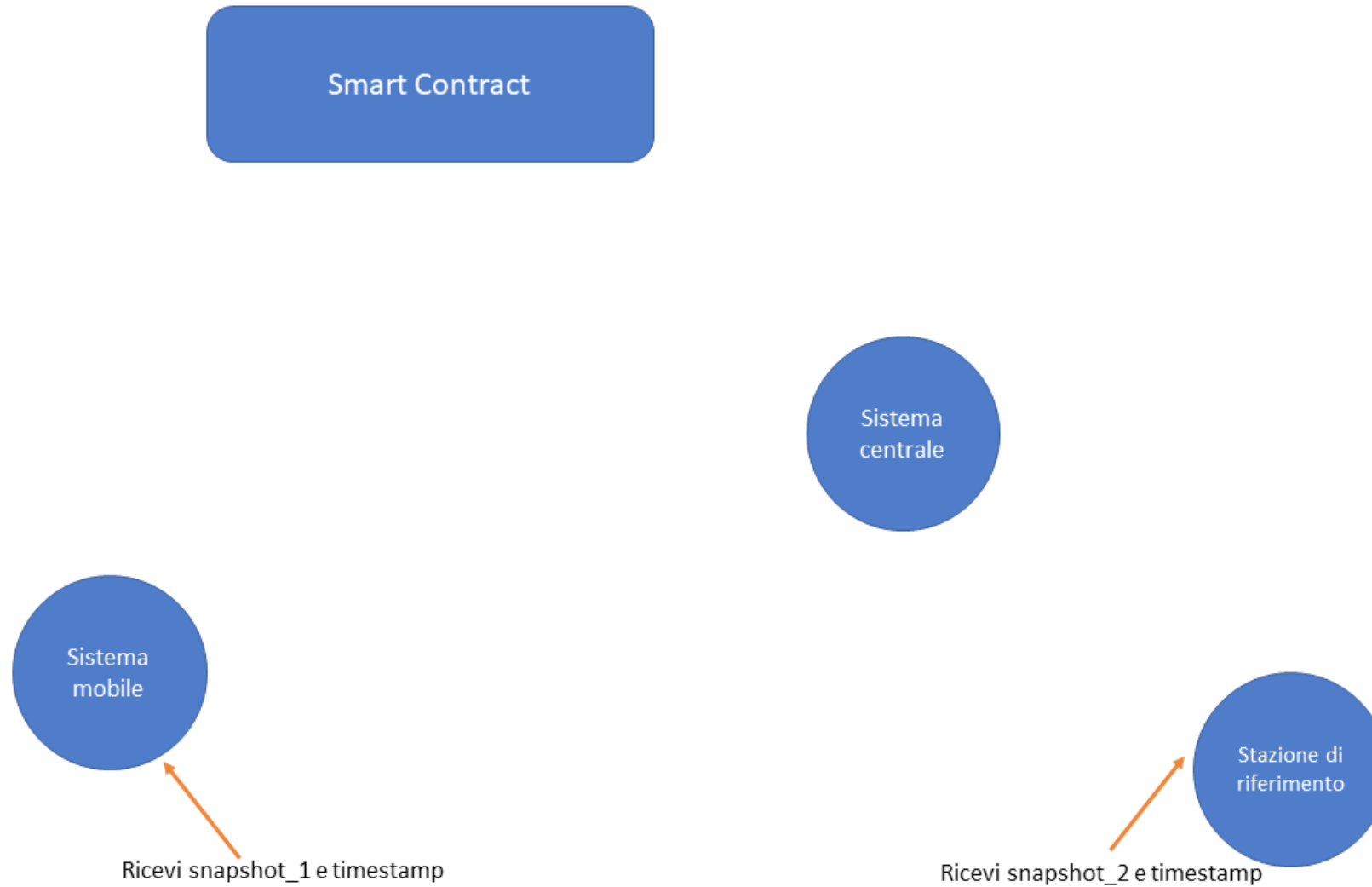
# Creazione dello Smart Contract



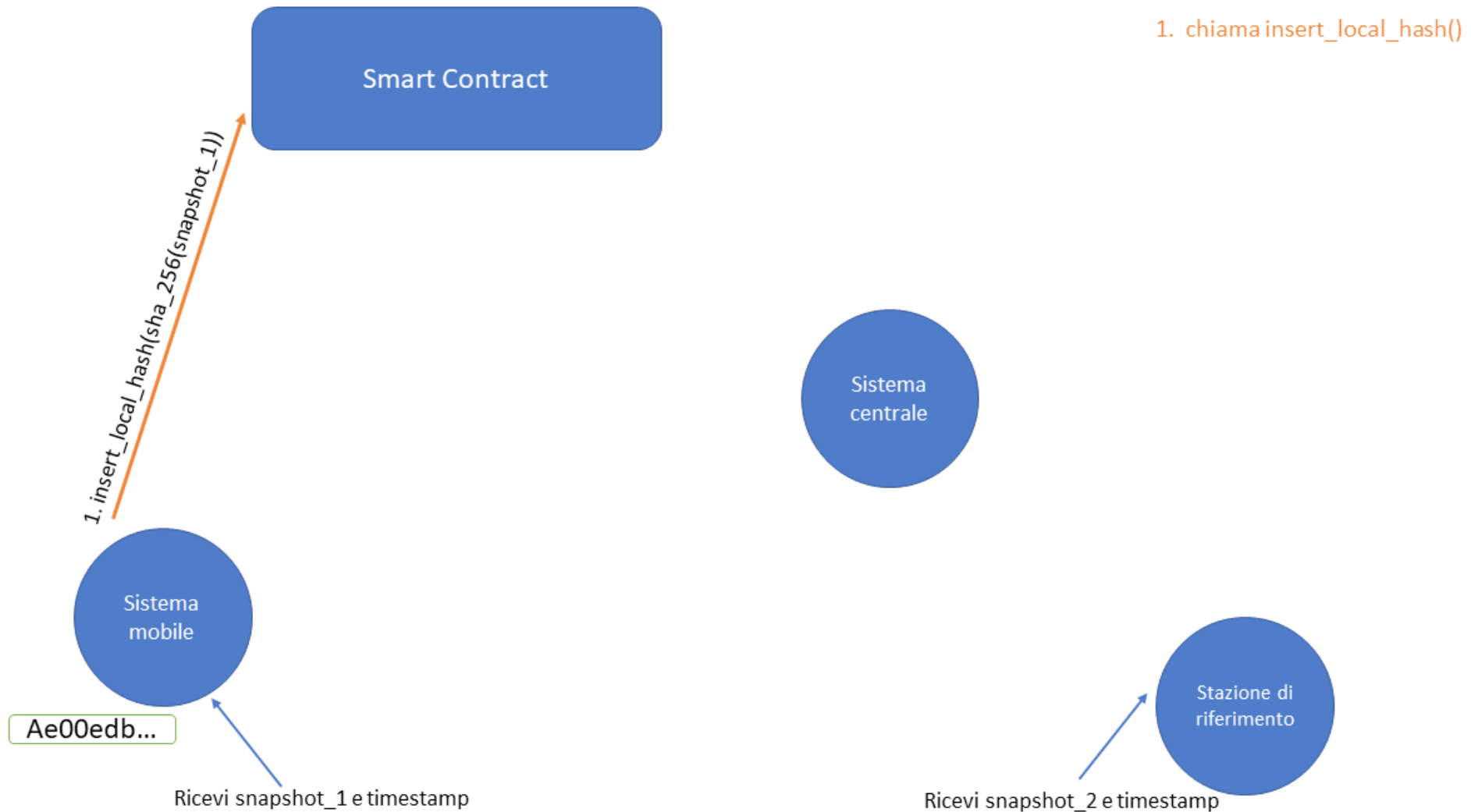
# Creazione dello Smart Contract



