DISIT SiiMobility Ontology

In input to Sii-Mobility Project http://www.disit.dinfo.unifi.it/siimobility.html

Knowledge base accessible as sparql entry point as shown from http://log.disit.org

http://www.disit.dinfo.unifi.it

info from info@disit.org

version 0.1, draft

date 24-01-2013

referent coordinator: paolo.nesi@unifi.it

Aim

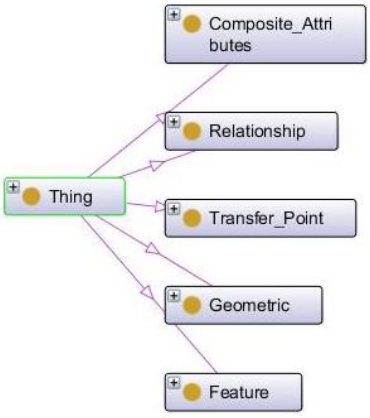
The development of a unified and integrated ontology for smart city including transport, infomobility and large set of other open data.

Short State of the Art

To realize the ontology that will be used within the SiiMobility project, was carried out a state of the art study on ontology related to Smart Cities.

The only ontology presented as a "help to transform cities into Smart Cities" is SCRIBE, made by IBM, on which not much information is available online free of charge [http://researcher.watson.ibm.com/researcher/view_project.php?id=2505].

Among other ontologies that may be related to Smart Cities, we mention the OTN, Ontology of Transportation Networks.



This ontology defines the entire transport system in all its parts, from the single road / rail, to the type of maneuvering that can be performed on a segment of road or public transport routes. As you can see from the figure, the OTN includes the concepts expressed in the 5 main macro classes, attributes composites (where there are classes like TimeTable, Accident, House_Number_Range, Validity_Period, Maximum_Height_Allowed), relationships (in which we find the Manoeuvre), transfer points (macroclass which includes classes such as Road, Road_Element, Building, and others), geometry (ie classes Edge and Face Node), and features (which contains classes such as Railways, Service, Road_and_Ferry_Feature, Public_Transport).

On the net there are also many other ontologies related to sensor networks, such as the SemanticSensorNetwork Ontology, which provides elements for the description of sensors and their observations [http://www.w3.org/2005/Incubator/ssn/ssnx/ssn] and FIPA Ontology which is more focused on the description of the devices and their properties both HW and SW

[http://www.fipa.org/specs/fipa00091/PC00091A.html#_Toc511707116]. We therefore believe these ontologies are to be taken into account when we will have data applicable to these areas.

Analyzing data made available by the PA (mainly by the Tuscany Region) it was found that, in relation to the roads graph, data could be easily mapped into large parts of the OTN, concerning to Street Guide; taking into account these observed similarities, we think it could be useful, in order to associate a more precise semantics to the various entities of our ontology, make explicit reference to the OTN, to make the concepts clearer and more easily linkable to other, wanting to follow guidelines for the implementation of Linked Open Data (http://www.w3.org/DesignIssues/LinkedData.html).

Following this principle, it was therefore made, partially for now, what will become the ontology inside the SiiMobility project.

