Group 5 SWLD Project: Drone Swarms Control Room

Gabriele Serafino

Università degli Studi di Genova

Abstract. In this project I have built an ontology regarding the management of fleets of drones useful for monitoring and measure various indices such as: pollution, water, temperature, etc. Each drone can carry several sensors useful for the purpose, the payload of each Drone is limited. Each drone takes off from a fixed station (therefore a position). In addition to drones, there are also stations useful for detecting the various indices, these, unlike drones, are fixed and can support an unlimited payload. Protegè was used for the construction of the ontology.

1 Introduction

This project was carried out with the purpose of offering a general view of the Drone Swarms Control Room context, The most important classes are as follows: DRONE (Related to the various drones), Sensor (Class that defines the available sensors), FlottaDRONE (referring to fleets (swarm) of drones) and Mission (which is referring to missions for which the intervention of a fleet of drones is required). Furthermore, the various properties of the classes and the relationships between them have been defined. E.g. The sum of the weights of the sensors mounted on a drone cannot exceed the maximum payload of the drone itself.

2 Classes

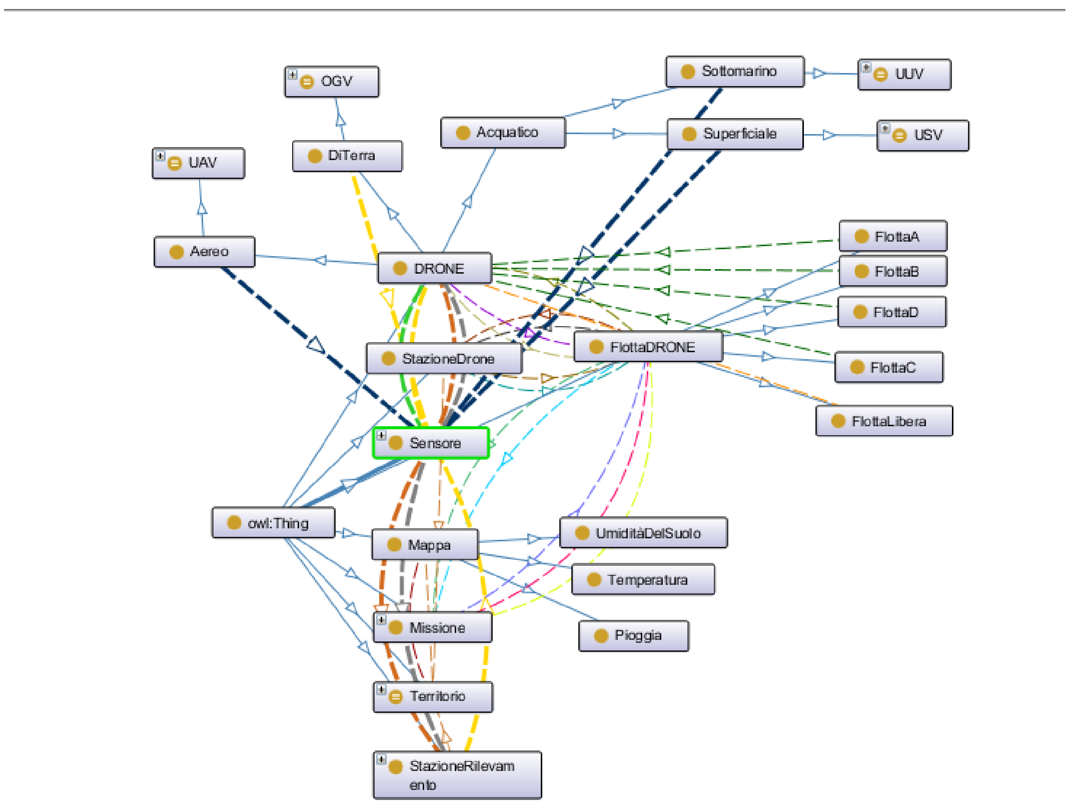


Fig. 1: Ontology overview.

