

# Indoor Positioning Using the OpenHPS Framework

---

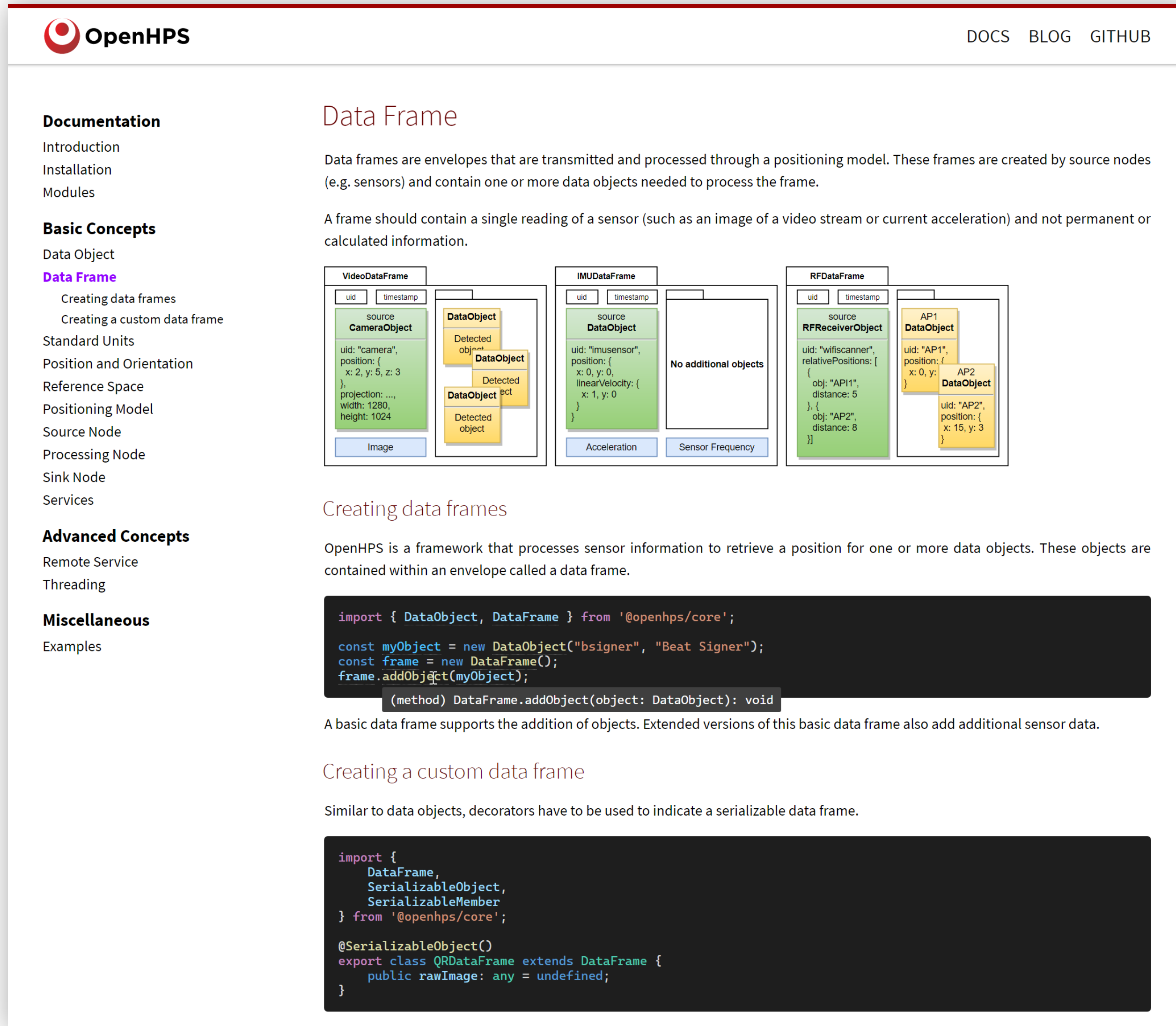
Maxim Van de Wynckel, Beat Signer

*Web & Information Systems Engineering Lab  
Vrije Universiteit Brussel*

# What is OpenHPS?



# An Open Source Hybrid Positioning System



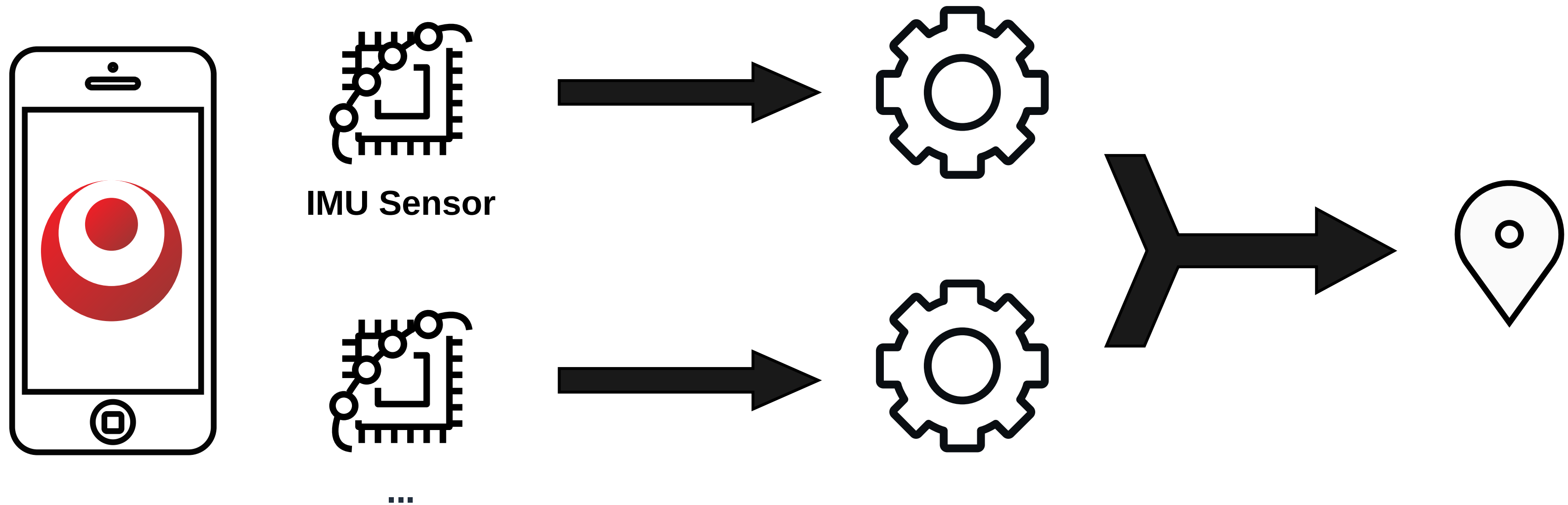
# What is OpenHPS?



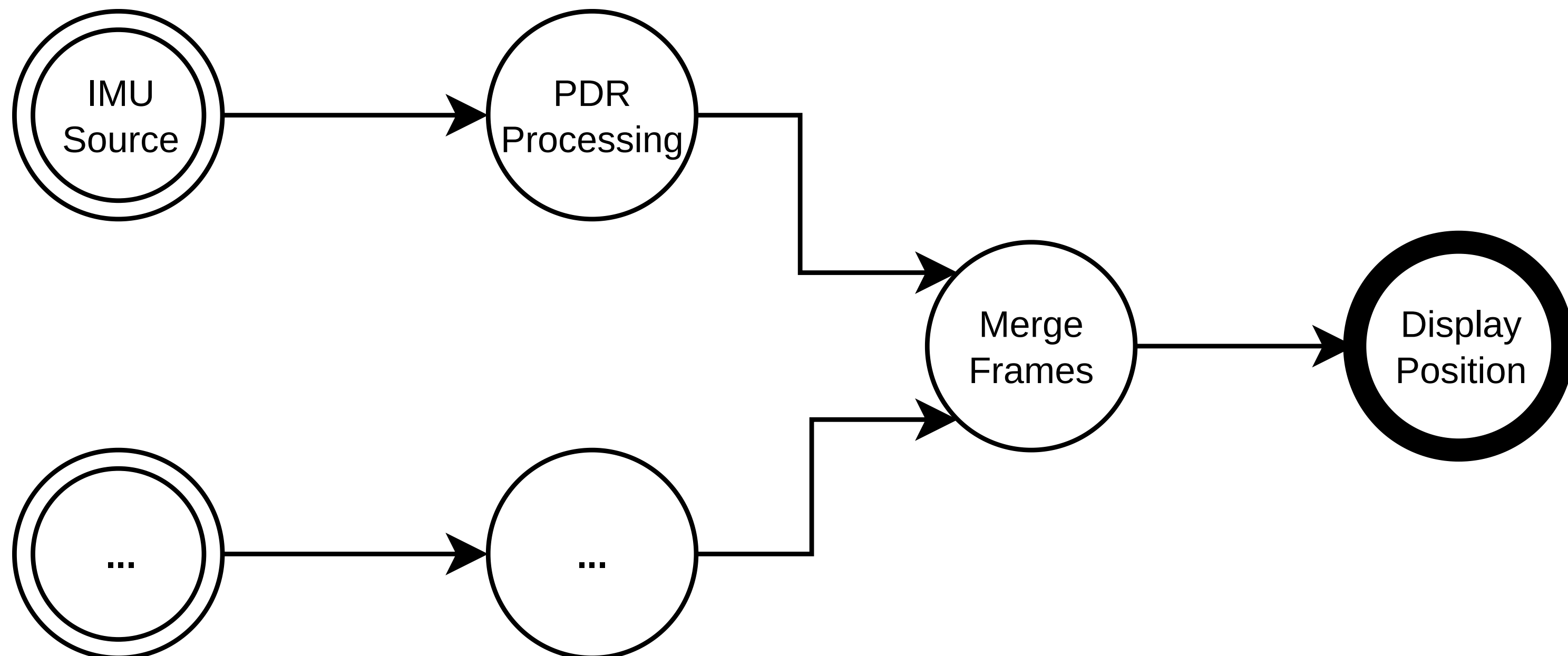
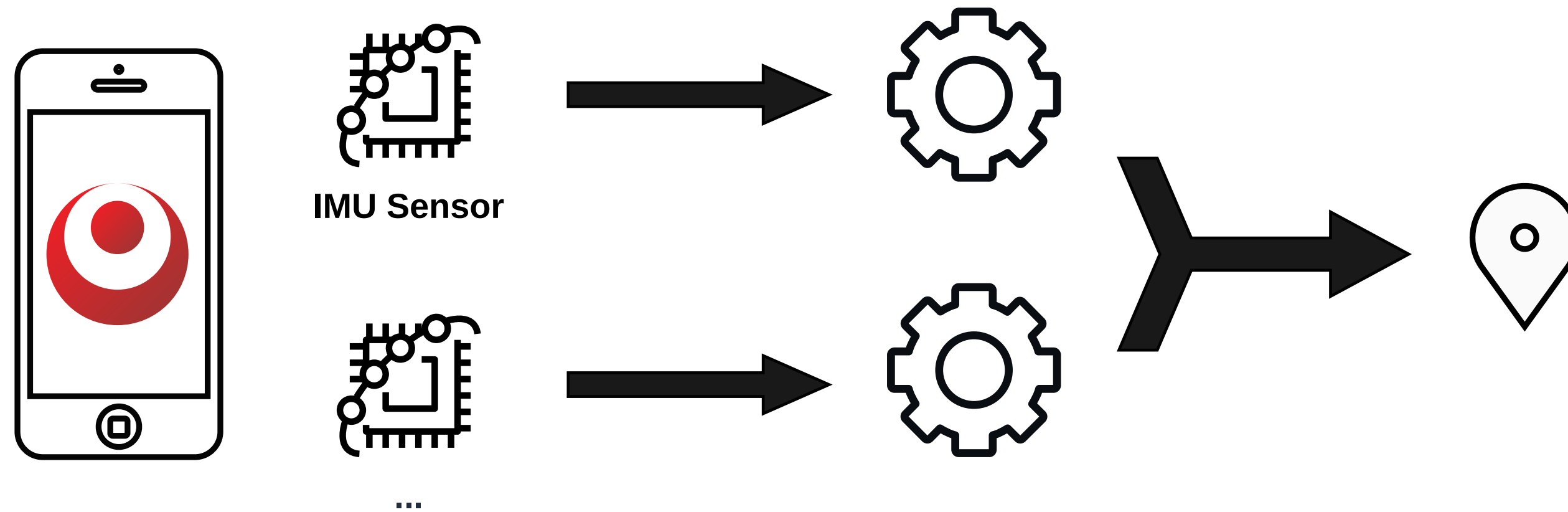
## An Open Source Hybrid Positioning System