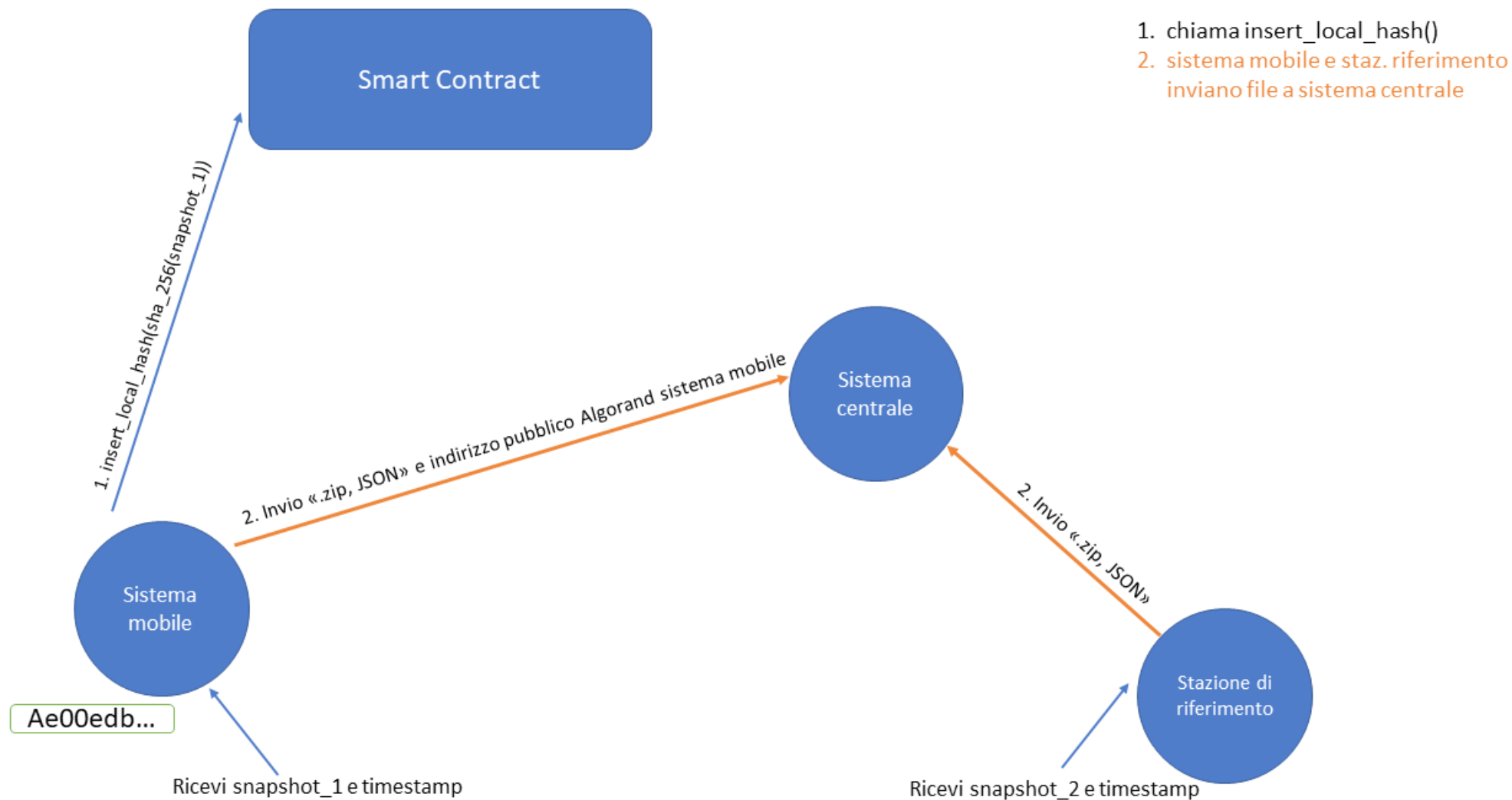
# Chiamate ai metodi dello Smart Contract



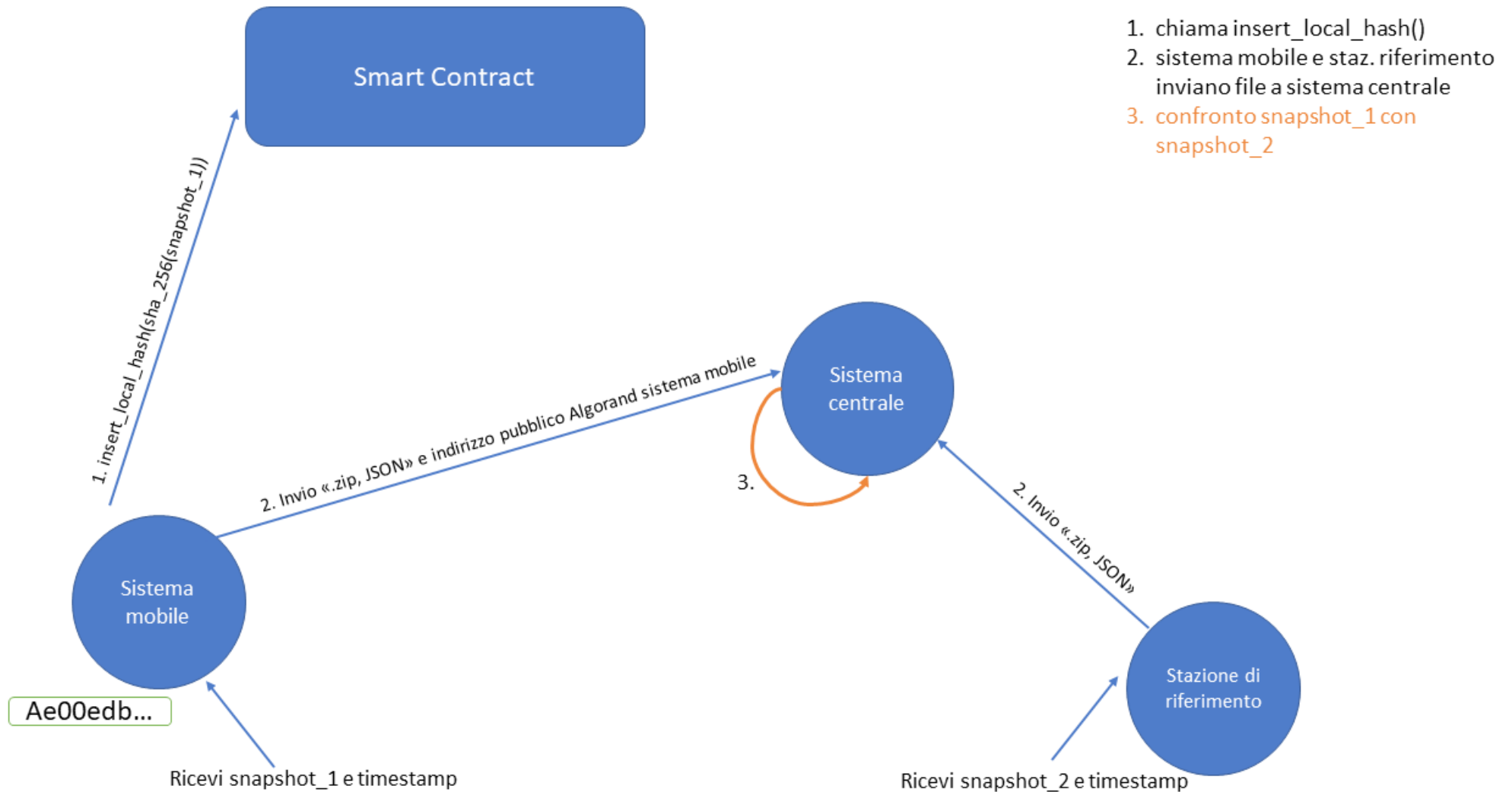
# Chiamate ai metodi dello Smart Contract