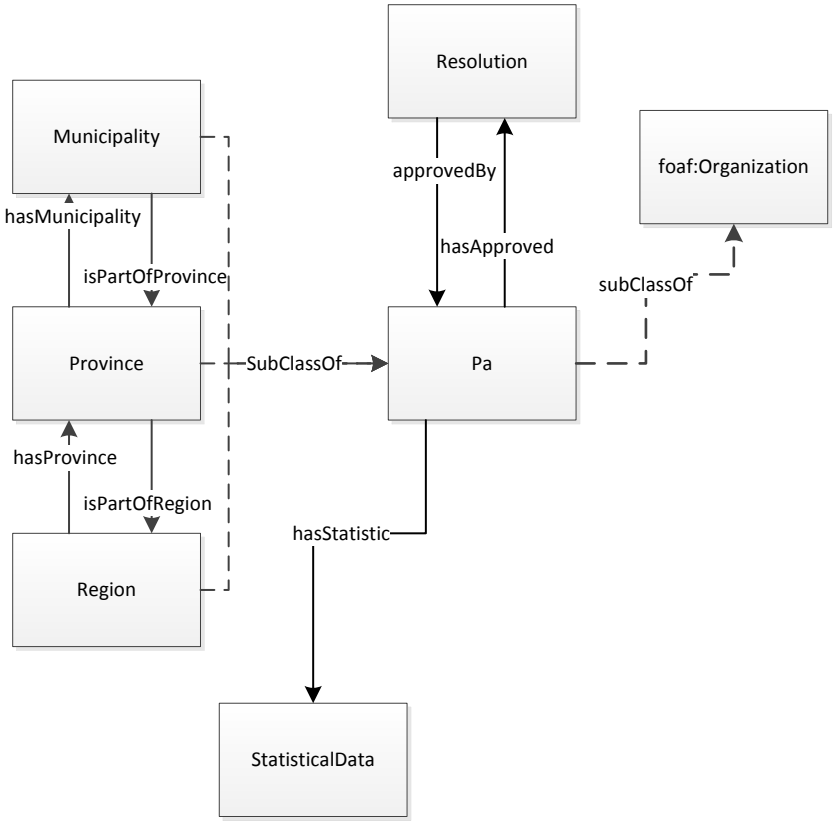
The DISIT Ontology

The SiiMobility project aims at enabling interconnection, storage and the next interrogation of data from many different sources, such as various portals of the Tuscan region (MIIC, Muoversi in Toscana, Osservatorio dei Trasporti), Open Data provoded by individual municipalities (mainly Florence). It is therefore evident that the ontology will be built, will not be small, and so it may be helpful to view it as consisting of various macro classes, and to be precise, at present, the following macro-categories have been identified:

1. Administration: the first macroclass that is possible to discover, whose main classes are PA, Municipality, Province, Region, Resolution.
2. Street Guide: formed by classes like Road, Node, RoadElement, AdminidtrativeRoad, Milestone, StreetNumber, RoadLink, Junction, Entry, EntryRule and Maneuver.
3. Points of Interest: includes all services, activities, which may be useful to the citizen, and that may have the need to reach. The classification of individual services and activities will be based on classification previously adopted by the Tuscany Region.
4. Local Public Transport: currently we have access to data relating to scheduled times of the leading LPT, the graph rail, and real-time data relating to ATAF services. This macroclass is then formed by many classes like TPLLine, Ride, Route, AVMRecord, RouteSection, BusStopForeast, Lot, BusStop, RouteLink, TPLJunction.
5. Sensors: the macroclass relative to data coming from sensors is developing. Currently in the ontology have been integrated data collected by various sensors installed along some roads of Florence and in that neighbourhood, and those relating to free places in the major parks of the whole region; in our ontology is already present the part relating to events/emergencies, where, however, the collected data are currently very limited in number plus several months old. In addition to these data, in this macroclass were included also data related to Lamma's weather forecast.
6. Temporal: macroclass pointing to include concepts related to time (time instants and time intervals) in the ontology, so that you can associate a timeline to the recorded events and can be able to make predictions.

Let us now analyze one by one the different macro classes identified.

The first macroclass, Administration is composed as shown in the following figure.