- ▶ Any technology
- ▶ Any algorithm
- ▶ Various use cases
- ▶ Flexible processing and output
  - Accuracy over battery consumption, reliability, ...
- ▶ Aimed towards
  - Developers
  - Researchers

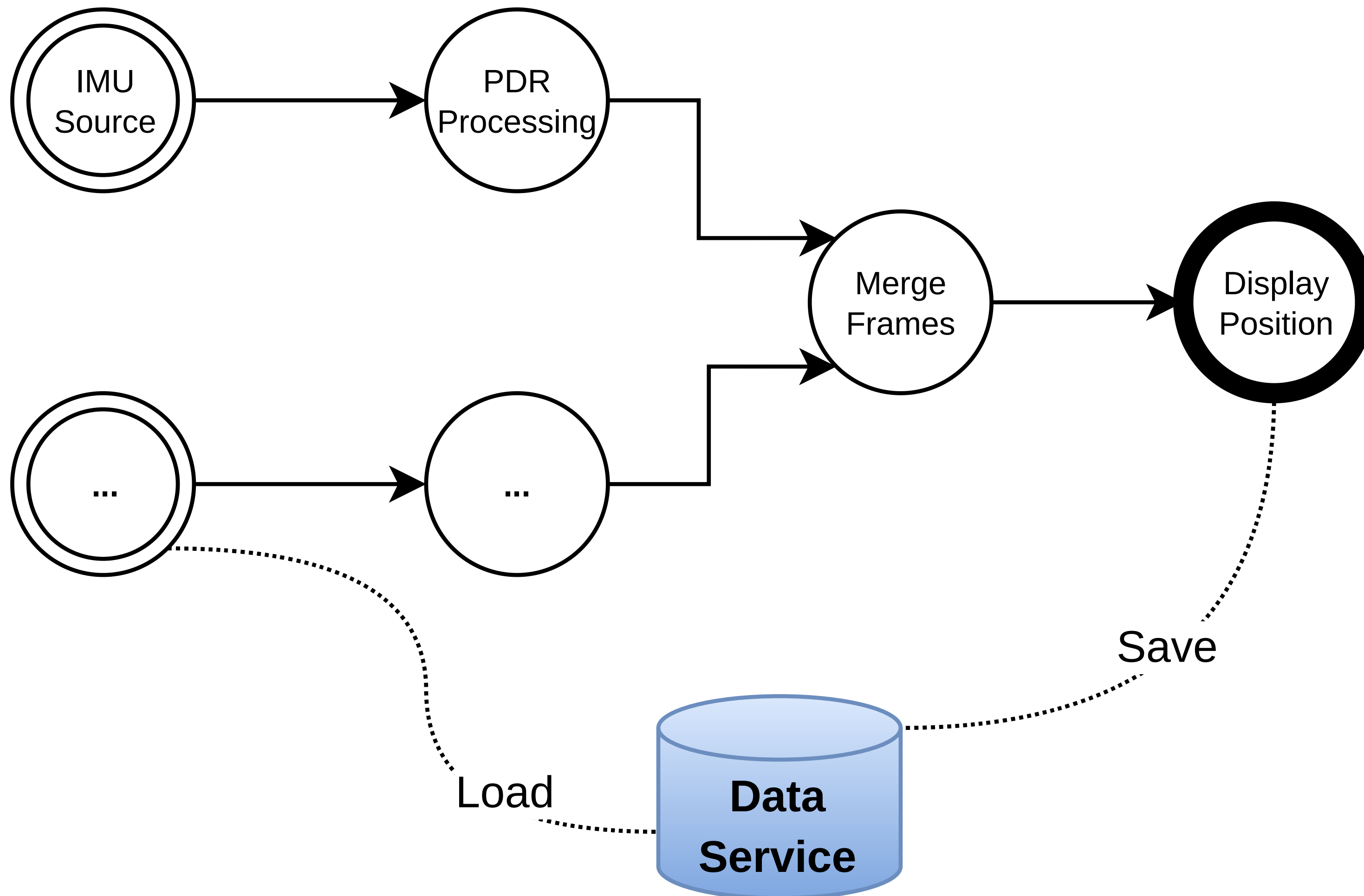
# Process Network Design



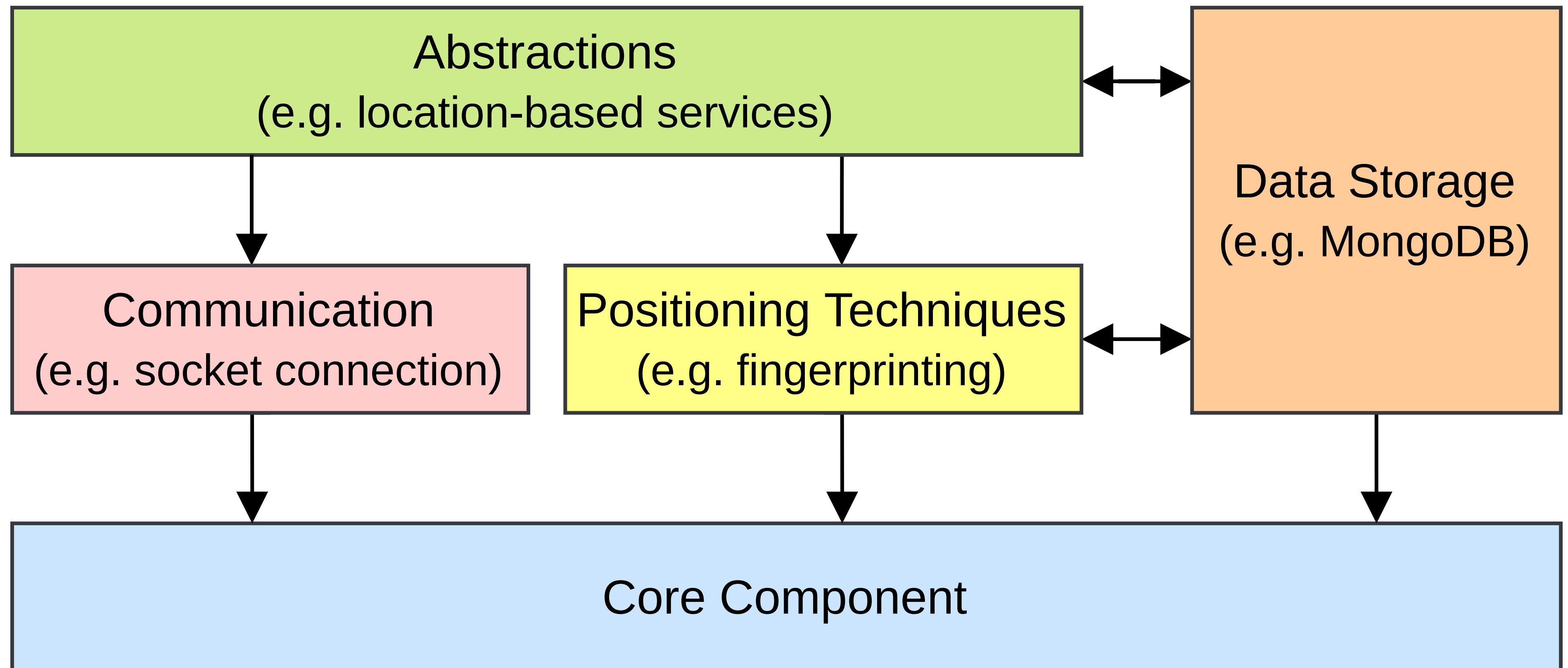
# Process Network Design ...



# Process Network Design ...



# Modularity



# Modularity ...

## **Communication**

Socket, MQTT, REST API, ...

## **Data Storage**

MongoDB, LocalStorage, RDF, ...

## **Positioning Algorithms**

IMU, fingerprinting, OpenVSLAM, ...

## **Abstractions**

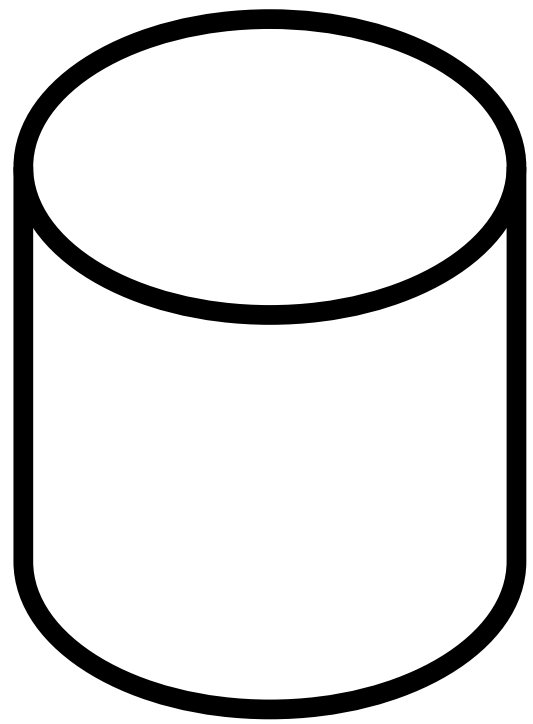
Geospatial, location-based services, geojson, ...

## **Other**

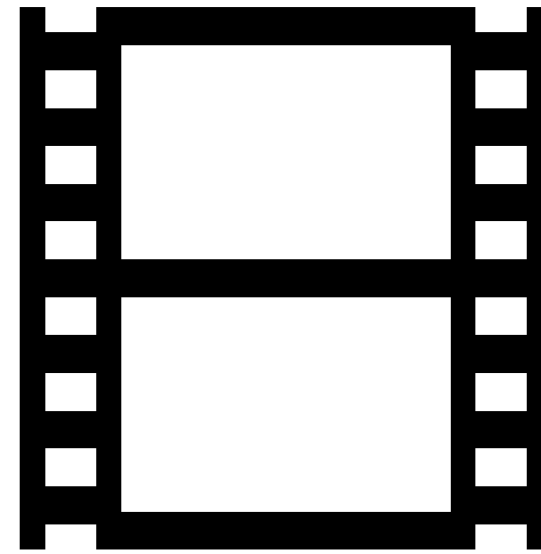
React-Native, NativeScript, Sphero, ...