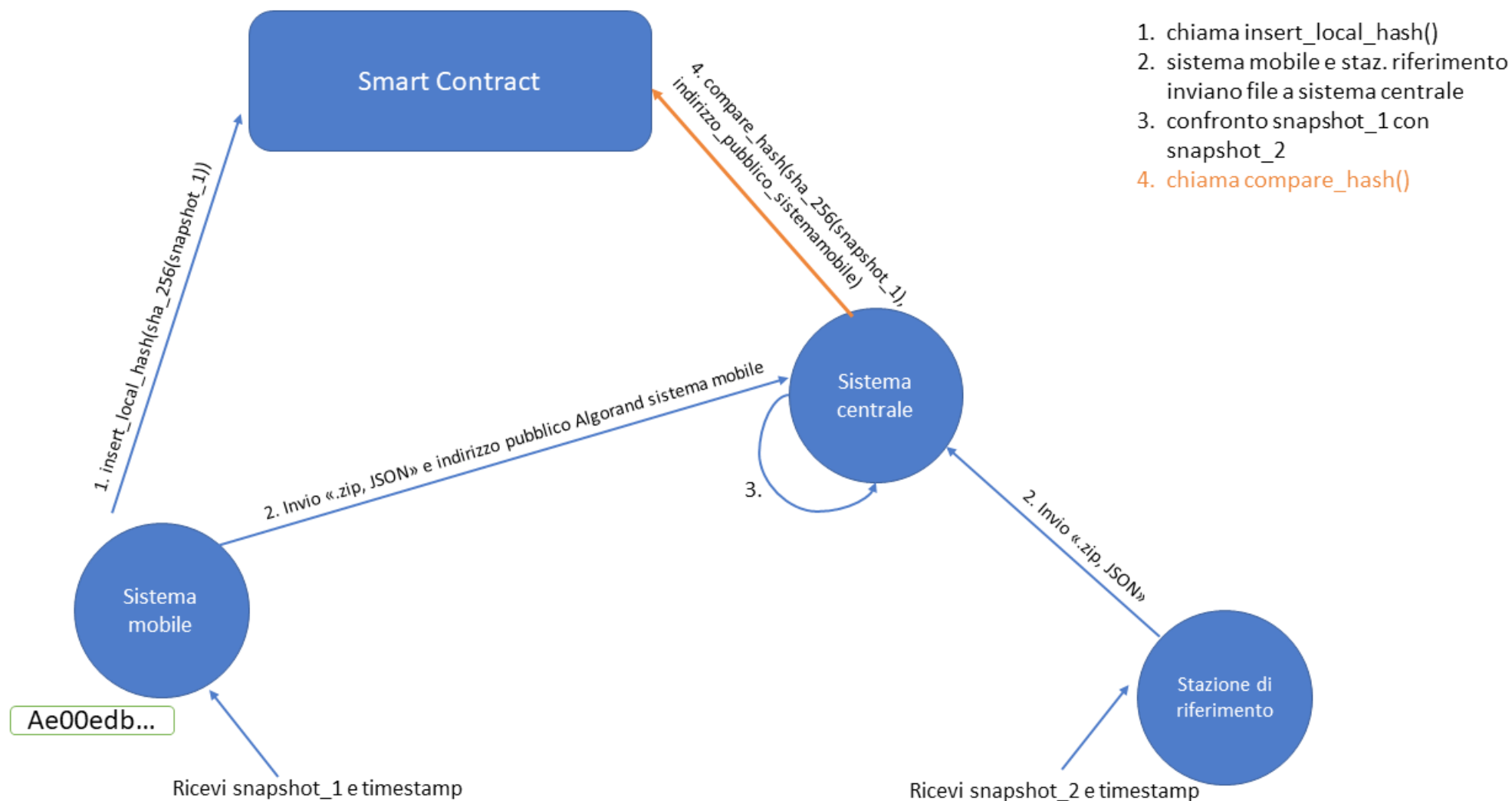
# Chiamate ai metodi dello Smart Contract