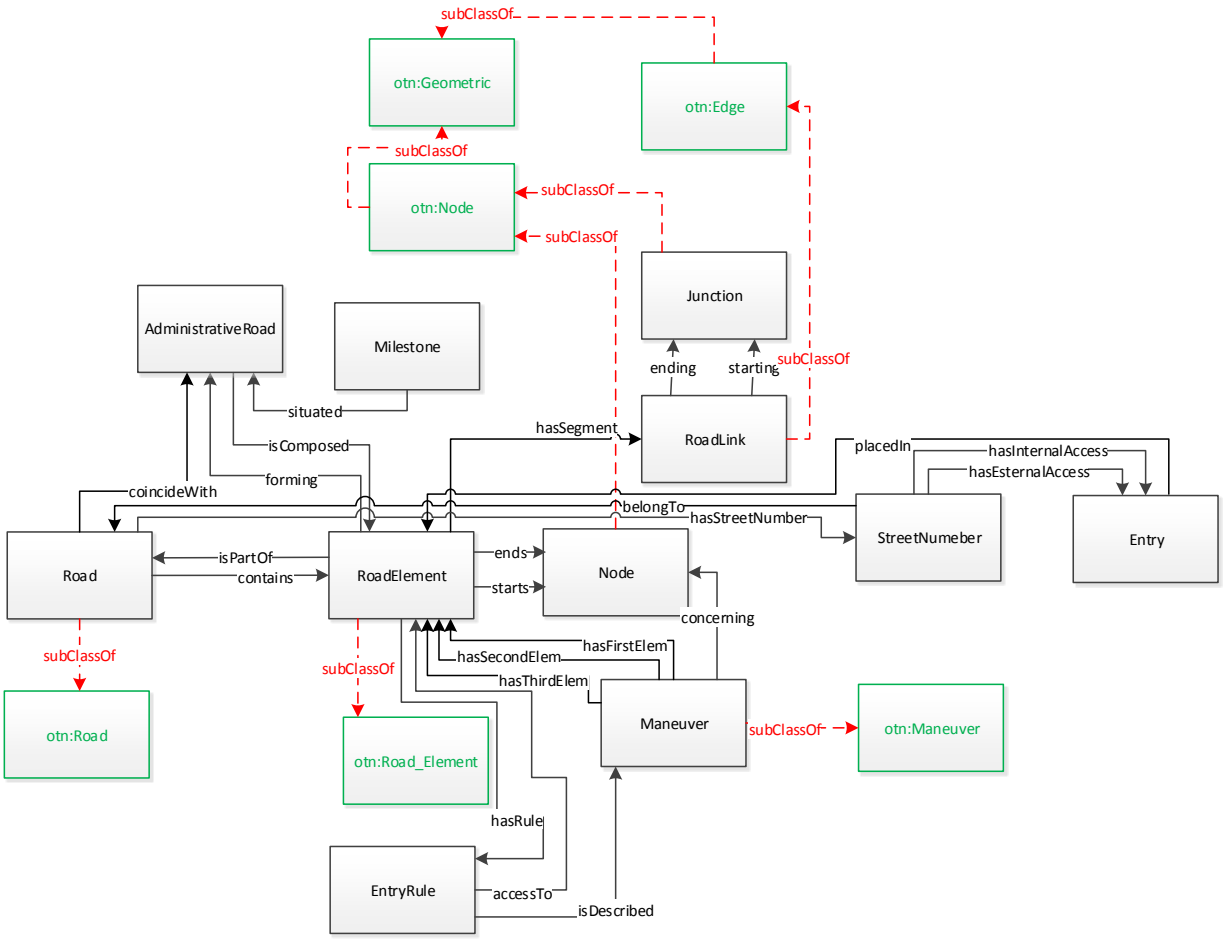
The main class is PA, which has been defined as a subclass of foaf: Organization, link that helps us to assign a clear meaning to our class. The 3 subclasses of PA are automatically defined according to the restriction on ObjectProperties (represented in the figure by solid lines). For example, the Region class is defined as a restriction of the class PA on ObjectProperty "hasProvince", so that only the PA that possess provinces, can be classified as a region. Another example: to define the PA elements that make up the Municipality class was instead used a restriction on ObjectProperty "isPartOfProvince," so if a PA is not assigned to a province, it cannot be considered a municipality.

During the establishment of the hierarchy within the class PA, for each step were defined pairs of inverse ObjectProperties: "hasProvince" and "isPartOfRegion", "hasMunicipality" and "isPartOfProvince."

Connected to the class PA through the ObjectProperty "hasApproved", we can find the Resolution class, whose instances are represented by the resolutions passed by the various PA note. "hasApproved" ObjectProperty has its inverse, that is, "approvedBy."

The last class in this macroclass is StatisticalData: given the large amount of statistical data related both to the various municipalities in the region, but also to each street, that class is shared by both Administration and Street Guide macroclass. As we will see in the next macroclass description, the class StatisticalData is connected to both Pa at Road through the ObjectProperty "hasStatistic."

The macroclass Street Guide is instead shown in the following figure.