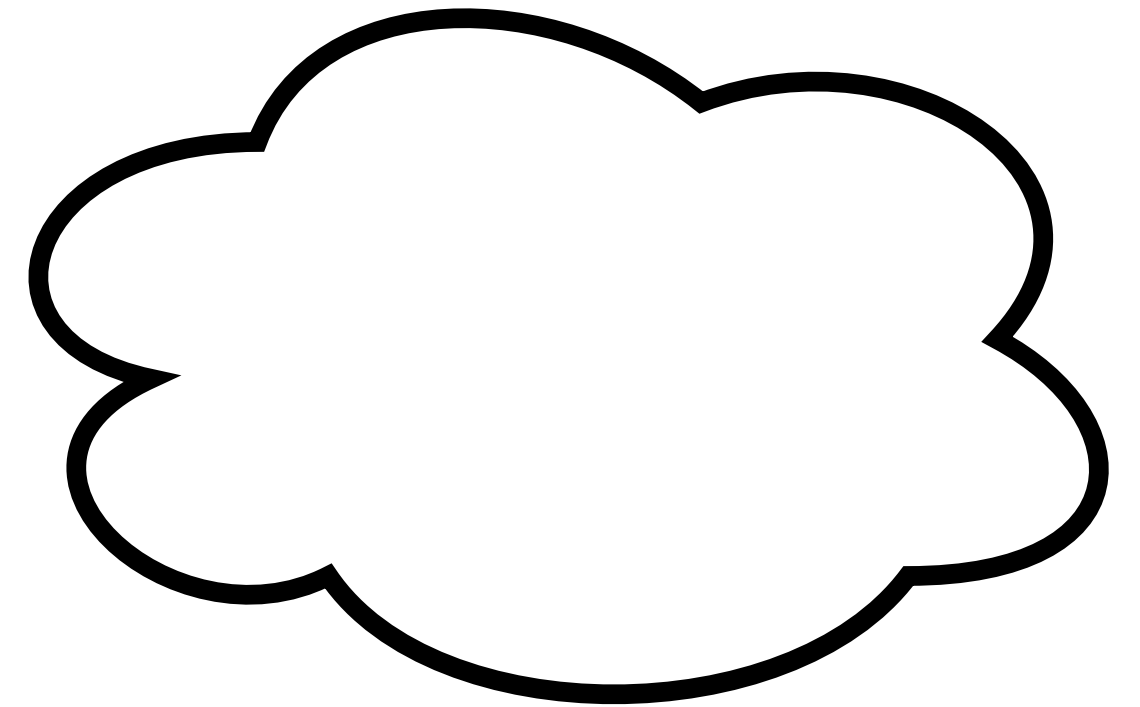
# Data Processing



**Knowledge**

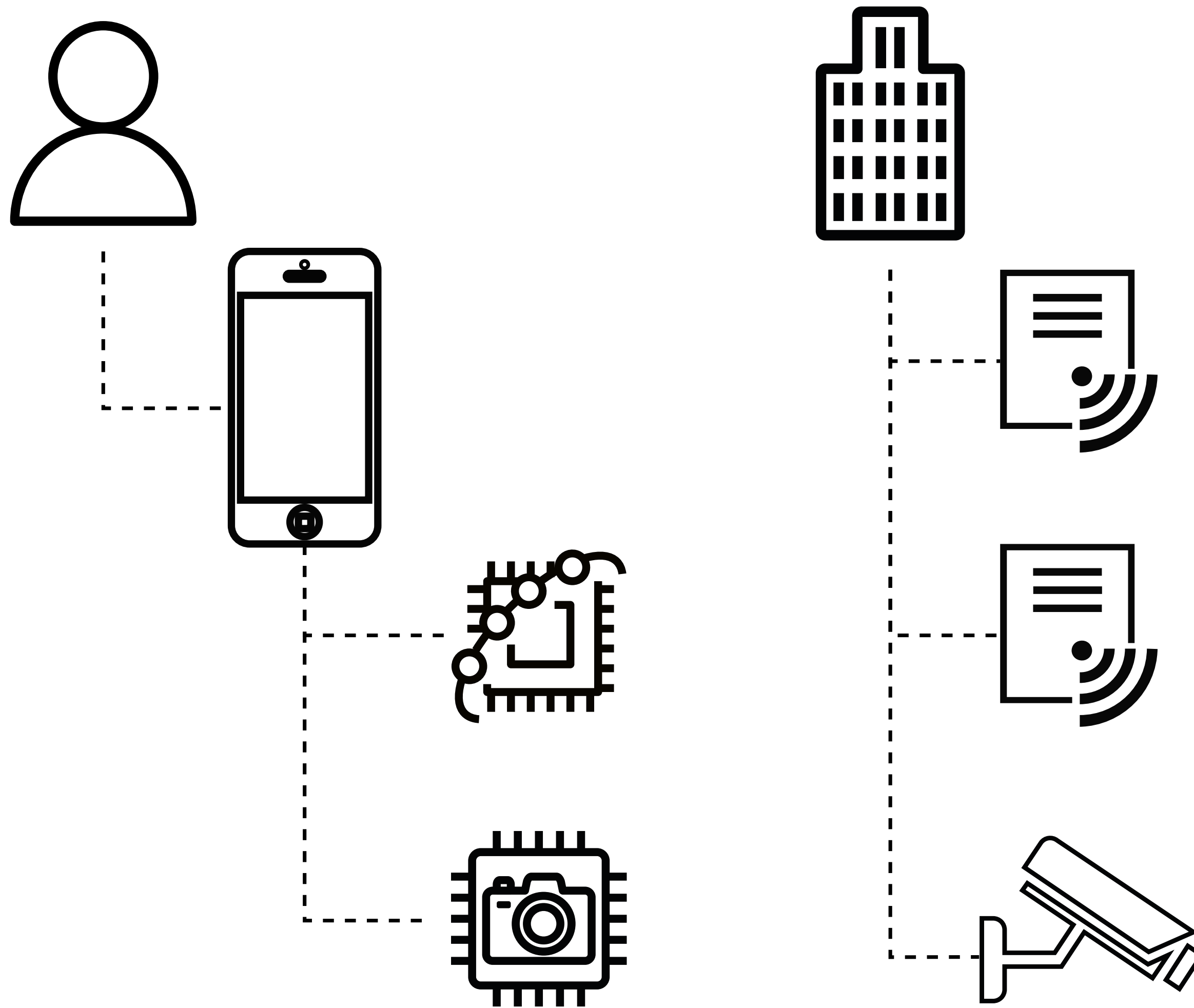


**Raw Data**



**Processed Data**

# DataObject



# Absolute and Relative Positions

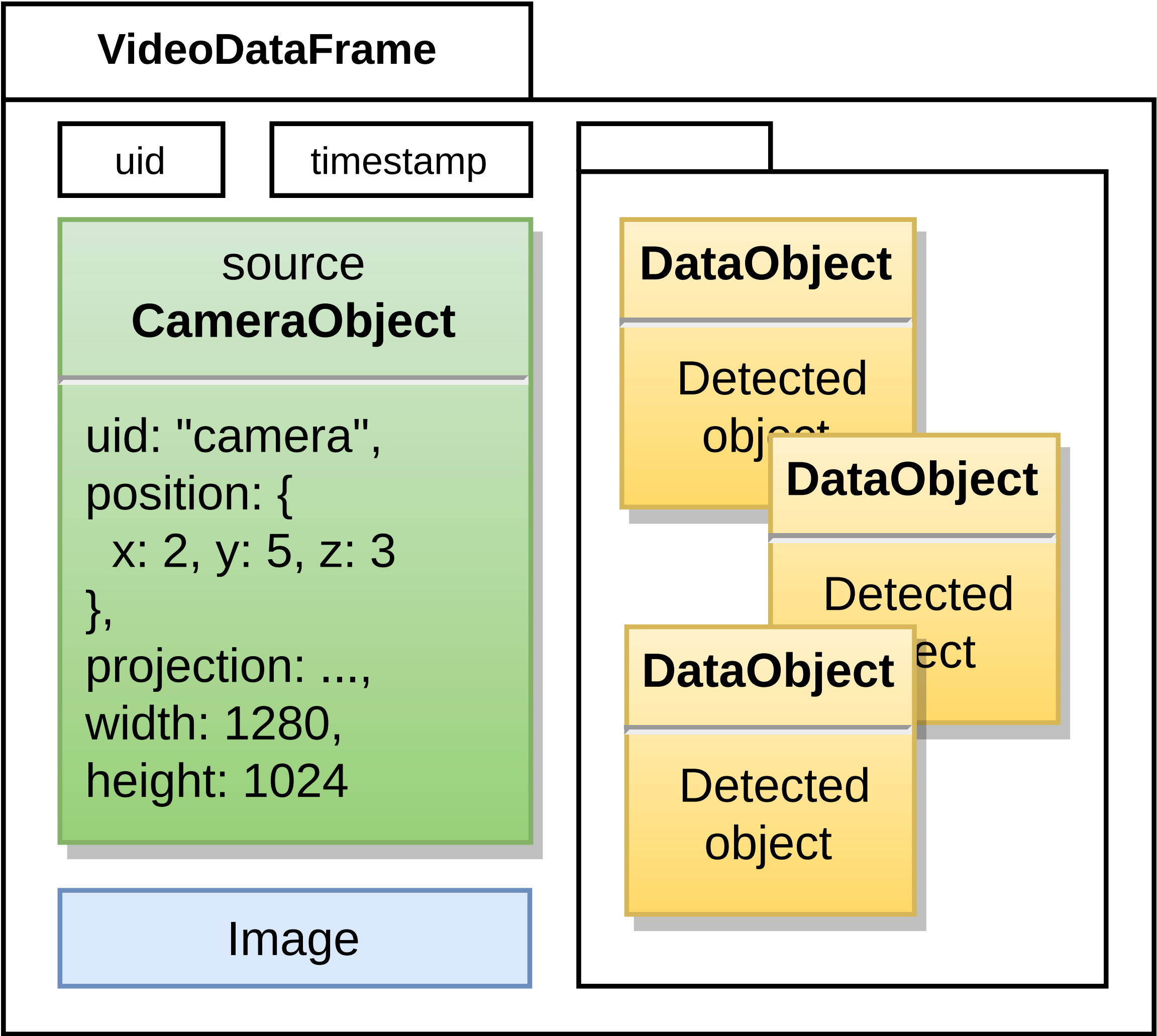
## Absolute

- ▶ 2D, 3D, Geographical, ...

## Relative

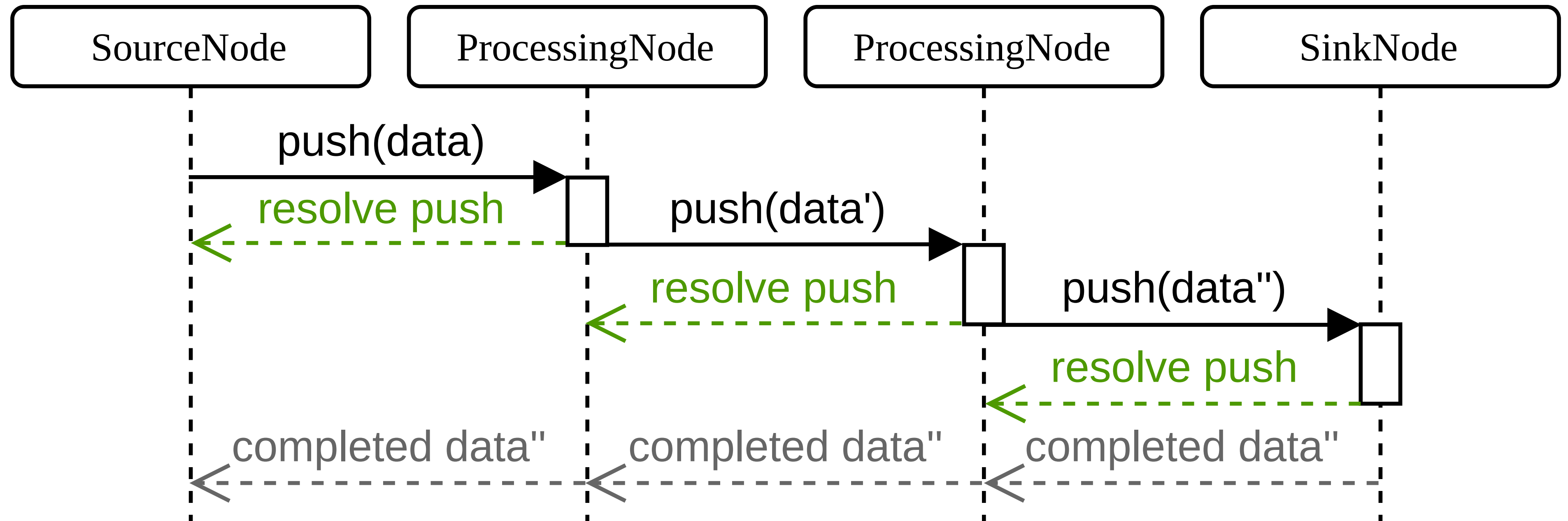
- ▶ Distance, angle, velocity, ...
- ▶ Relative to another *object*

# DataFrame



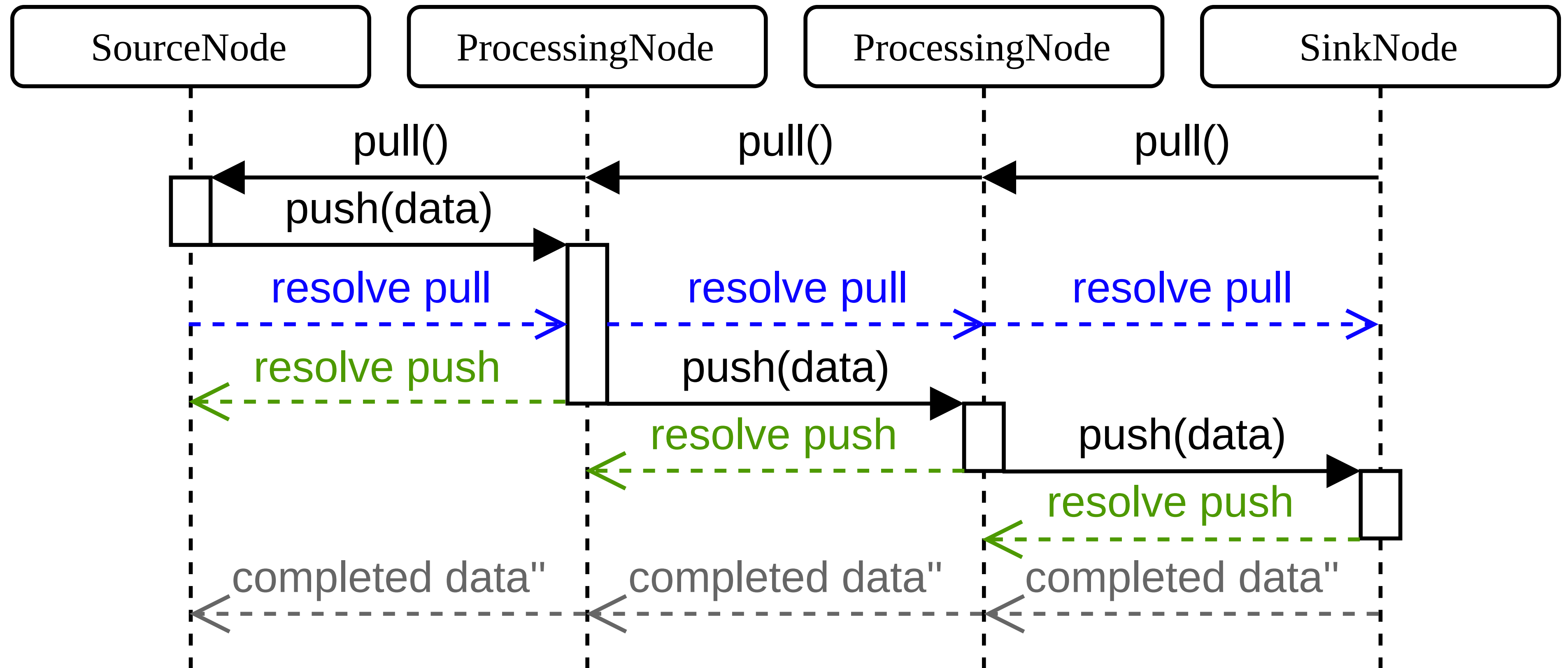
# DataFrame ...