# Chiamate ai metodi dello Smart Contract

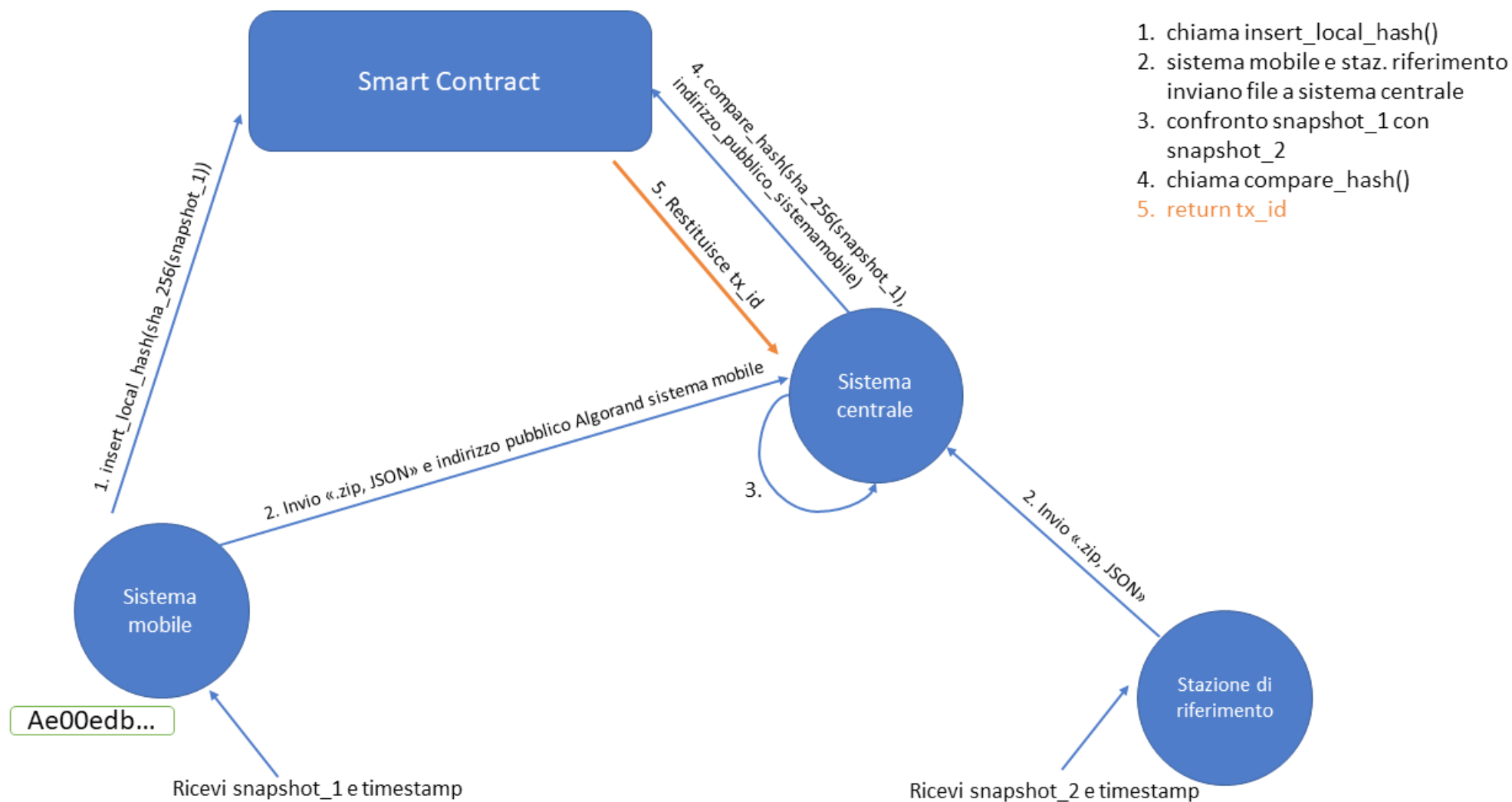


# Chiamate ai metodi dello Smart Contract

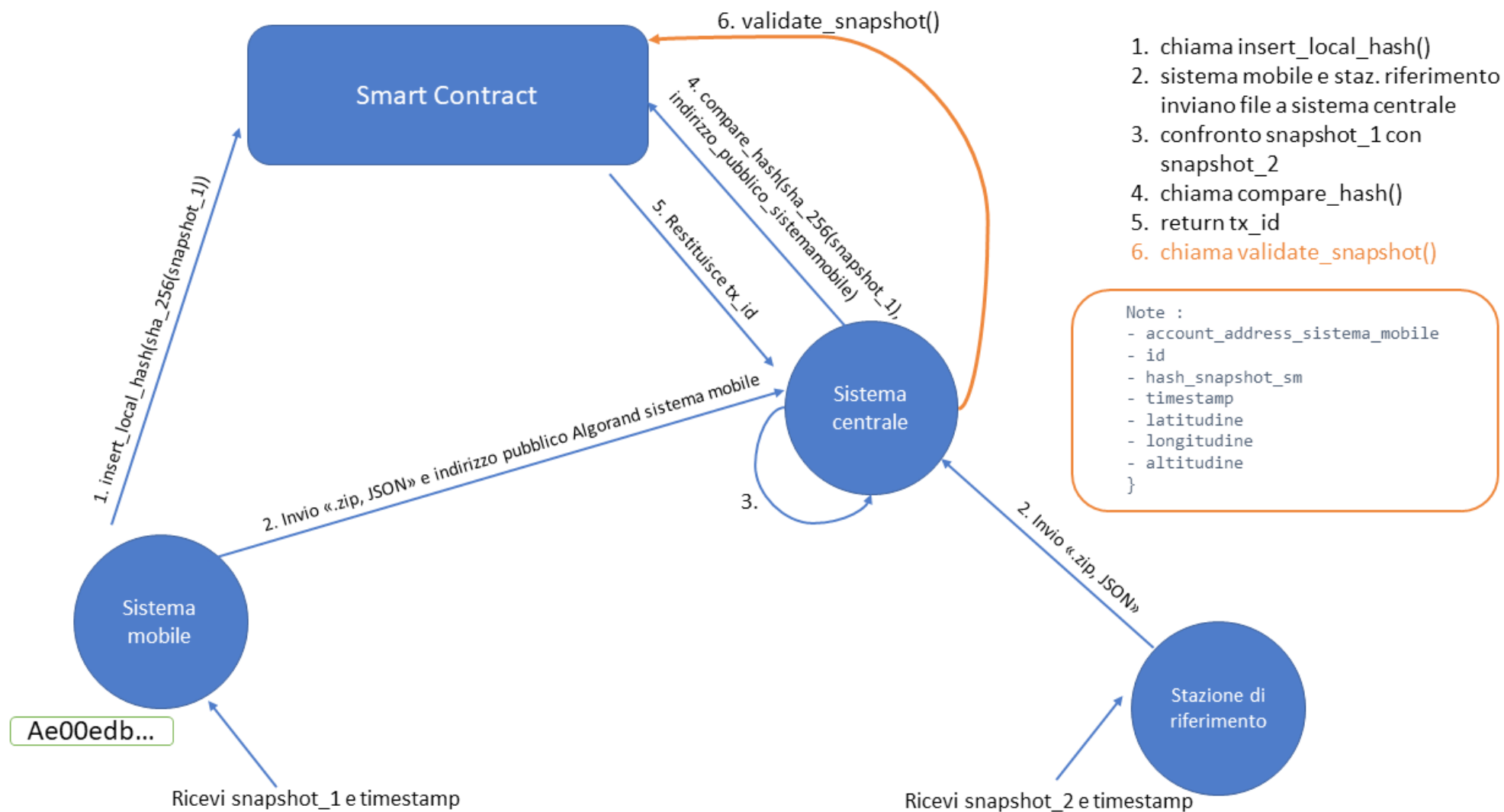