The main class, in the middle of Street Guide macroclass, is RoadElement, which is defined as a subclass of the corresponding element in the ontology OTN, ie Road_Element. Each RoadElement is delimited by a start node and an end node, detectable by the ObjectProperties "starts" and "ends", which connect the class object of the class Node. Some restrictions have been specified in the RoadElement class definition, related to the Node class: a road element must have both "starts" and "ends" ObjectProperty, both with a cardinality exactly equal to 1. One or more road elements forming a road: the class Road is in fact defined as a subclass of the corresponding class in the OTN, ie the homonymous Road, and with a cardinality restriction on "contains" ObjectProperty, which must be a minimum equal to 1, ie cannot exist a road that does not contain at least one road element. Also the class AdministrativeRoad, which represents the administrative division of the roads, is connected to the class RoadElement through two inverse ObjectProperty "isComposed" and "forming", while it is connected with only one ObjectProperty to the Road class ("coincideWith"). Better clarify the relationship that exists between Road, AdministrativeRoad and RoadElement: a Road's instance can be connected to multiple instances of AdministrativeRoad (eg if a road crosses the border between the two provinces), but the opposite is also true (eg when a road crosses a provincial town center and assumes different names), ie there is a N: M relationship between the two classes. On each road element is possible to define access restrictions identified by the class EntryRule, which is connected to the class RoadElement through 2 inverse ObjectProperties, ie "hasRule" and "accessTo". The EntryRule class is defined with a restriction on the minimum cardinality of ObjectProperty "accessTo" (set equal to 0), which in most cases only one element has an associated road, but in some exceptional cases, there is no association. Access rules allow to define uniquely a permit or limitation access, both on road elements (for example due to the presence of a ZTL) as just seen, but also on maneuvers; for this reason, the class maneouvre and the class EntryRule are connected by "isDescribed"

ObjectProperty. The term maneuver refers primarily to mandatory turning maneuvers, priority or forbidden, which are described by indicating the order of road elements involving. By analyzing the data from the Roads graph has been verified that only in rare cases maneuvers involving 3 different road elements and then to represent the relationship between the classes maneouvre and RoadElement, were defined 3 ObjectProperties: "hasFirstElem", "hasSecondElem" and "hasThirdElem", in addition to the ObjectProperty that binds a maneuver to the junction that is interested, that is, "Concerning" (because a maneuver takes place always in the proximity of a node). In the maneouvre class definition there are cardinality restrictions, set equal to 1 for "hasFirstElem" and "hasSecondElem" and set maximum cardinality to 1 for "hasThirdElem", as for the maneuvers that affect only 2 road elements, this last ObjectProperty is not defined.

As previously mentioned, each road element is delimited by two nodes (or junctions), the starting one and the ending one. It was then defined Node class, subclass of the same name belonging to ontology OTN Node class. The Node class has been defined with a restriction on DataProperty geo: lat and geo: long, two properties inherited from the definition of subclass of geo: SpatialThing belonging to ontology Geo wgs84: in fact, each node can be associated with only one pair of coordinates in space, and cannot exist a node without these values.

The Milestone class represents the kilometer stones that are placed along the administrative roads, that is, the elements that identify the precise value of the mileage at that point, the advanced of the route from the starting point. A Milestone may be associated with a single AdministrativeRoad, and is therefore defined a cardinality restriction equal to 1, associated with ObjectProperty "placedIn". Also the Milestone class is defined as subclass of geo:SpatialThing, but this time the presence of coordinates is not mandatory (restriction on maximum cardinality must be equal to one, but that does not exclude the possible lack of value).

The street number is used to define an address, and it is always logically related to at least one access, in fact every street number always corresponds to a single external access, which can be direct or indirect; sometimes it can also be a internal access. Looking at this relationship from the access point of view, you can instead say that each of these is logically connected to at least one street number. Were then defined StreetNumber and Entry classes.

With the data owned is possible to connect the class StreetNumber to the class RoadElement and to the class Road, respectively through the ObjectProperties "standsIn" and "belongTo." This information is actually redundant and if deemed appropriate, may be delete the link connect to the class Road, in favor of the one towards RoadElement, more easily obtainable from data; it is for this reason that for the ObjectProperty "standsIn" has been defined also the inverse "hasStreetNumber".

Within the ontology therefore, the StreetNumber class is defined with a cardinality restriction on the ObjectProperty "standsIn", which must be equal to 1.