## Pushing Data



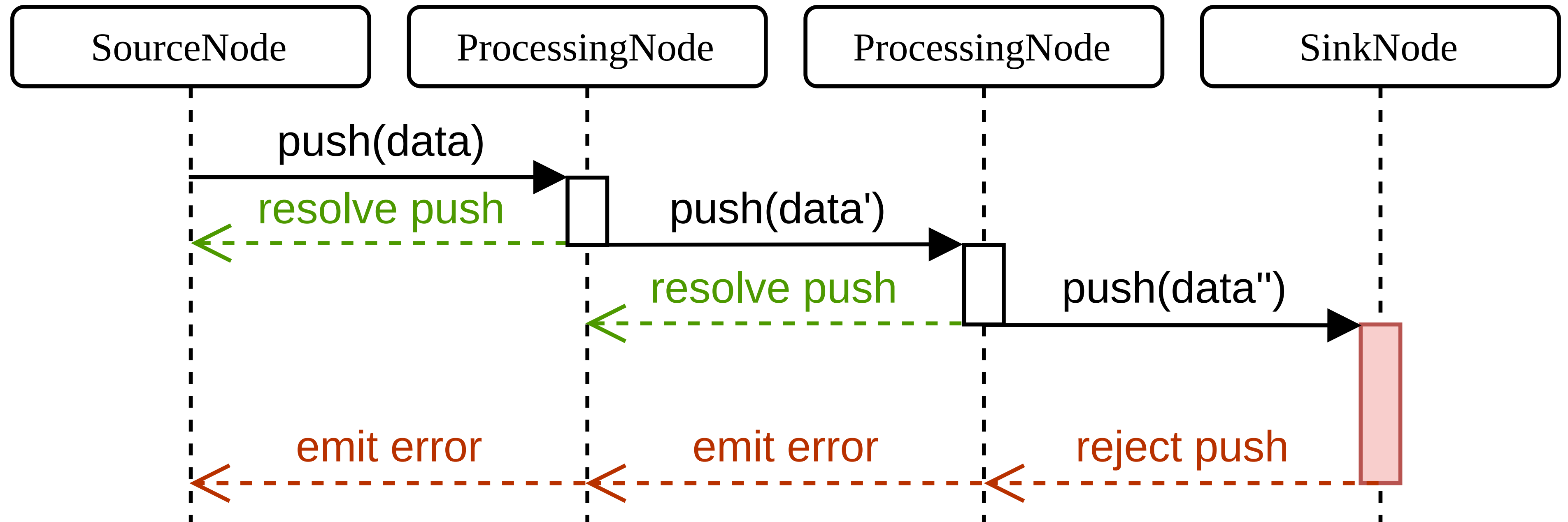
# DataFrame ...

## Pulling Data



# DataFrame ...

## Pushing Error



# SymbolicSpace

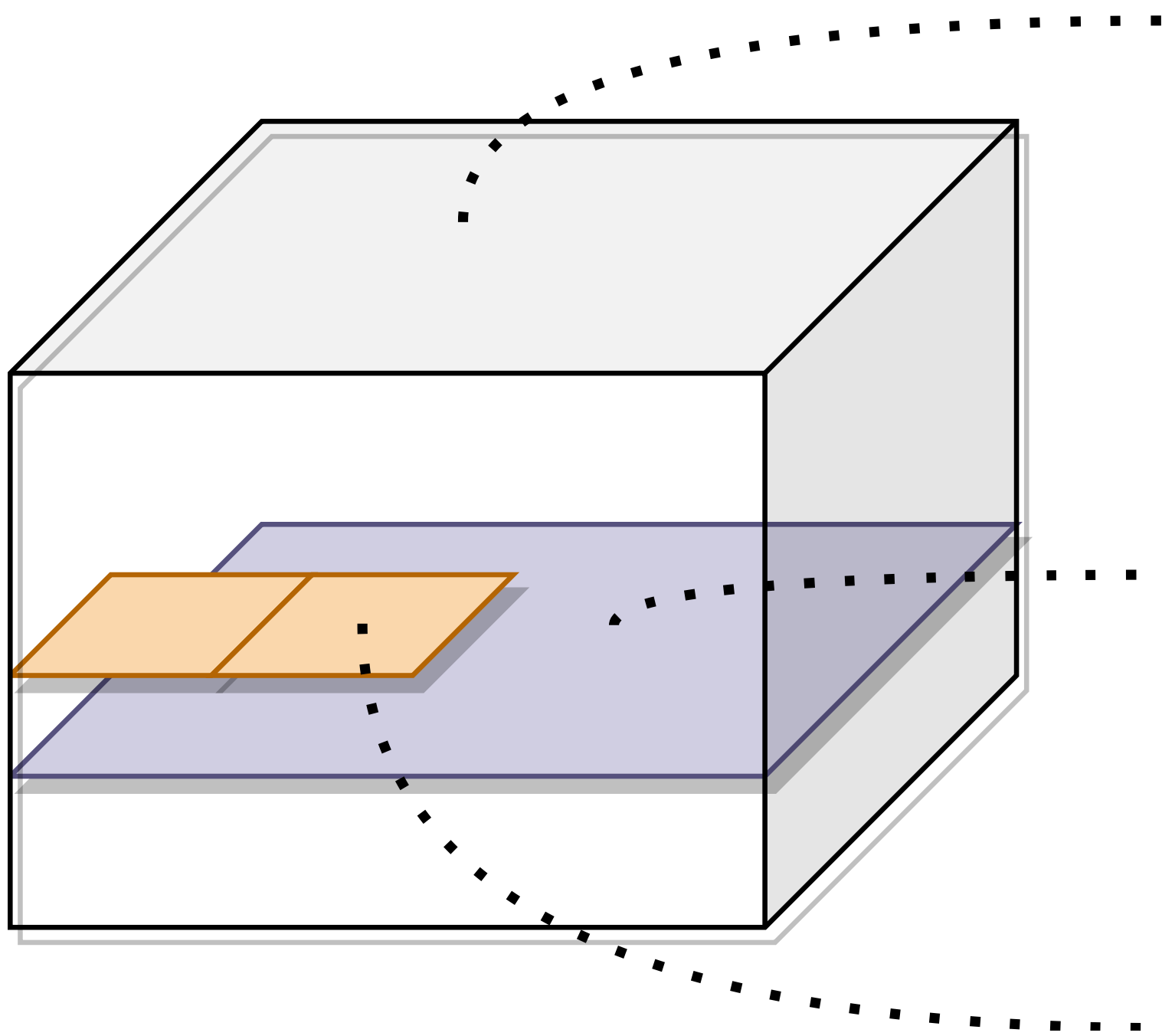
*An object that semantically defines a space*

- ▶ Spatial hierarchy
- ▶ Graph connectivity with other spaces
- ▶ Geocoding
- ▶ GeoJSON compatibility
- ▶ Can be used as a location
- ▶ Can be extended ...





# SymbolicSpace ...



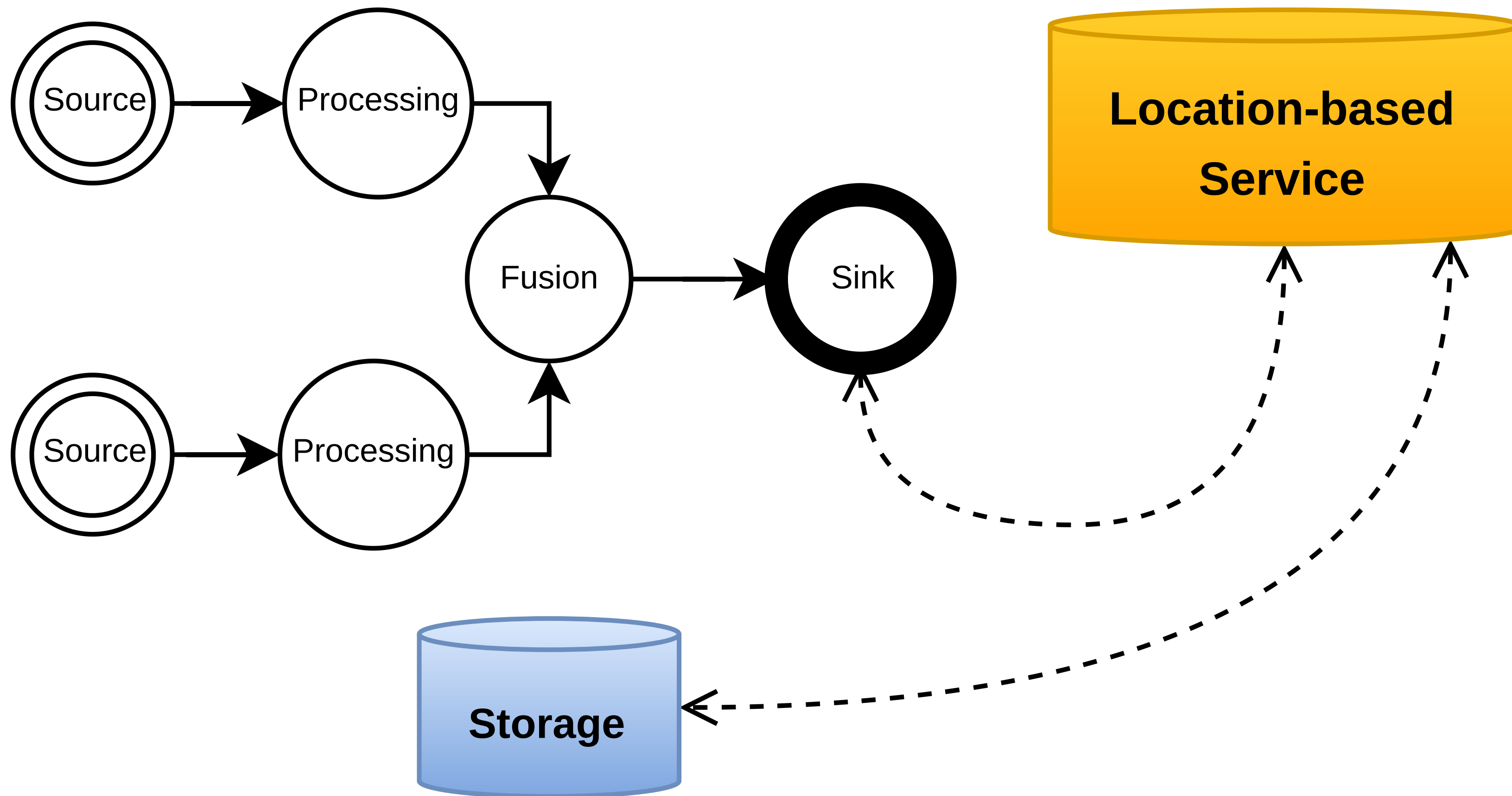
```
const building = new Building("PL9")
  .setBounds({
    topLeft: new GeographicalPosition(
      50.8203,
      4.3922),
    width: 46.275,
    height: 37.27,
    rotation: -34.04
  });

const floor = new Floor("PL9.3")
  .setBuilding(building)
  .setFloorNumber(3);

const office = new Room("PL9.3.58")
  .setFloor(floor)
  .setBounds([
    new Absolute2DPosition(4.75, 31.25),
    new Absolute2DPosition(8.35, 37.02),
  ]);
```