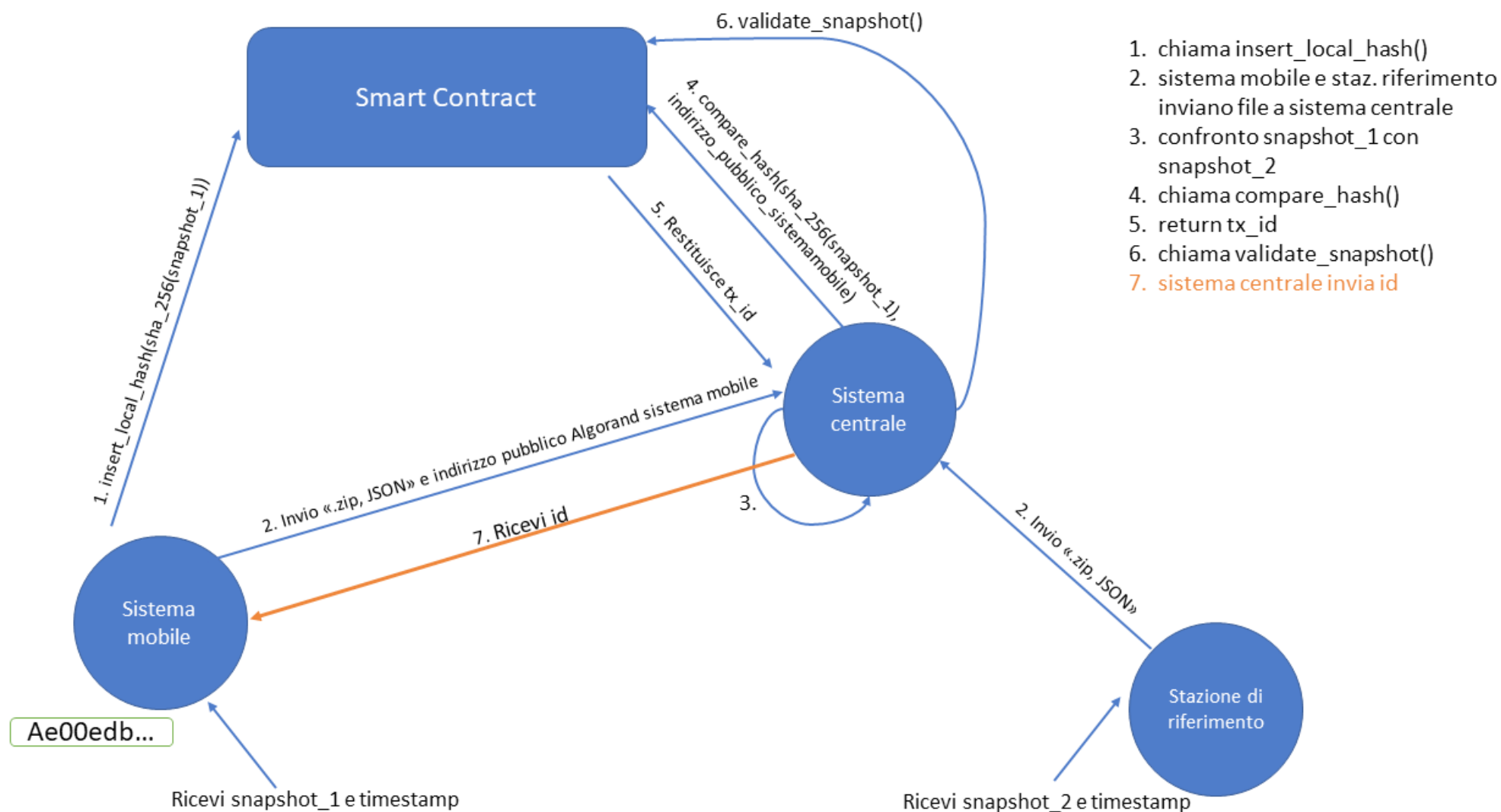
# Chiamate ai metodi dello Smart Contract



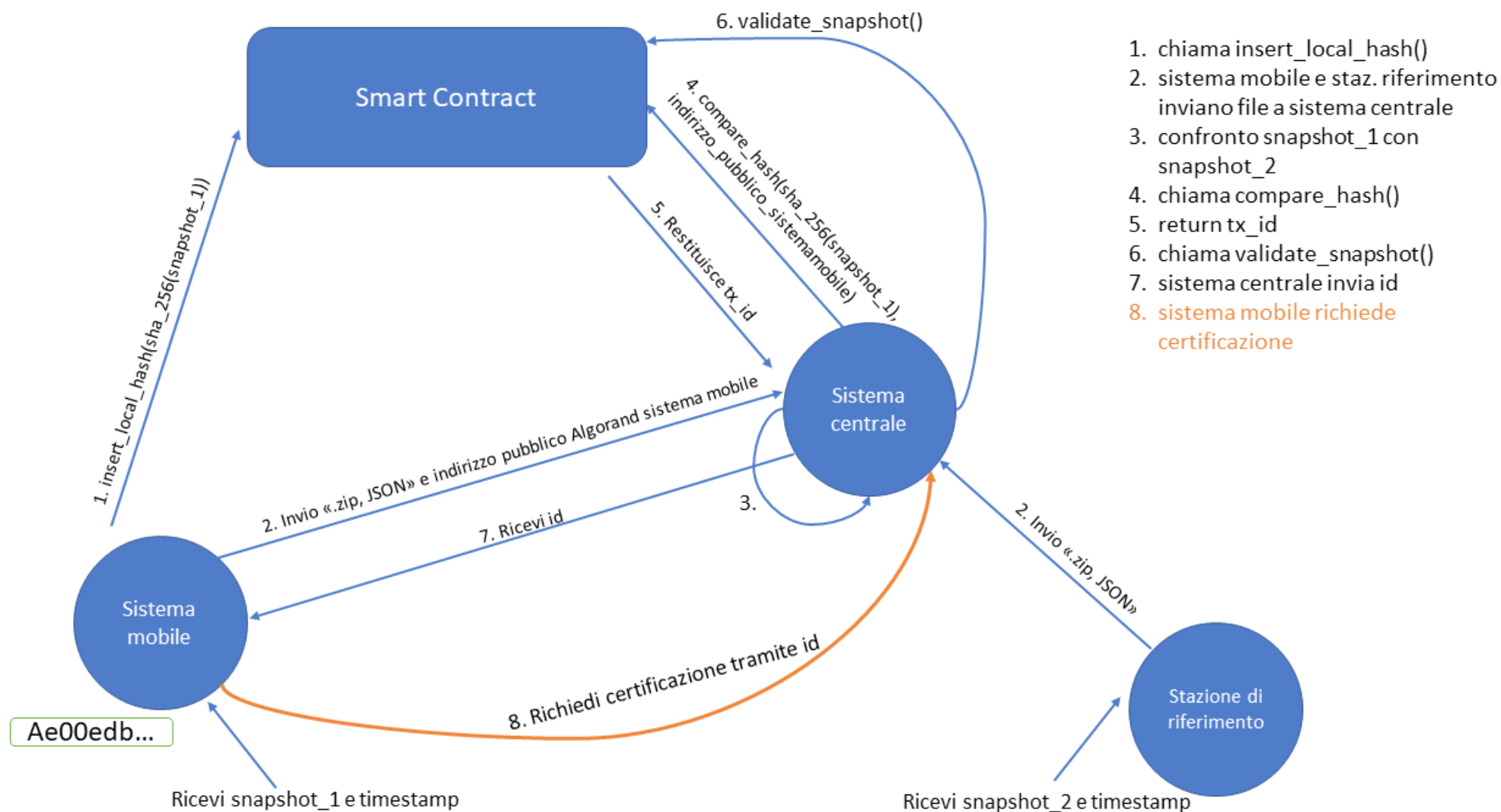
# Chiamate ai metodi dello Smart Contract