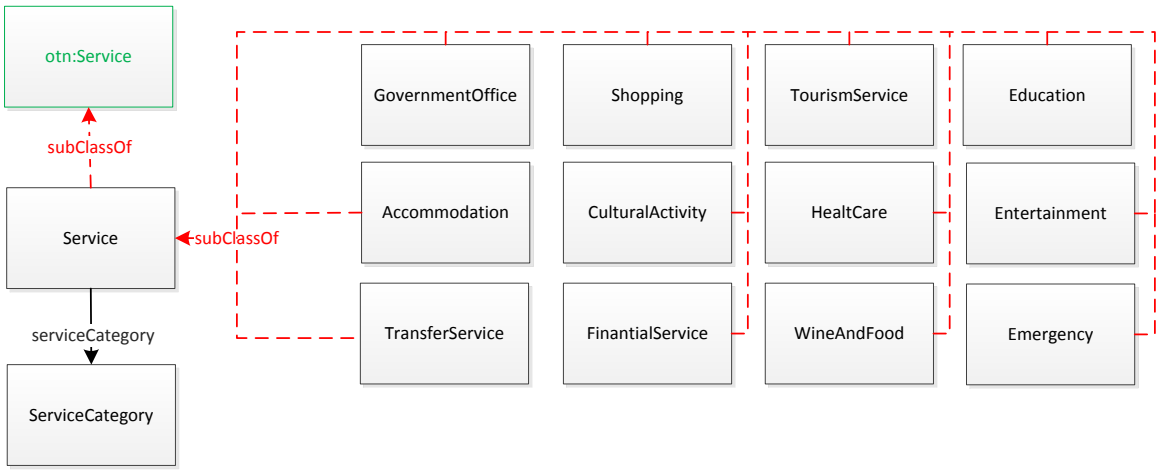
Entry class also can be connected to both the street number and road element where is located. The relationship between Entry and StreetNumber, is defined by two ObjectProperties, "hasInternalAccess" and "hasExternalAccess", on which have been defined cardinality restrictions, since, as mentioned earlier, a street number will always have only one external access, but could also have an internal access (the latter restriction it is in fact defined by setting the maximum cardinality equal to 1, ie they are allowed values 0 and 1). Also the Entry class is defined as a subclass of geo: SpatialThing, and is possible to associate a

maximum of one pair of coordinates geo:lat and geo:long to each instance (restriction on the maximum cardinality of the two DataProperty of the Geo ontology, set to 1, so which may take 1 value , or none).

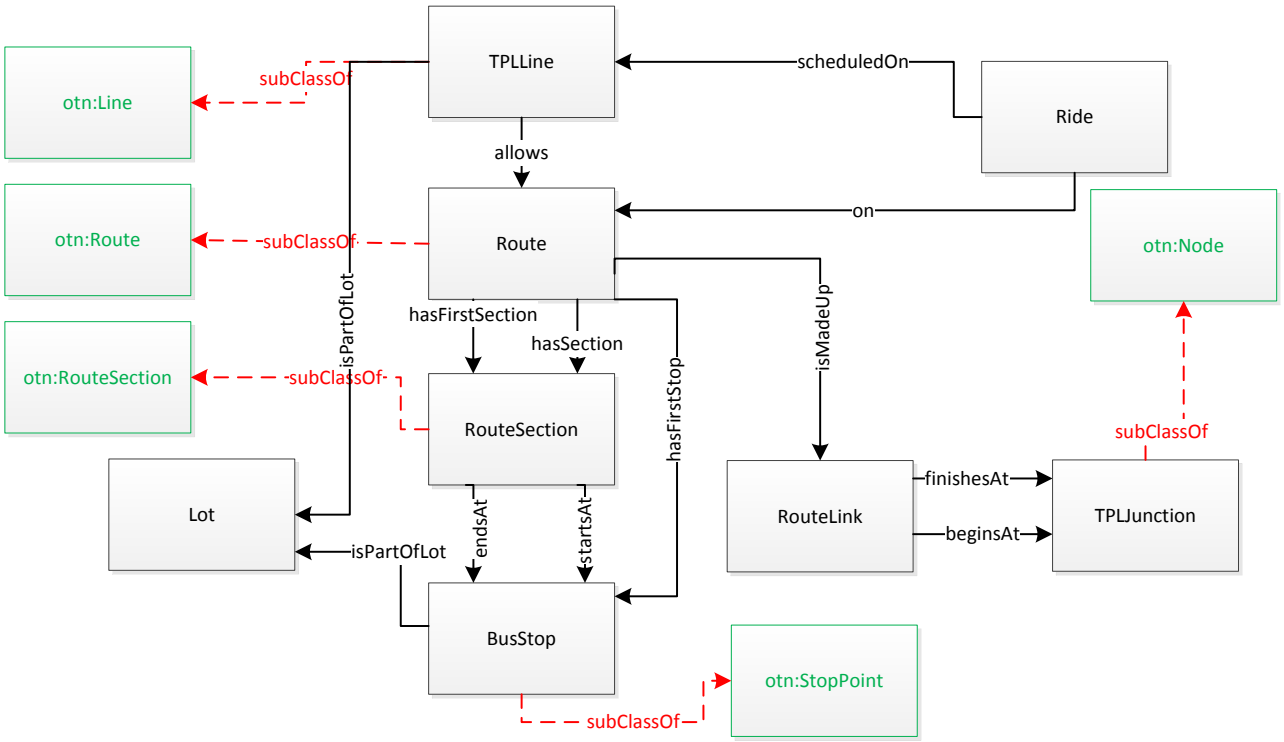
The Street macroclass is connected to 2 different Administration through the ObjectProperties "OwnerAuthority" and "managingAuthority", which as the name suggests, clearly represent respectively the public administration which has extensive administrative, or public administration that manages the road element . This leaves out the representation, only the streets of private property, for which we have not yet identified the best representation in the ontology .