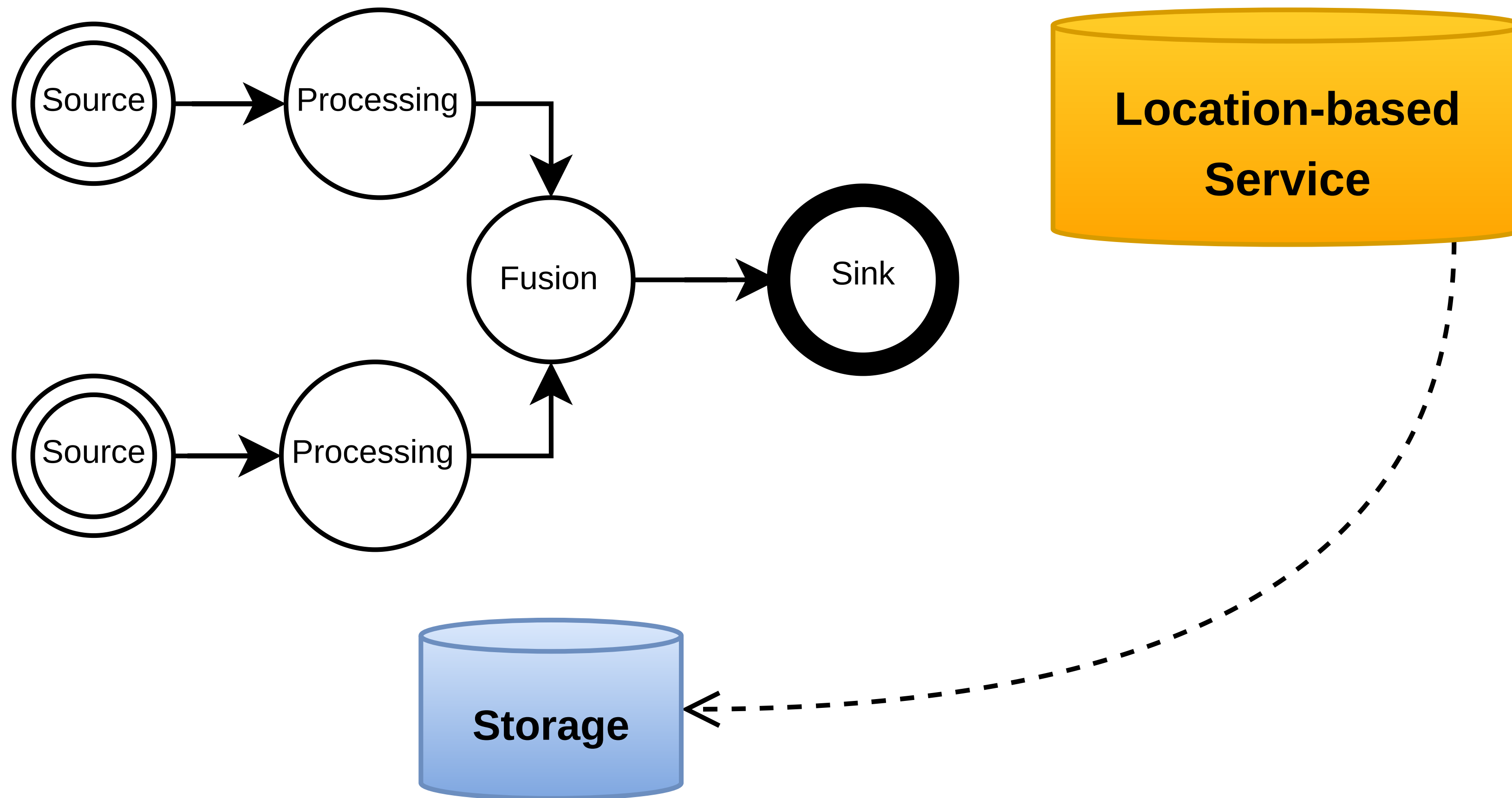
# Location-based Service

`getCurrentPosition("me", ...)`



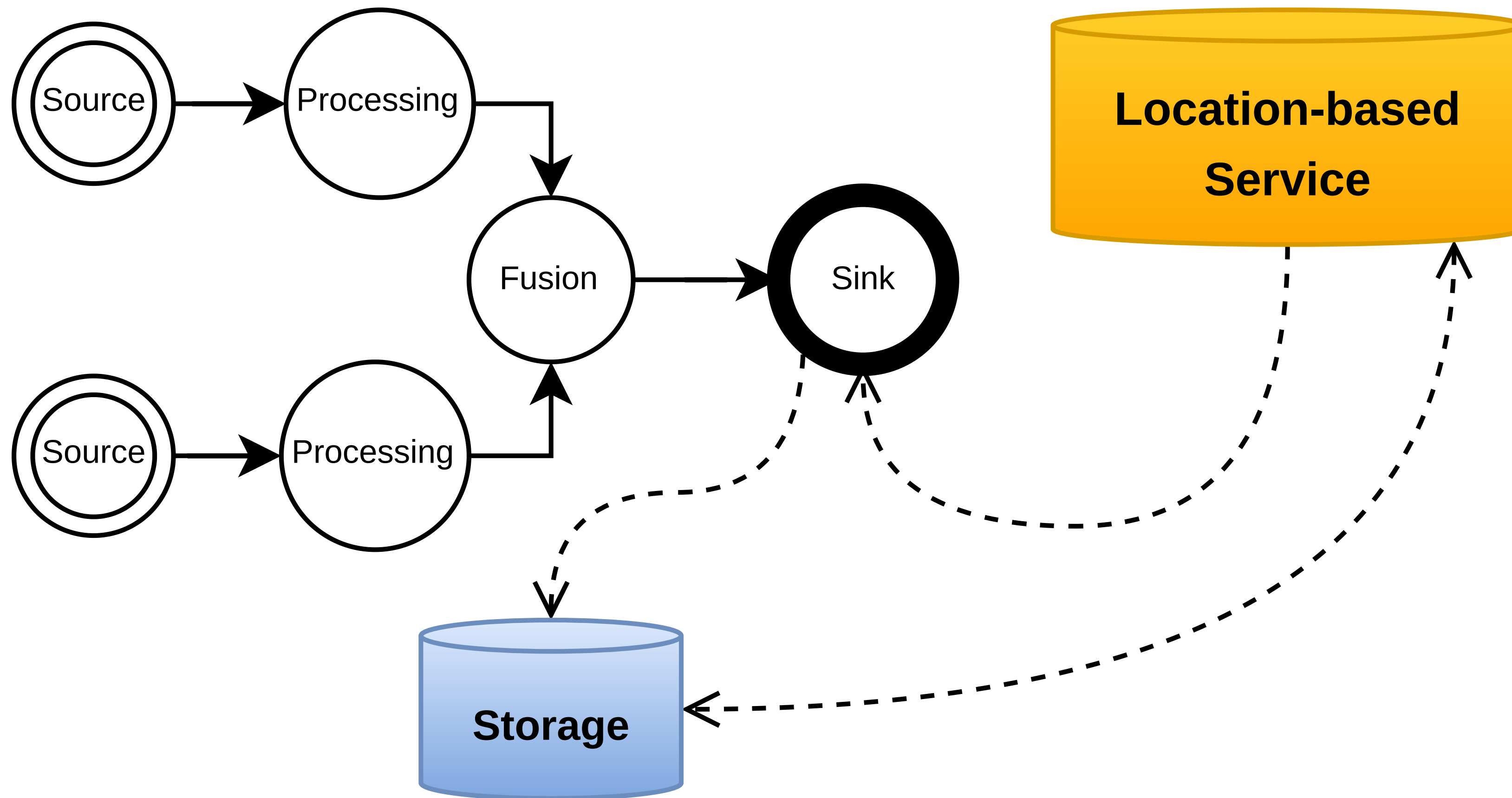
# Location-based Service ...

`setCurrentPosition("me", ...)`



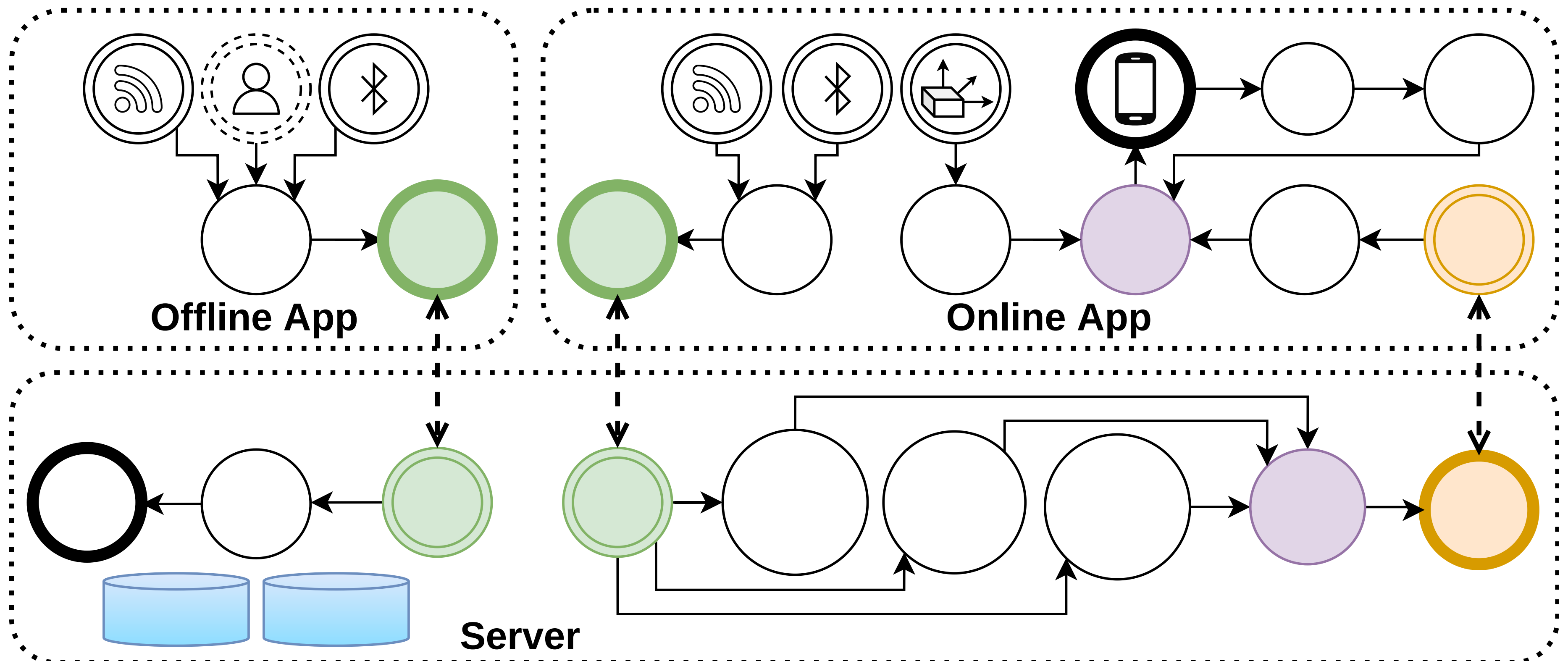
# Location-based Service ...