In this representation i have considered this constraints:

- A Drone belongs exactly to 1 Fleet.
- A Drone can have built in several Sensors.
- A Fleet can have 1 or more Drones associated to it, depends on the fleet (A,B,C,D, FlottaLibera).
- The Sensor can be used by only one Drone or StazioneRilevamento
- Each StazioneDrone or StazioneRilevamento or Mappa or Missione belongs only to one Territorio (They all have a specific position)
- A Fleet can have in charge at max 1 Mission and 1 Mission can be assigned to only one Fleet
- Each Fleet depart from only one StazioneDrone

2.1 Drone Class

This class models the various drones they are used for making some measurements in places which are not covered by StazioneRilevamento or places in which the specific StazioneRilevamento has some faults (the places are defined by the missions) , 3 types of drones has been evaluated and for each kind of drones there is one or more sub categories as listed below :

- Aereo.
  - UAV (Unmanned Aerial Vehicle).
- DiTerra.
  - UGV (Unmanned Ground Vehicle).
- Acquatico.
  - Sottomarino.
    - UUV (Unmanned Underwater Vehicle).
  - Superficiale.
    - USV (Unmanned Surface Vehicle).

Each Drone has an ID, a specific maximum payload and a battery duration.

2.2 FlottaDrone class

This class defines the fleets of drones, i considered 5 kind of fleets:

- FlottaA which is composed at max by 3 drones.
- FlottaB which is composed at max by 2 drones.
- FlottaC which is composed at max by 5 drones.
- FlottaD which is composed at max by 4 drones.
- FlottaLibera which is composed by some drones (no limit).

Each fleets depart by a specific StazioneDrone which is fixed in a place.

2.3 StazioneDrone class

This class defines the station from which the fleets departs for reaching the mission.

2.4 Missione class

The missions are the tasks that the drone have to do in specific places, a mission can be related to a place in which there isn't any StazioneRilevamento able to do the measurements, or place in which there is a station but it doesn't work anymore. Each mission belonging to this class has a specific position and a specific typology, the fleets of drones have to reach these places for doing different tasks, for instance, if the typology of the mission is "1" it means that there is the needing of aerial drones so the fleet can have only aerial drones.

2.5 Sensore class

This class defines the sensors which can be mounted on a drone or in a StazioneRilevamento, i had considered these kind of sensors:

- AirborneOpticalPulse(Fulm)Sensor which is for detecting lightning strikes.
- BarometricPressureSensor which is for the pressure.
- DopplerRadarSensor which is used for detecting precipitations.
- HumiditySensor for the humidity.
- SnowLevelSensor for measure the level of the snow.
- SolarRadiationSensor for detecting the solar radiation.
- TermometroSensor for the temperature.
- WaterlevelSensor for measure the level of the water.
- WindSensor for the wind speed (anemometro).
- WebcamSensor which is a webcam.

Each sensor has a weight.

2.6 Mappa class

This class refers to the information about indices on a certain time an a certain place:

- Pioggia. which is a detection of the quantity of rain in terms of water (m3) in a precise time and in a specific place.
- Temperatura. which is a detection of the temperature (C) in a precise time and in a specific place.
- UmiditàDelSuolo. which is a detection of the humidity of the soil (g/m3) in a precise time and in a specific place.

2.7 Territorio Class

Each element belonging this class has a pair of value (Latitude and Longitude).