From a cartographic point of view, however, each road element is not a straight line, but a broken line, which will follow the actual course of the road. To represent this situation, the classes RoadLink and Junction have been added: thanks to the interpretation of the KMZ file, we retrieved the set of coordinates that define each RoadElement, and each of these points will be added to the ontology as an instance of Junction (defined as a subclass of geo: SpatialThing, with compulsory single pair of coordinates) . Each small segment between two Junction is instead an instance of the class RoadLink, which is defined by a restriction on the ObjectProperty "ending" and "starting" (both are obliged to have cardinality equal to 1), which connect the two classes.

For the third macro class, that is, points of interest, have defined a generic class Service, which is associated with the NACE code , ie the code ISTAT classification of economic activities, which could be used in future as a filter to define the various subclasses, in place of the division into the categories defined by the Region of Tuscany, in order to make more precise research of the various types of services. Unfortunately, however, due to the high percentage of codiciAteco missing , the current division of services into subclasses is based on value dell'objectProperty serviceCategory : that properties associated with each service an individual belonging to the class ServiceCategory (formed by exactly the individuals that have been found on the data of Tuscany). All areas of interest have not yet been defined, since we have not yet provided a list of all services that will be included, but for now in fact , relying on a small list of POI recovered from the site of the MIIC, the following classes have been identified: Accommodation, GovernmentOffice, TourismService, TransferService, CulturalActivity, FinantialService, Shopping, healtcare, Education, Entertainment, Emergency and WineAndFood. The Accommodation class for example, was defined as a restriction of the Service class on the ObjectProperty serviceCategory, that must take one of the following values: villaggio_vacanze, albergo_hotel, casa_per_vacanze, casa_di_riposo, casa_per_ferie, bed_and_breakfast, hostel, residenza_turistica_alberghiera, farmhouse, residence_di_villeggiatura, centri_accoglienza_e_case_alloggio, camping, residenze_epoca, rifugio_alpino. Similarly, other classes have been defined, including among the possible values of the ObjectProperty, those corresponding to that type of service.



The fourth macro class TPL is not complete yet, it only presents data and classes relating to metropolitan bus lines of Florence and tramway, we assume that this distribution can easily adapt also to the suburban lines of the bus, but we were able to check this statement only in presence of official data.



Each TPL lot, represented by Lot class, is composed of a number of bus or tram lines (class TPLLine), and this relationship is represented by the ObjectProperty "isPartOfLot", which connects each instance of TPLLine to the corresponding instance of Lot. The TPLLine class is defined as a subclass of otn:Line. Each line includes at least one race, identified through a code provided by the TPL company; the TPLLine class was in fact connected to the class Ride through the ObjectProperty "scheduledOn", on which is also defined as a limitation of cardinality exactly equal to 1, because each stroke may be associated to a single line.