`watchPosition("me", ...)`

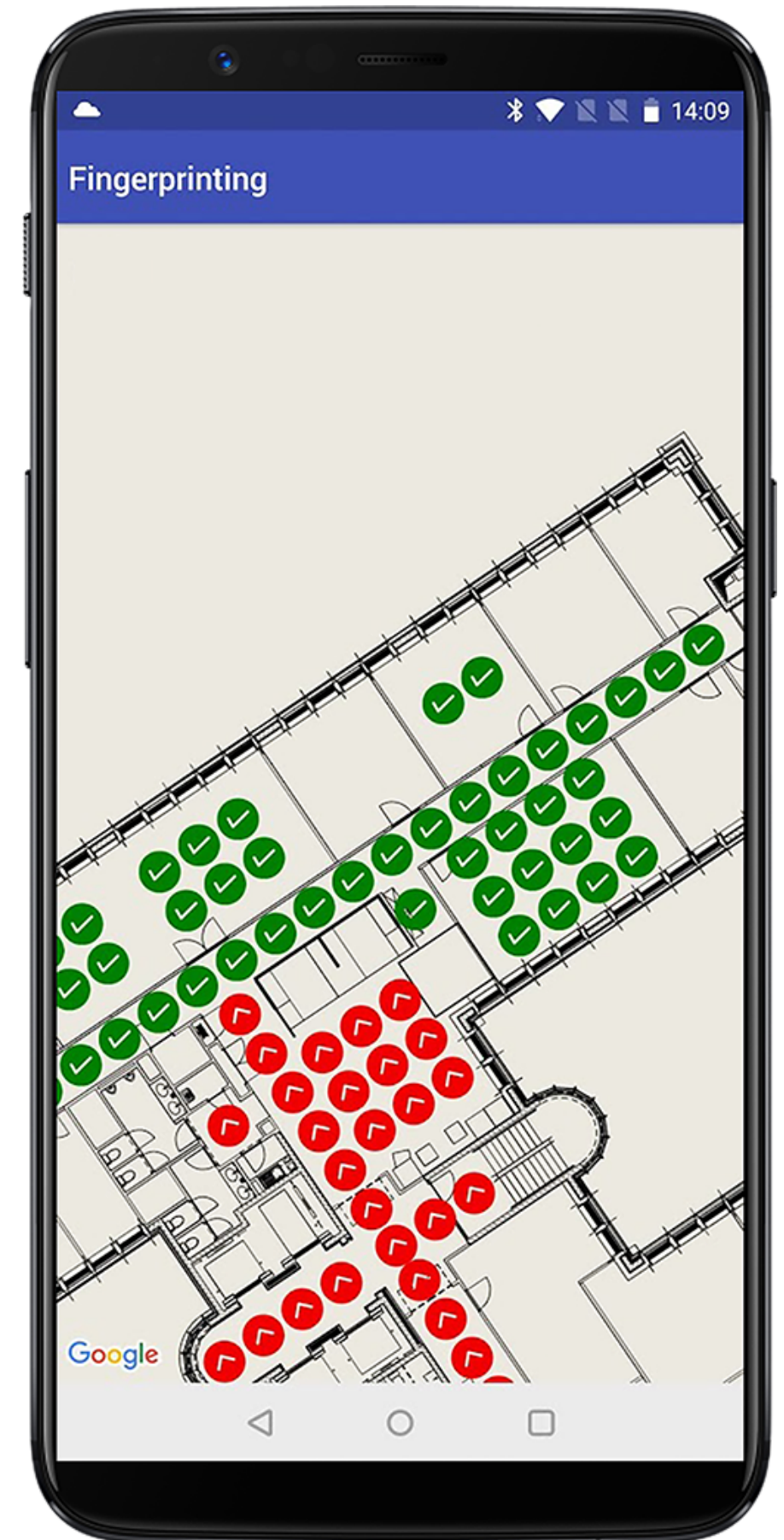
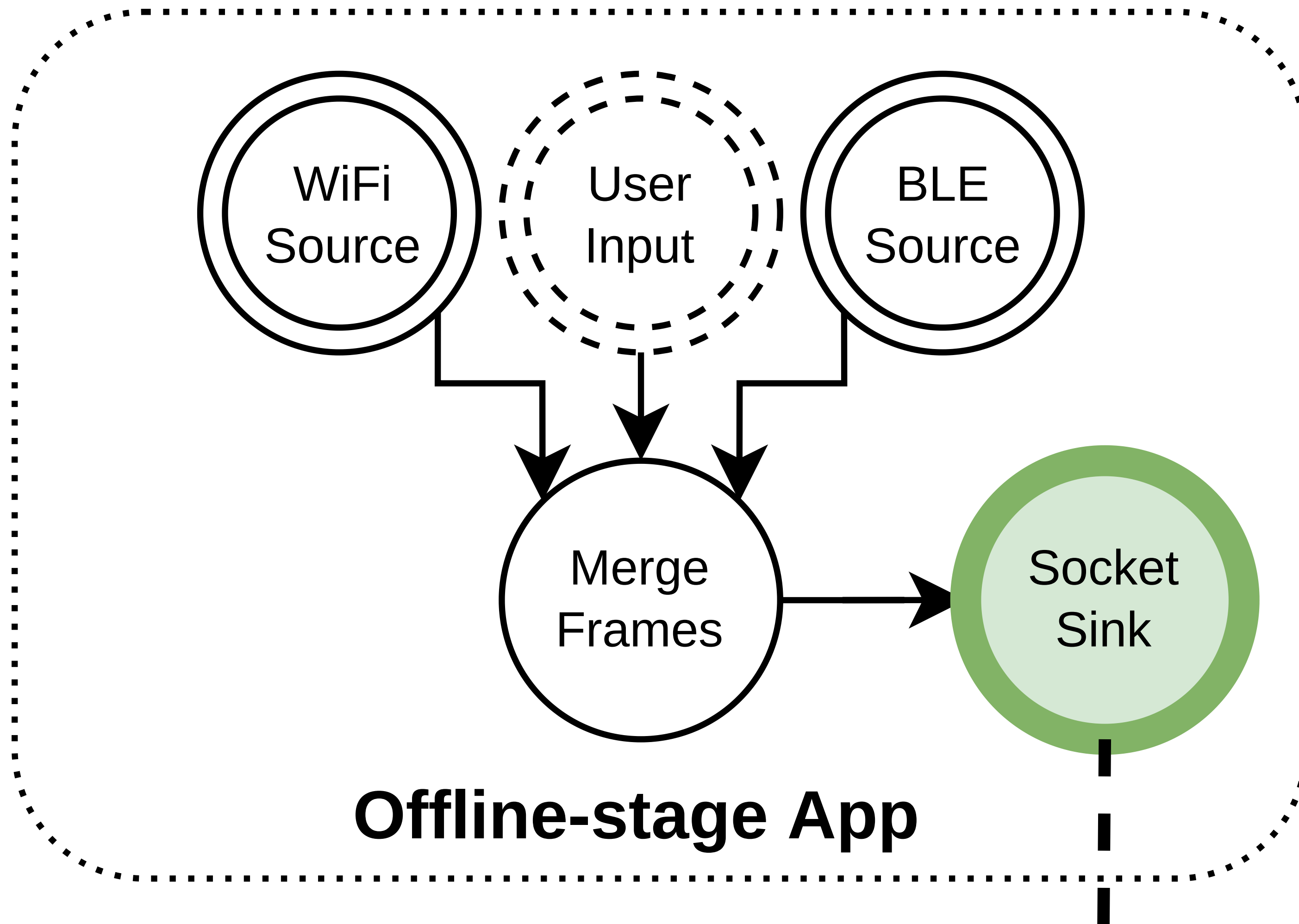


# Demonstration

- ▶ Indoor positioning **use case**
- ▶ Use **existing techniques**
- ▶ Validation of **flexibility** and modularity