# Chiamate ai metodi dello Smart Contract



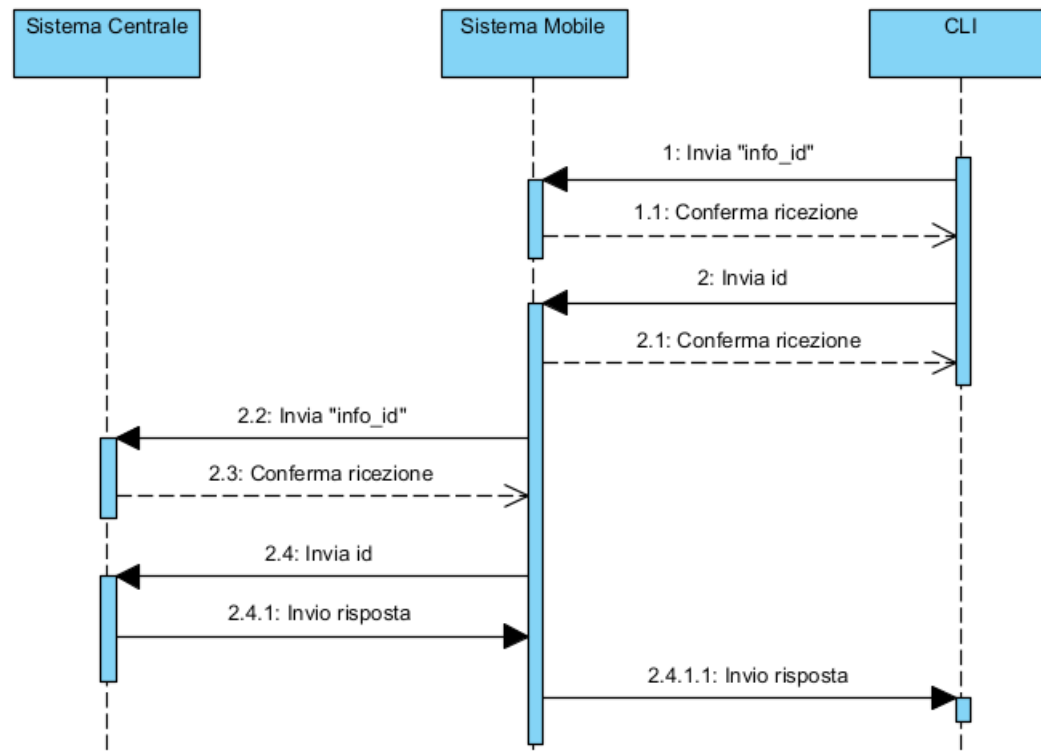
# Chiamate ai metodi dello Smart Contract



# Cli v1

Sono state implementate **due versioni** dell'interfaccia a linea di comando cli, utilizzato per richiedere informazioni sugli snapshot ricevuti attraverso gli id.

Nella prima versione, si sfrutta il collegamento con il sistema mobile per poter richiedere le informazioni sugli snapshot.



## Cli v2

L'indexer **fornisce** un'interfaccia API REST di chiamate API per supportare la ricerca dei dati sulla Blockchain di Algorand.

In questa versione v2 invece, il terminale si interfaccia direttamente con la blockchain comunicando attraverso l'indexer.