Each race follows at least one path (the 1:1 match has not been tested on all data, as soon as it occurred, eventually will include a restriction on the cardinality of the ObjectProperty that connects the two classes, namely "On"), and the paths can be in a variable number even if referring to a same line: in most cases are

2, ie the forward path and backward path , but sometimes also come to be 3 or 4, according to possible extensions of paths deviations or maybe performed only in specific times.

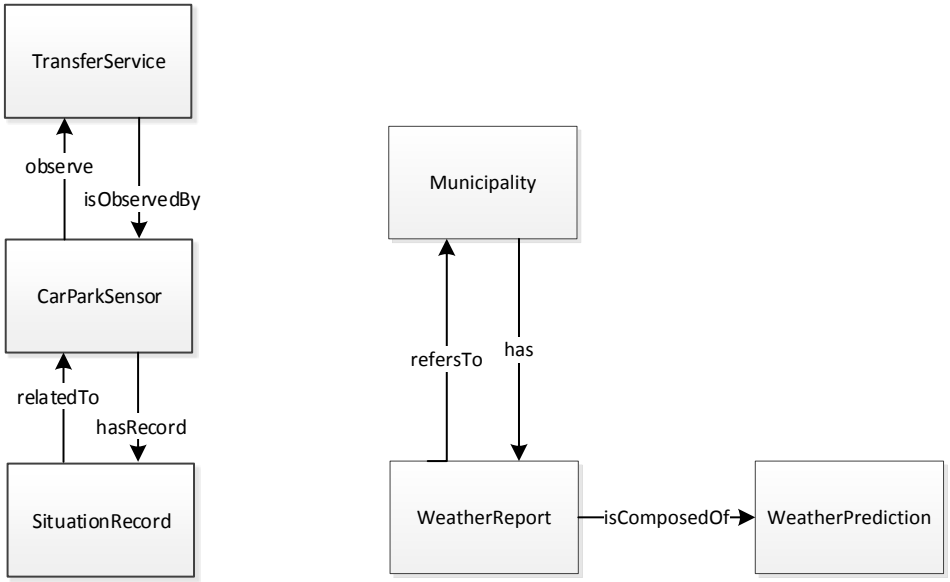
Each path is considered as consisting of a series of road segments delimited by subsequent stops : to model this situation, it was decided to define two ObjectProperty linking Route and RouteSection classes, "hasFirstSection" and "hasSection", since, from a cartographic point of view, wanting to represent the path that follows a certain bus; knowning the first segment and the stop of departure, you can obtain all the other segments that make up the complete path and starting from the second bus stop, identified as a different stop from the first stop, but that it is also contained in the first segmen, we are able to reconstruct the exact sequence of the stops, and then the segments, which constitute the entire path . For this purpose has been defined also the ObjectProperty "hasFirstStop", which connects the classes Route and BusStop .