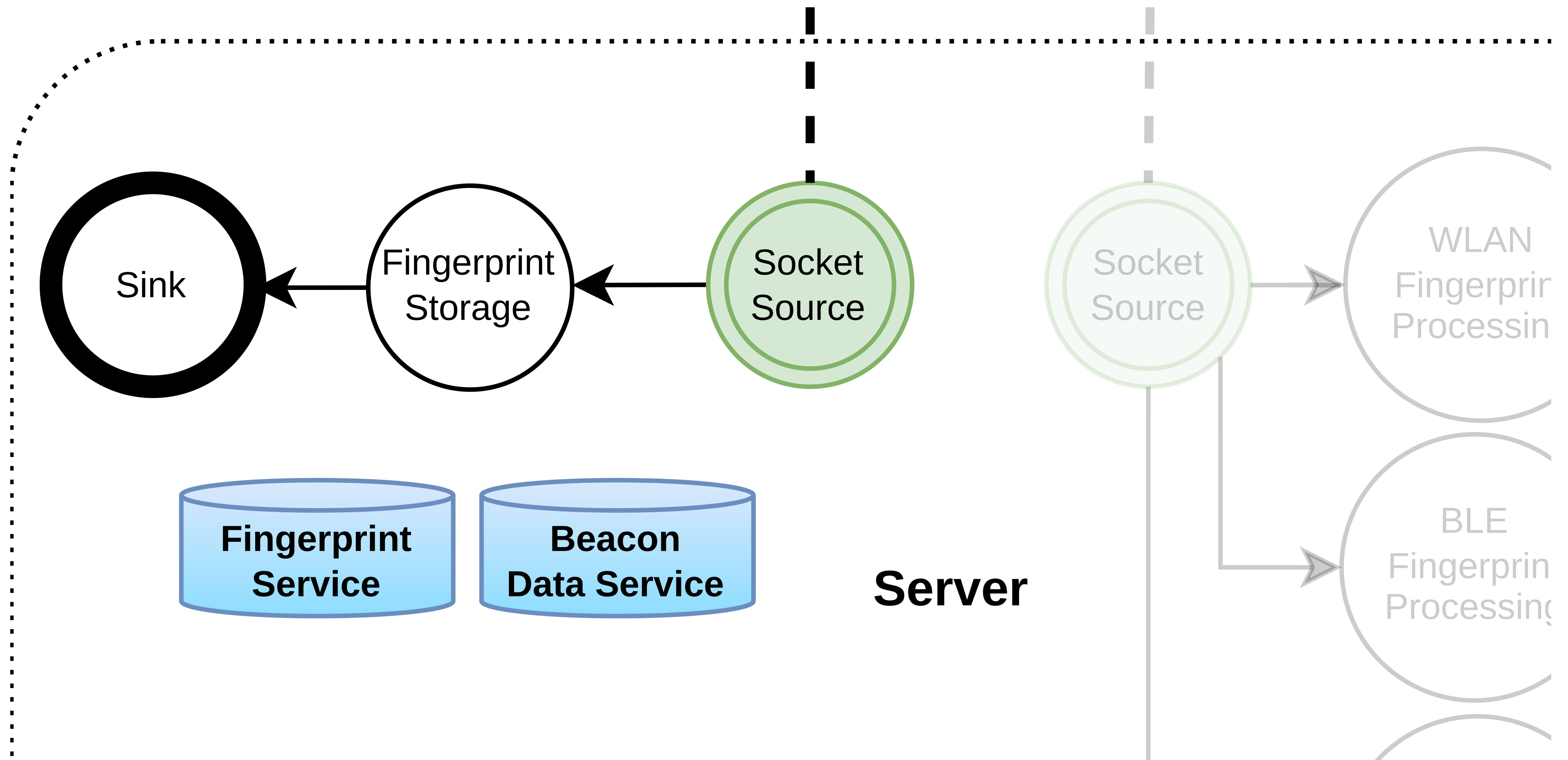


# Positioning Model

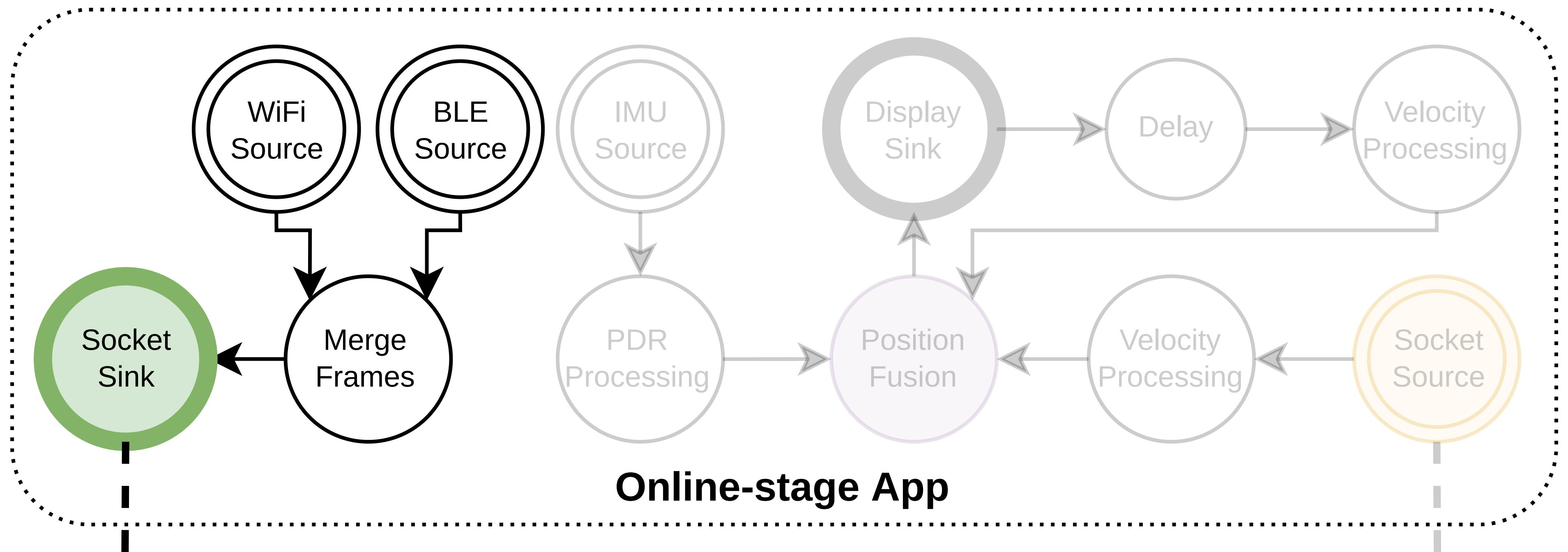




# Positioning Model ...

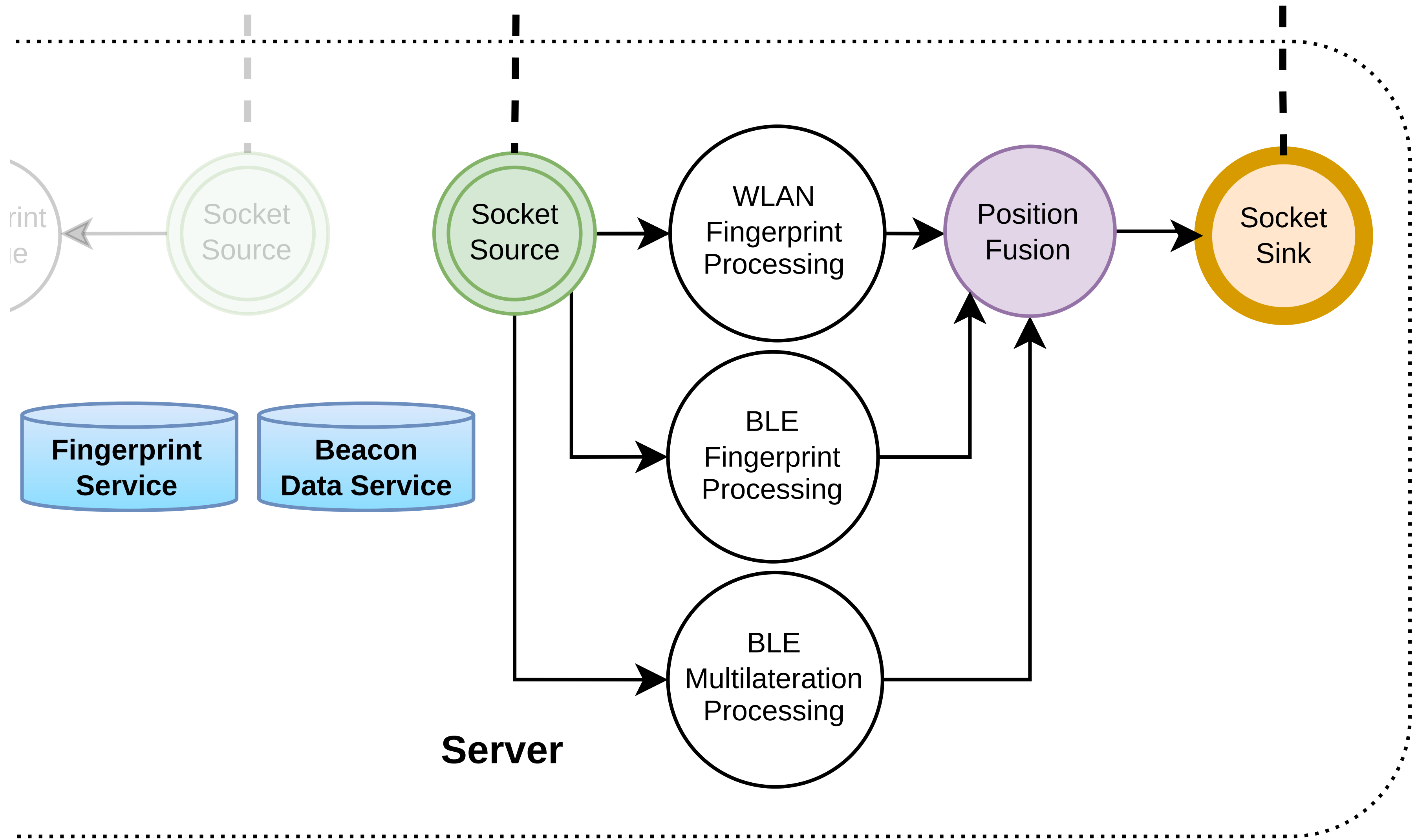


# Positioning Model ...

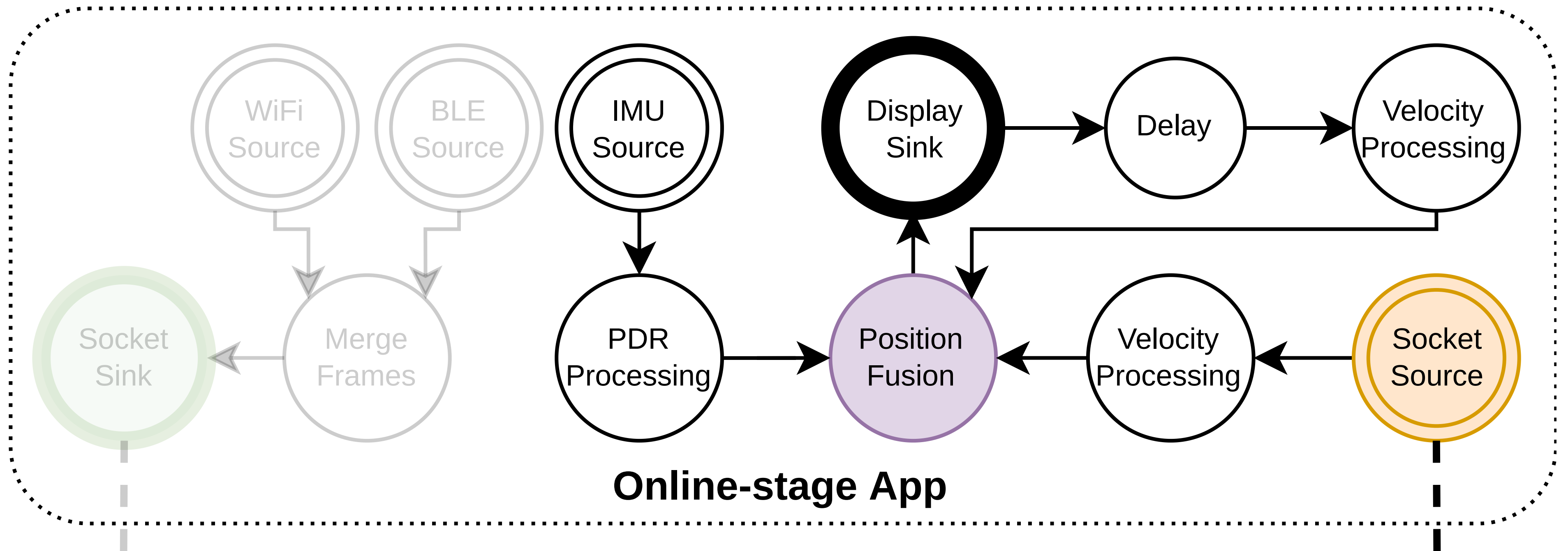




# Positioning Model ...

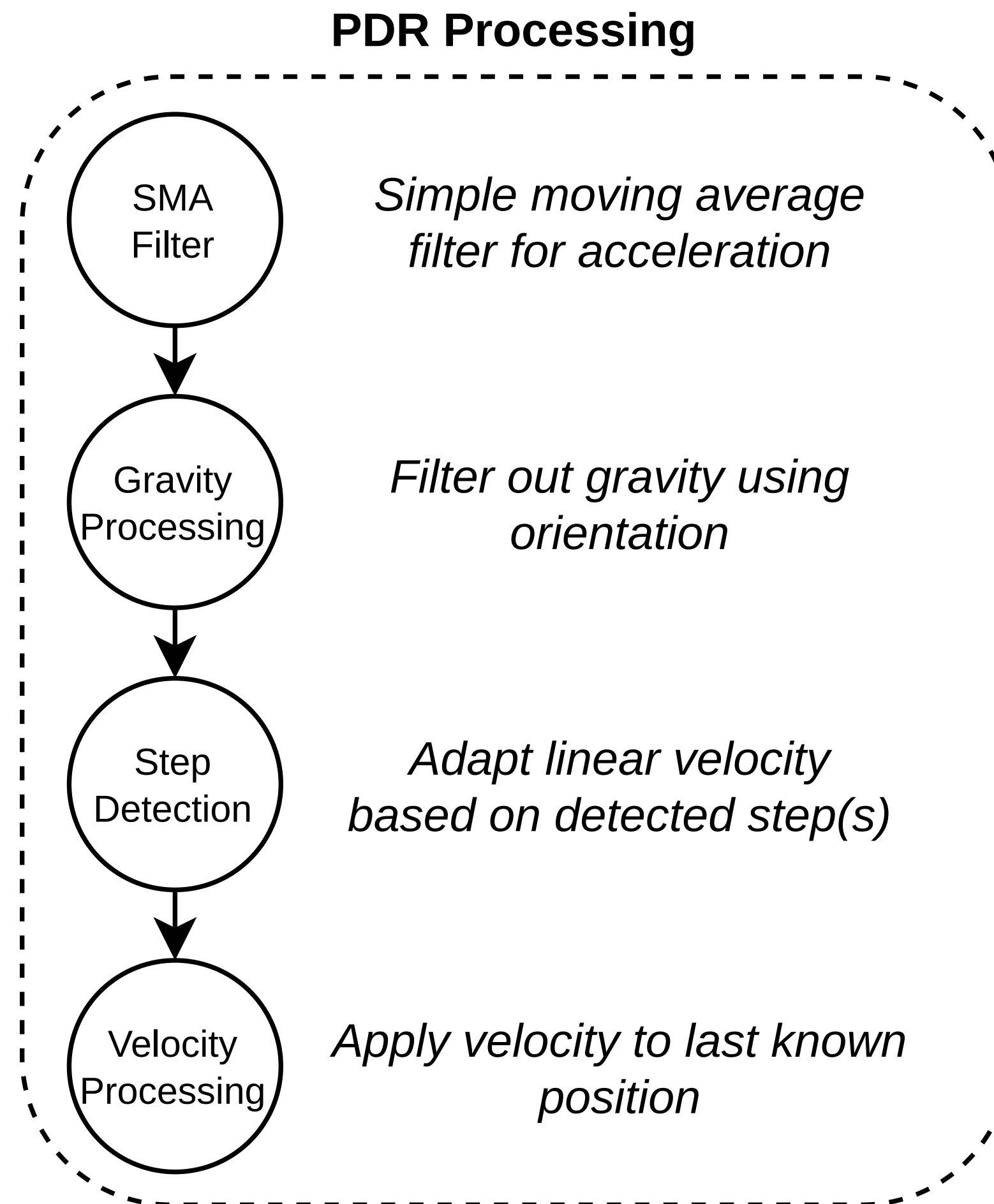


# Positioning Model ...



# Positioning Model

## Online App

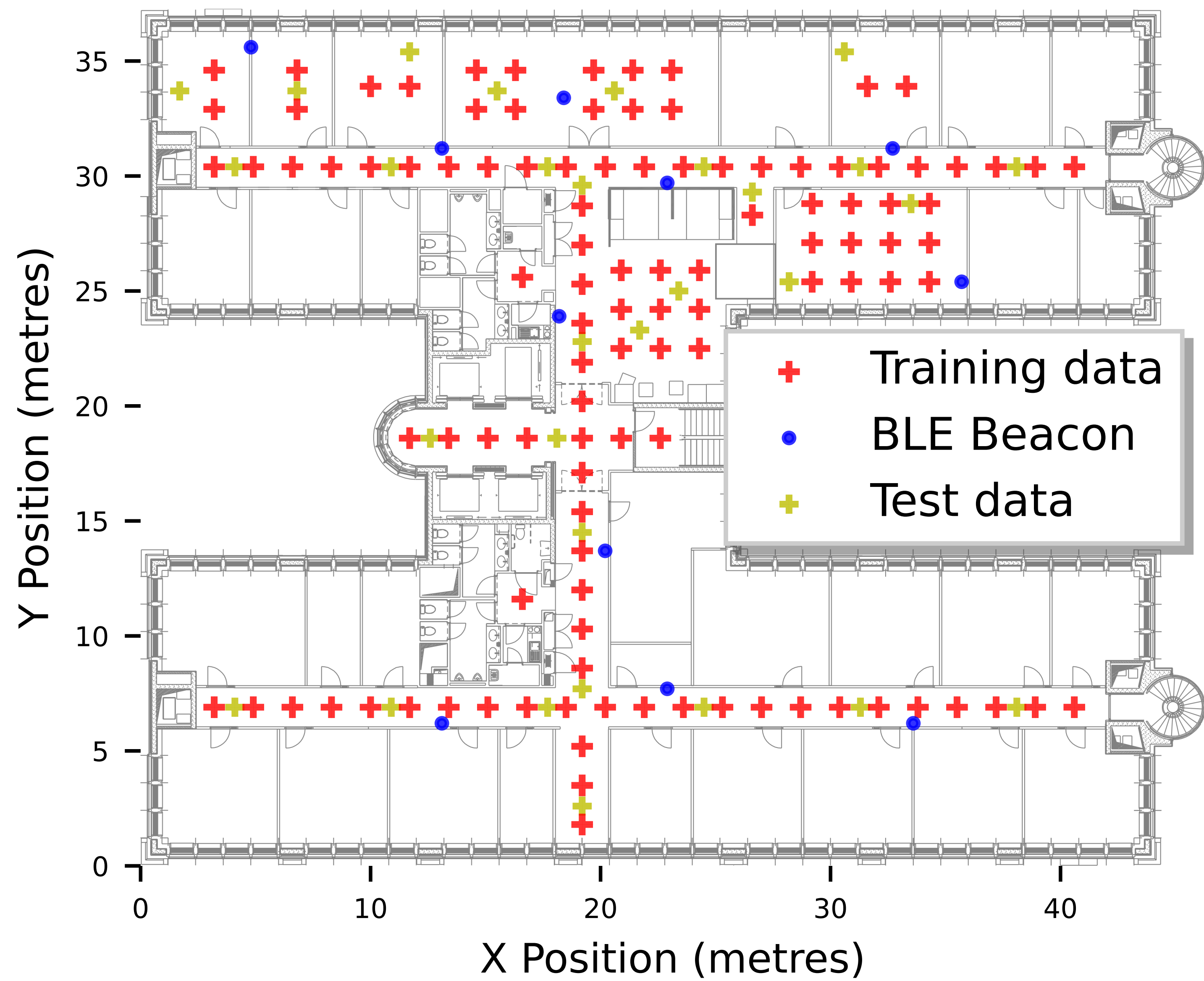


# Positioning Model

## Online App

```
ModelBuilder.create()
  .addShape(GraphBuilder.create()
    .from(new IMUSourceNode({
      source: new DataObject(phoneUID),
      interval: 20,
      sensors: [
        SensorType.ACCELEROMETER,
        SensorType.ORIENTATION
      ]
    })))
  .via(new SMAFilterNode(
    frame => [frame, "acceleration"],
    { taps: 10 }
  ))
  .via(new GravityProcessingNode({
    method: GravityProcessingMethod.ABSOLUTE_ORIENTATION
  })))
```