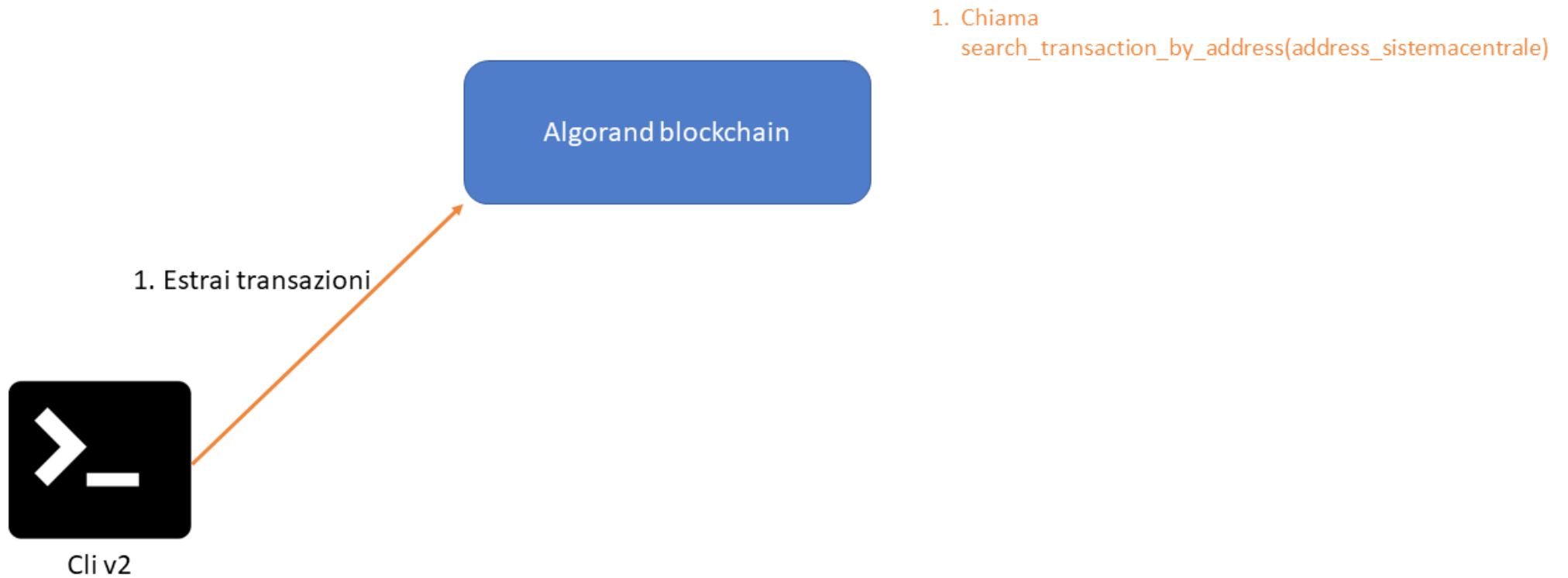
# Cli v2

Algorand blockchain



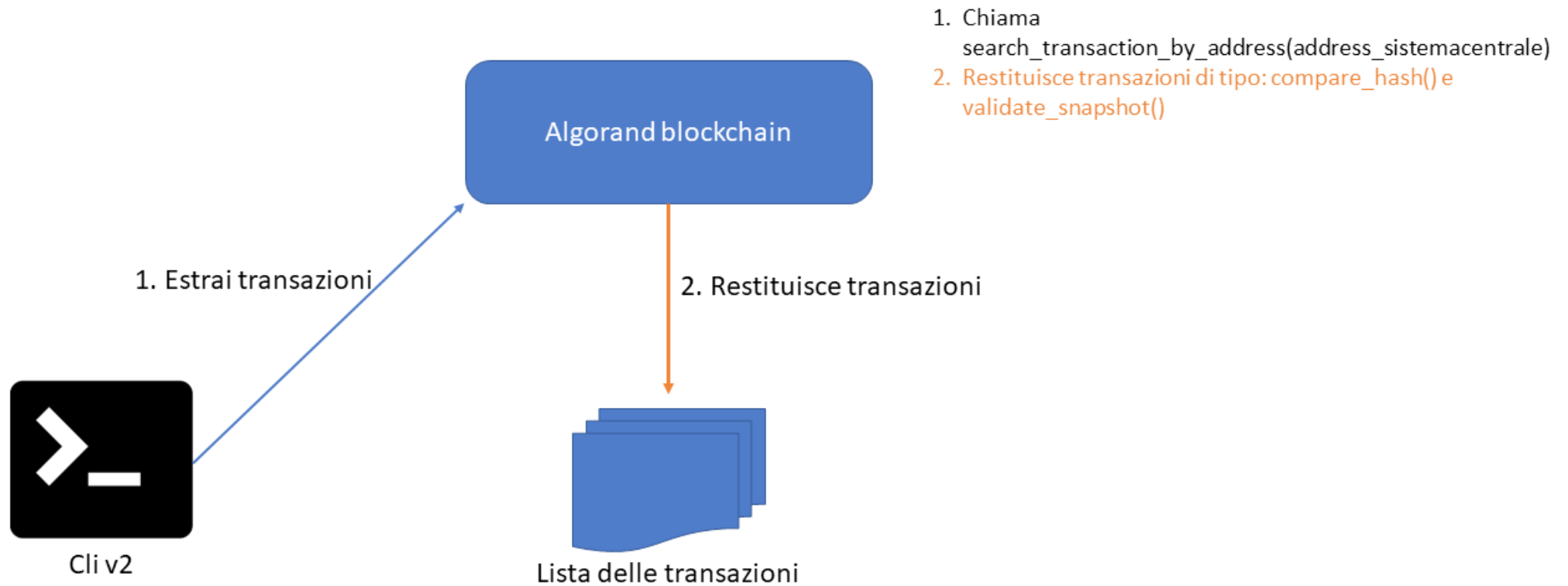
Cli v2

# Cli v2

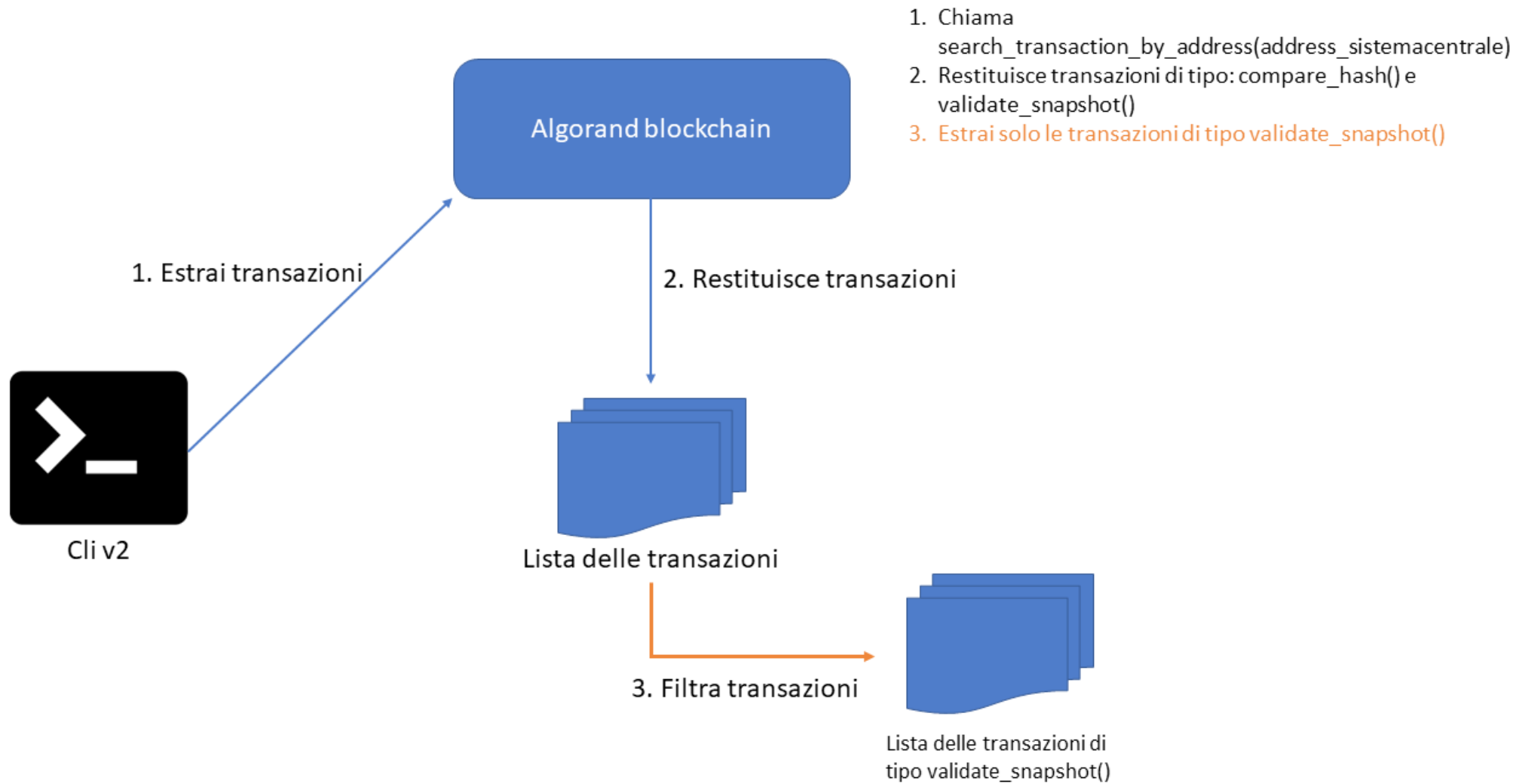




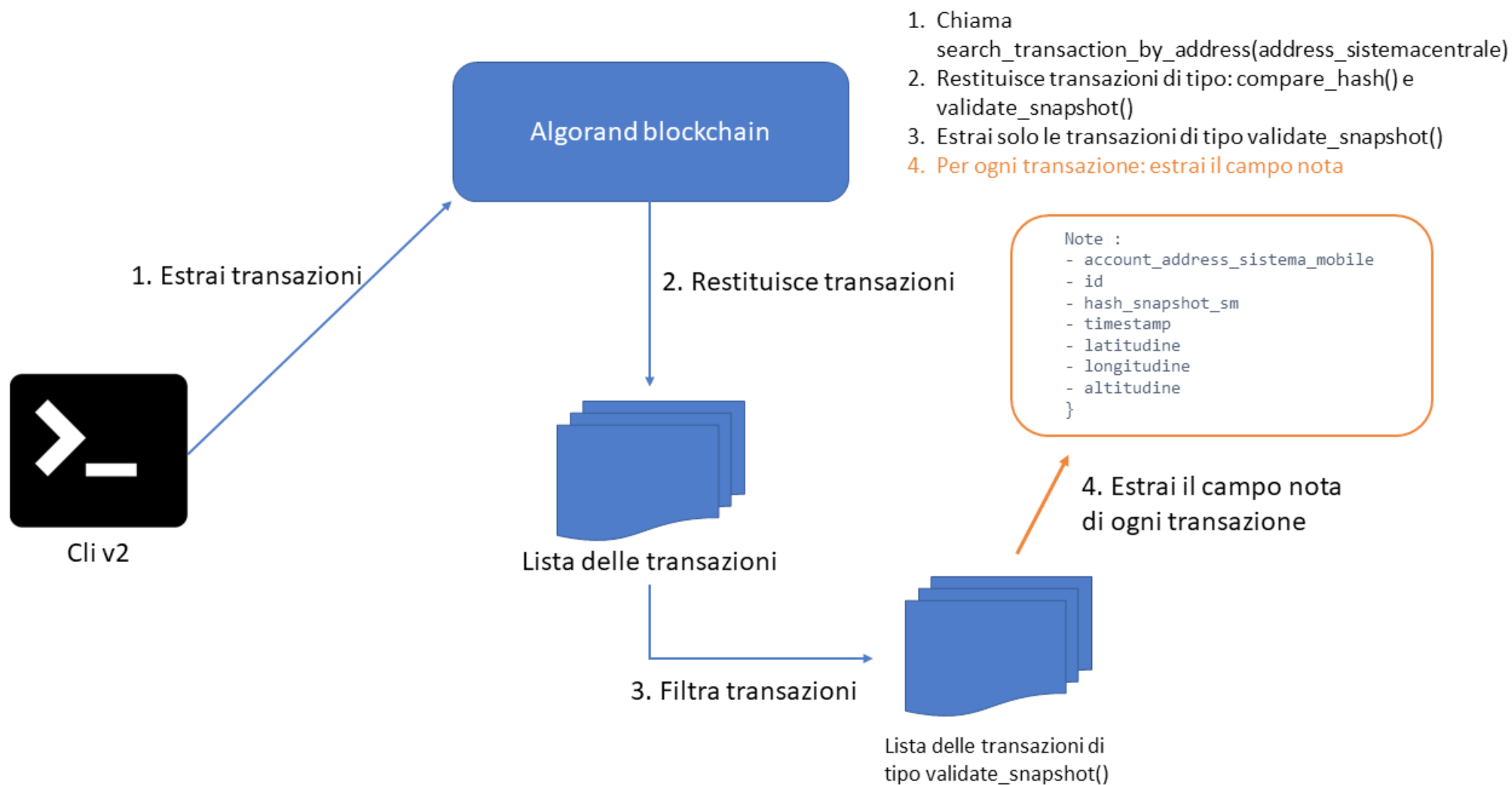
# Cli v2



# Cli v2



# Cli v2

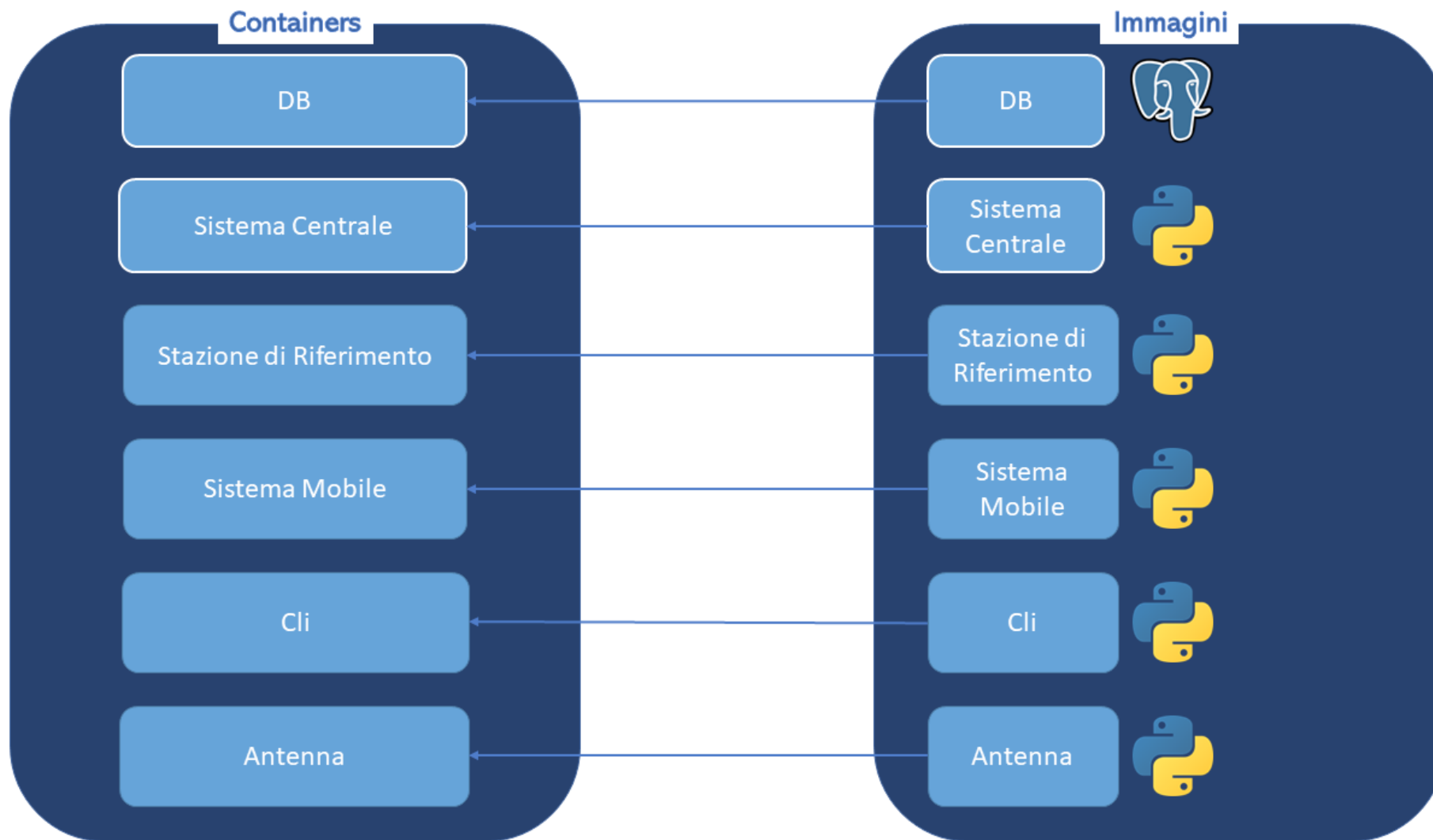


# Cli v2

```
# python cli.py
Connessione TCP con sistema mobile (host:sistemamobile, porta:12390)
digita 'help' per i comandi disponibili
>>stampa_lista_id
ID                                     Time      Date      Validazione
775408744375bf33ce8fc512b1b483e612903ff640fdd28856c03a5236fc948c  14:45:26  2022-07-25  True
>>info_id 775408744375bf33ce8fc512b1b483e612903ff640fdd28856c03a5236fc948c