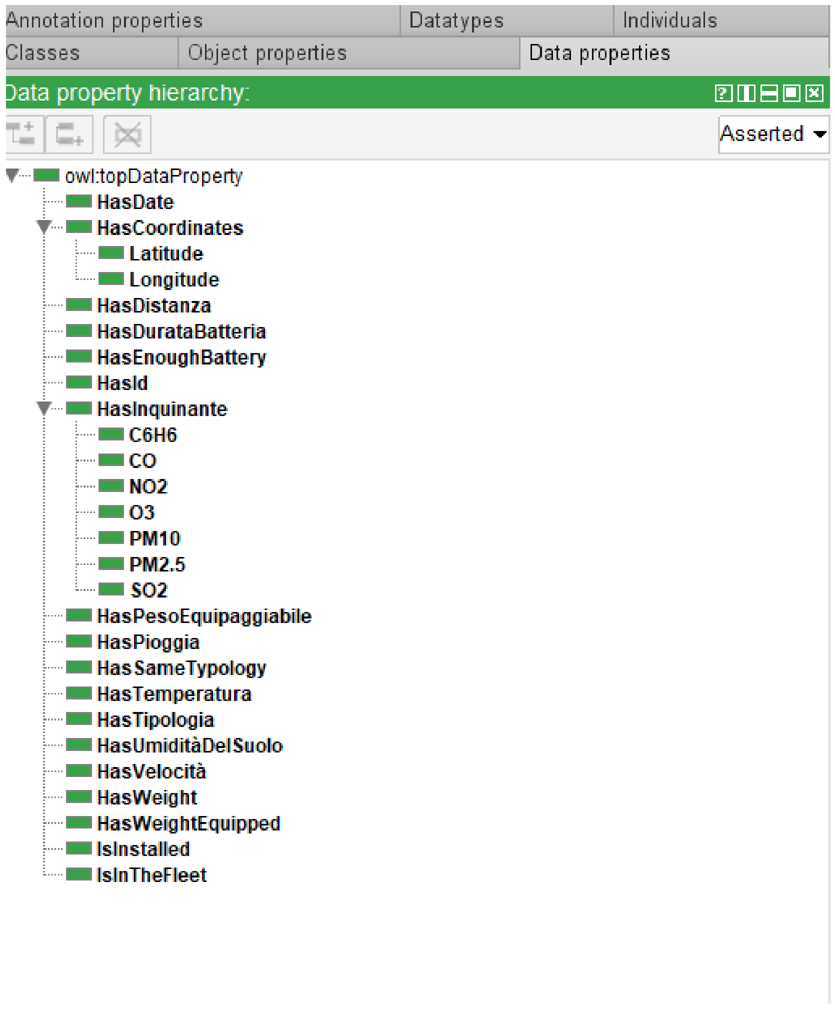
2.8 StazioneRilevamento Class

This class had several subclasses, one for each station listed in the site Omirl.regione.liguria.it. E.g. StazioneBagnaturaFogliare, StazioneFulminazioni, StazioneEliofanie ecc. In this project i had thought that if one of these station as some fault maybe a mission can be detected in the place of the station.

3 Properties

3.1 Data Properties

This section is dedicated to the properties that i had considered for the elements belonging the classes above.



```
owl:topDataProperty
    HasDate
    HasCoordinates
        Latitude
        Longitude
    HasDistanza
    HasDurataBatteria
    HasEnoughBattery
    HasId
    HasInquinante
        C6H6
        CO
        NO2
        O3
        PM10
        PM2.5
        SO2
    HasPesoEquipaggiabile
    HasPioggia
    HasSameTypology
    HasTemperatura
    HasTipologia
    HasUmiditàDelSuolo
    HasVelocità
    HasWeight
    HasWeightEquipped
    IsInstalled
    IsInTheFleet
```

Fig. 2: List of Data Properties

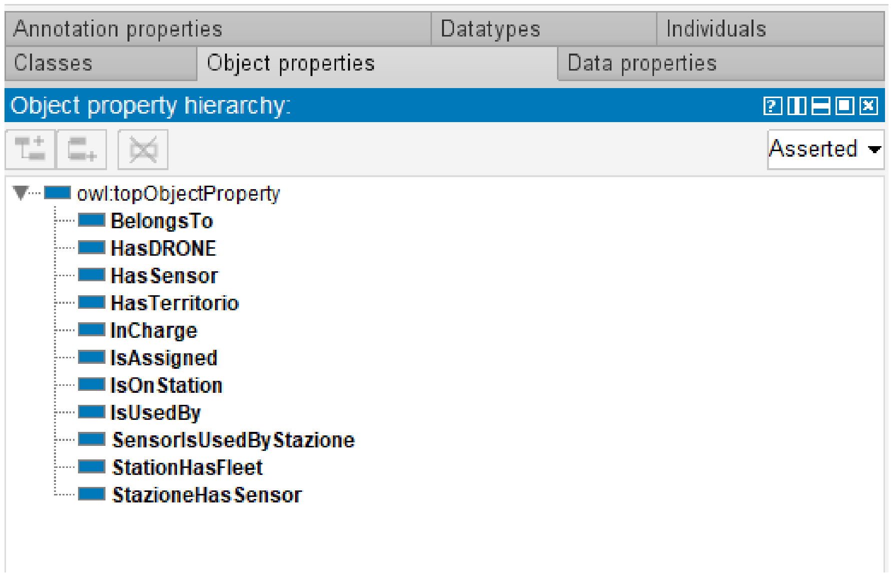
Some explanations about this properties:

- HasDate is an attribute with domain Mappa, so each value has a date.
- HasCordinates is composed by two sub-properties (Longitude and Latitude) both are real numbers.
- HasDistanza is a property related to the distance that a fleet has with respect to a specific mission. (It is assigned once a fleet has a mission in charge) it is expressed in meters(m) and it is a real number.
- HasDurataBatteria is the value (in seconds) of remaining autonomy of a specific drone
- HasEnoughBattery this is a boolean value which states if a drone has enough battery for joining a fleet assigned to a specific mission..
- HasId is the id of drones, fleets, sensor, missions and drone station. This is an integer.
- HasInquinante this property has several sub property each one for a different kind of pollutant (µg/m3) this property belongs to element of rilevamento which is a subclass of StazioneInquinamento which is subclass of StazioneRilevamento.
- HasPesoEquipaggiabile this is the max payload that a drone can carry (kg)
- HasPioggia is the value of the map for the rain (m3).
- HasSameTypology is a flag which is true if a Drone is of the same typology (aerial, aquatic,ground) of the mission assigned to his fleet, false otherwise.
- HasTemperatura is the value of the map for the Temperature (C).
- HasTipologia is an attribute for the missions each mission can have a typology 1 (aerial), typology 2 (ground) typology 3 (aquatic) on the base of these typologies there is a needing of differents fleets of drones for the mission.

– HasUmiditàDelSuolo is the value of the map for the Humidity of the soil (g/m3).
– Hasvelocità is the speed of the drone (m/s) which is for simplicity costant (each drone has his own costant speed).
– HasWeight is the weight of the sensor.
– HasWeightEquipped is the sum of all the the weights of the sensors equipped to a specific drone.
– IsInstalled is a flag which is true if a sensor is installed on a drone false otherwise. (it is installed if his weight added to the cumulative weight on the drone is less or equal to the max payload of the drone).
– IsInTheFleet is another flag that is true if the drone belongs to the fleet for doing a specific mission false otherwise. (it is true if HasEnoughBattery and HasSameTypology are both true).

3.2 Object Properties

The Object properties listed below defines the relationship among the classes.



owl:topObjectProperty
BelongsTo
HasDRONE
HasSensor
HasTerritorio
InCharge
IsAssigned
IsOnStation
IsUsedBy
SensorIsUsedByStazione
StationHasFleet
StazioneHasSensor

Fig. 3: List of Object Properties

– BelongsTo an object belonging Drone have a relationship whit an object which belongs the class FlottaDRONE.
– HasDRONE is the inverse relationship of the one above.
– HasSensor is a relationship between an object of the class Drone and one of the class Sensor.
– HasTerritorio is the relationship of all the object belonging the classes which has a territory and an object of class territory.
– InCharge is the inverse relationship of IsAssigned
– IsAssigned is a relationship between an object of the class Missione and one of the class FlottaDRONE.
– IsOnStation is the relationship between an object of FlottaDRONE and an object of StazioneDrone.
– IsUsedBy is the inverse relationship of HasDRONE.
– SensoreIsUsedByStazione It is the inverse relationship of StazioneHasSensor.
– StationHasFleet is the inverse relationship of IsOnStation.
– StazioneHasSensor an object belonging StazioneRilevamenti has a relationship whit an object which belongs the class Sensore.

4 Constraints and SWRL Rules

I used SWRL Rules in order to define constraints regarding The relationships among the missions, the fleets, the drones, and the sensors.

i defined 7 rules:

1. CumulativeWeightRule: the sum of the weights of all the sensors cannot be higher then the payload of the drone. If a weight of a sensor added to the previous sensors' weights installed ("HasWeightEquipped") is higher than the max payload the data-property "IsInstalled" remains false. Otherwise the property become true and the weight of the sensor will be added to the value within "HasWeightEquipped" property.
2. FleetMissionDistanceRule: if a fleet is assigned to a mission it will be evaluated the distance between the StazioneDrone in which we have the fleet and the location of the mission.
3. DroneBatteryRule: if the autonomy of the battery (of a drone belonging to a fleet which is assigned to a specific mission) is enough for doing two times the distance evaluated with "FleetMissionDistanceRule" considering the speed of the drone the data-property "HasEnoughBattery" will be set to "true".
4. MissionTypologyAerialRule: if the drone belongs to the sub class "Aereo" and the typology of the mission is "1" the property HasSameTypology will be set on "true".
5. MissionTypologyAquaticRule: if the drone belongs to the sub class "DiTerra" and the typology of the mission is "2" the property "HasSameTypology" will be set on "true".
6. MissionTypologyAerialRule: if the drone belongs to the sub class "Acquatico" and the typology of the mission is "3" the property HasSameTypology will be set on "true".
7. DroneJoinMissionRule: if the property "HasEnoughBattery" and the property "HasSameTypology" are both true, the property "IsInTheFleet" will be set on "true".

References