# Dataset



# Dataset ...

**Total BLE Beacons: 11**

**Total detected WLAN access points: 220**

**Total stable WLAN access points: 199**

|  | Training | Test |
|--|----------|------|
| <b>Datapoints</b>                                | 110      | 30   |
| <b>Total fingerprints</b>                        | 440      | 120  |
| <b>Duration</b> (per orientation)                | 20s      | 20s  |
| <b>Avg. WLAN Scans</b> (per fingerprint)         | 6        | 6    |
| <b>Avg. BLE Advertisements</b> (per fingerprint) | 16       | 15   |

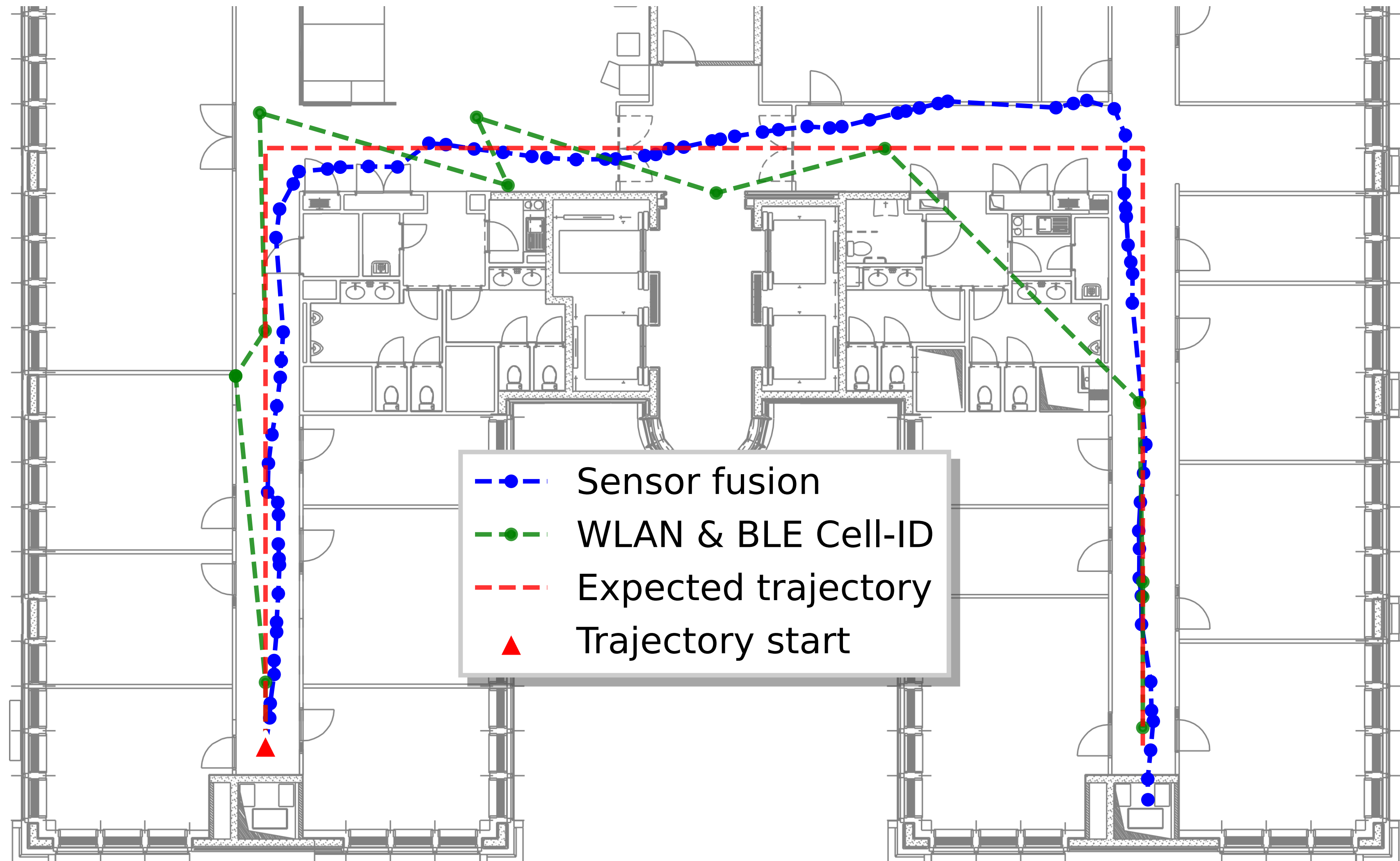
# Validation Results

## Static Positioning

|                      | <b>WLAN</b> fingerprinting | <b>BLE</b> fingerprinting | <b>BLE</b> multilateration | <b>Fusion</b>  |
|----------------------|----------------------------|---------------------------|----------------------------|----------------|
| <i>failed points</i> | 0                          | 6                         | 12                         | 0              |
| <i>average error</i> | 1.23 m                     | 3.23 m                    | 4.92 m                     | 1.37 m         |
| <i>minimum error</i> | 0.01 m                     | 0.17 m                    | 0.74 m                     | 0.01 m         |
| <i>maximum error</i> | 4.77 m                     | 15.39 m                   | 19.26 m                    | 9.75 m         |
| <i>hit rate</i>      | 95.82 %                    | 80.83 %                   | 52.50 %                    | <b>96.67 %</b> |

# Validation Results ...

## Trajectories

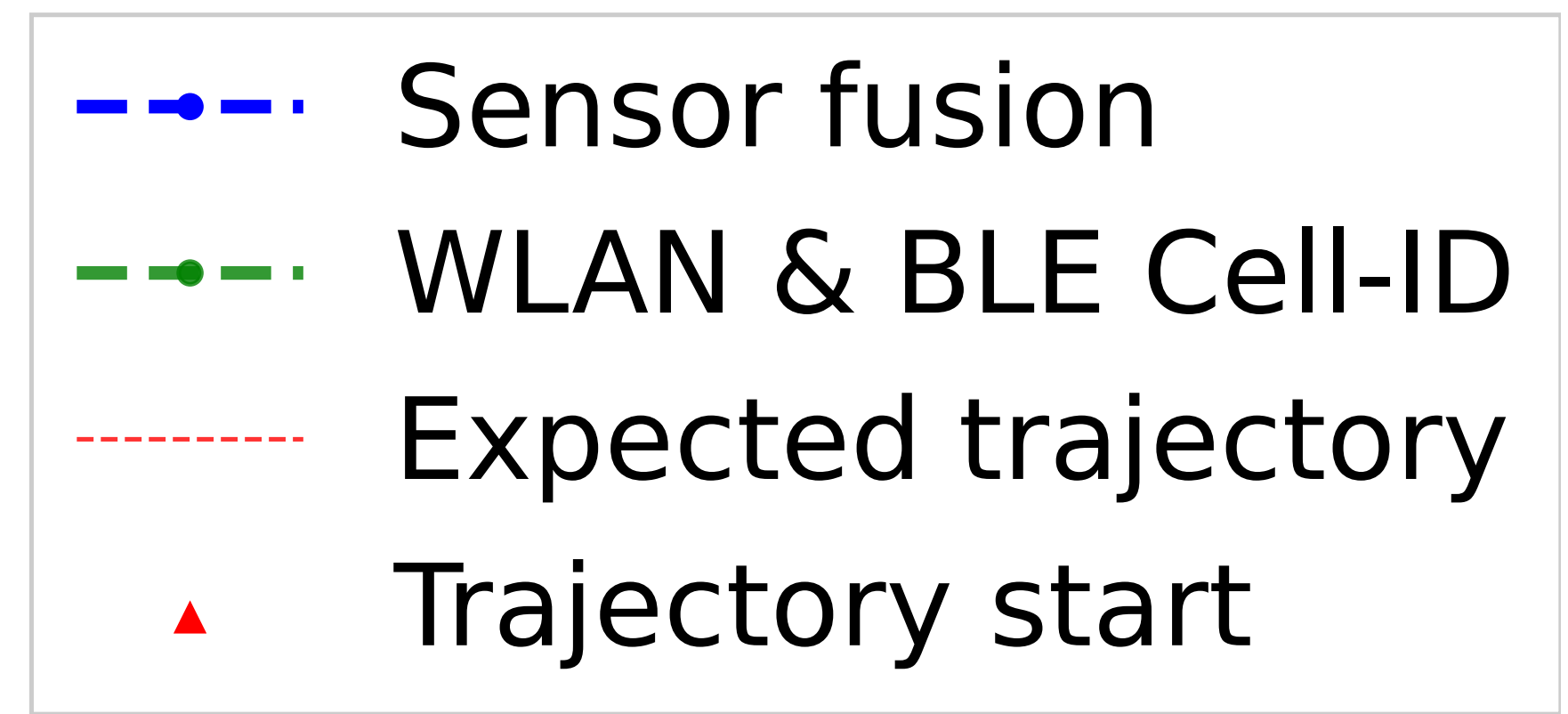




# Validation Results ...

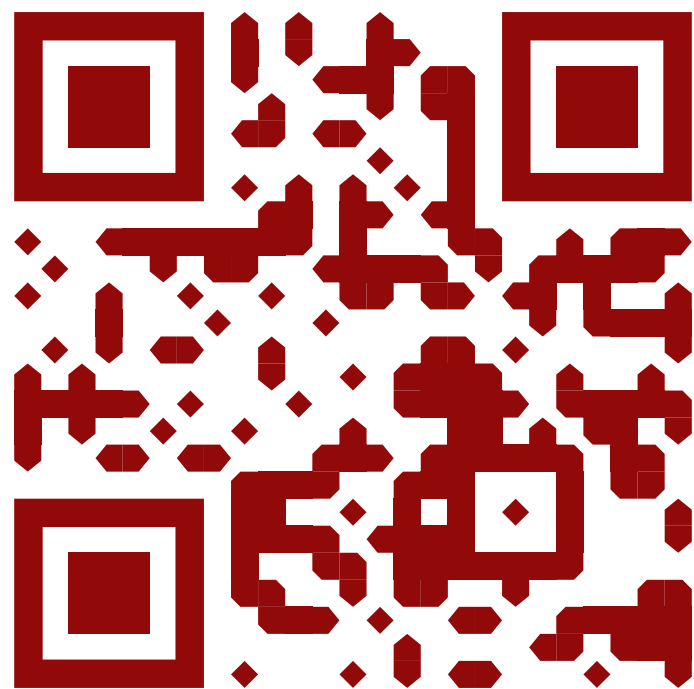
## Trajectories

|                                 | WLAN + BLE | WLAN + BLE + IMU |
|---------------------------------|------------|------------------|
| <i>average error</i>            | 3.28 m     | 1.26 m           |
| <i>maximum error</i>            | 9.60 m     | 3.10 m           |
| <i>average update frequency</i> | 3.04 s     | 0.52 s           |



# Contributions and Conclusions

- ▶ OpenHPS: **open source** framework for hybrid positioning
  - Aimed towards **developers** and **researchers**
- ▶ **Abstractions** such as location-based services and spaces
- ▶ Validation of an indoor positioning use case
- ▶ Configurable and interchangeable **nodes** and **services**
- ▶ **Public dataset** with multiple orientations



Visit <https://openhps.org> for additional resources, documentation, source code and more!