{'timestamp': 1658760326, 'latitudine': '67:Nord', 'longitudine': '39:Ovest', 'altitudine': 1315}
>>verifica_id 775408744375bf33ce8fc512b1b483e612903ff640fdd28856c03a5236fc948c
Lo snapshot con id 775408744375bf33ce8fc512b1b483e612903ff640fdd28856c03a5236fc948c è stato certificato correttamente
>>
```



# Architettura del progetto con docker-compose



# Sviluppi futuri

- IPFS (Interplanetary File System)

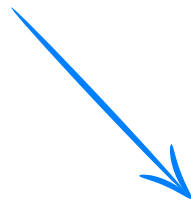


# Sviluppi futuri

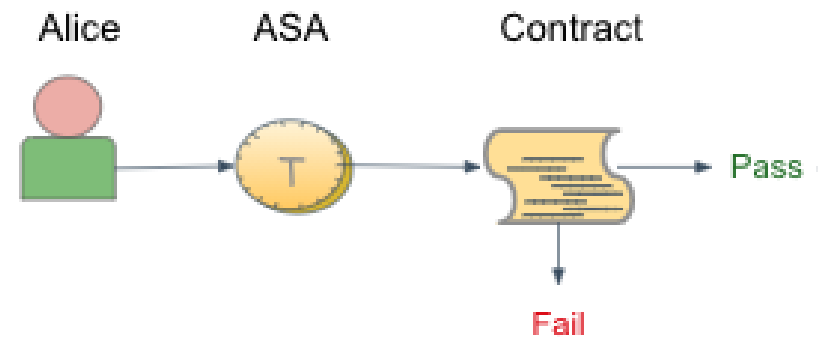
- IPFS (Interplanetary File System)



- Asset Token Frozen



Token da utilizzare come prova dell'autenticità del sistema mobile per richiamare lo smart contract.





**Grazie per l'attenzione!**