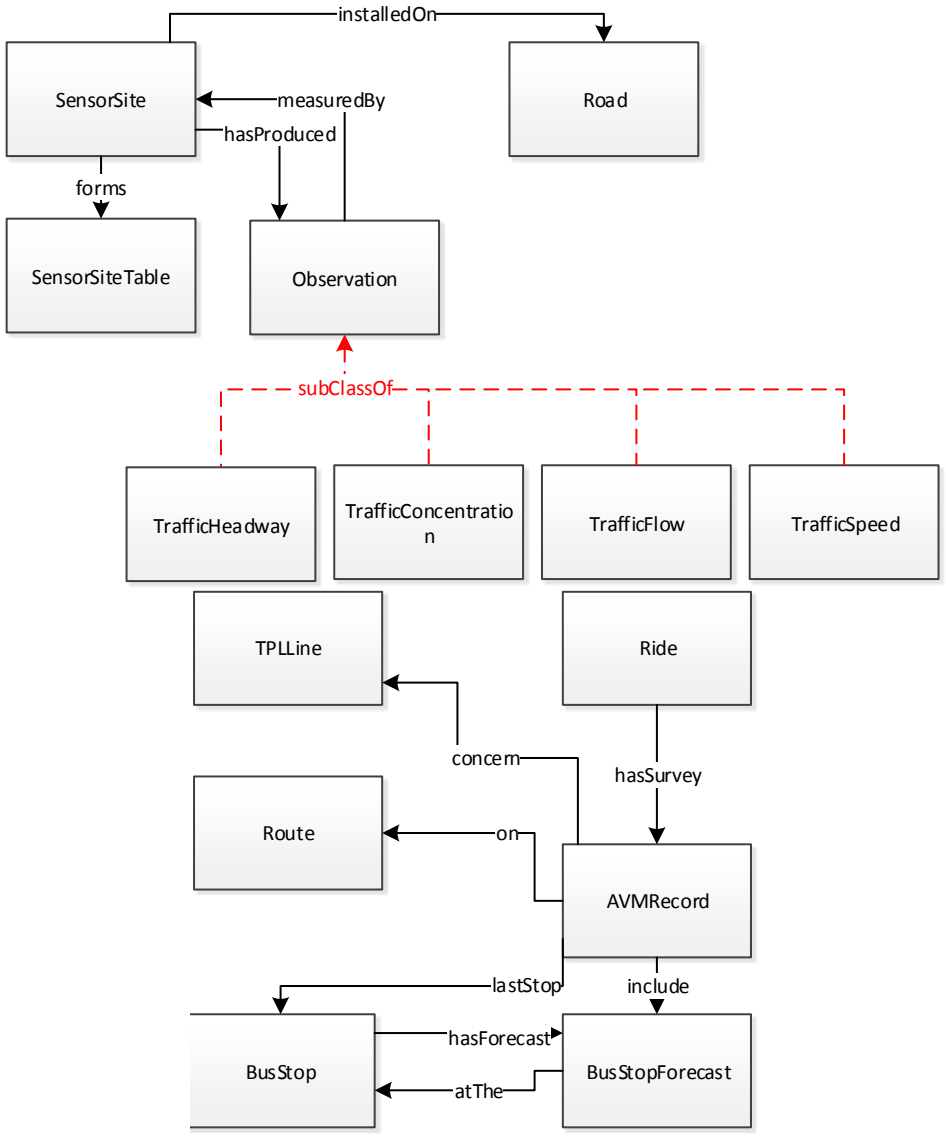
Applying the same type of modeling used for road elements, two ObjectProperty have been defined, "endsAt" and "startsAt", to connect each instance of RouteSection to two instances of the class BusStop, class in turn defined as a subclass of OTN: StopPoint. Each stop is also connected to the class Lot, through the ObjectProperty "isPartOfLot", with a 1:N relation because there are stops shared by urban and suburban lines so they belong to two different lots.

Possessing also the coordinates of each stop, the class BusStop was defined as a subclass of geo:SpatialThing, and was also termed a cardinality equal to 1 for the two DataProperty geo:lat and geo:long.

Wishing then to represent to a cartographic point of view the path of a bus, ie a Route instance, we need to represent the broken line that composes each stretch of road crossed by the means of transport itself and to do so, the previously used modeling has been reused to the road elements: we can see each path as a set of small segments, each of which delimited by two junctions: were then defined RouteLink and TPLJunction classes, and the ObjectProperty "beginsAt" and "finischesAt". The RouteLink class was defined as a cardinality restriction of both just mentioned ObjectProperty, imposing that it is always equal to 1 . The Route class is instead connected to the RouteLink class through "isMadeUp" ObjectProperty.

Sensors Macro class has not yet been completed, but for now it consists of parts shown in the figure, and respectively relating to the car parks sensors, to the weather sensors, to the sensors installed along roads and rails and to the AVM systems.


TransferService
observe
isObservedBy
CarParkSensor
relatedTo
hasRecord
SituationRecord
Municipality
refersTo
has
WeatherReport
isComposedOf
WeatherPrediction


installedOn
SensorSite
measuredBy
hasProduced
Road
forms
SensorSiteTable
Observation
subClassOf
TrafficHeadway
TrafficConcentratio
n
TrafficFlow
TrafficSpeed
TPLLine
Ride
concern
hasSurvey
Route
on
AVMRecord
lastStop
include
BusStop
hasForecast
atThe
BusStopForecast

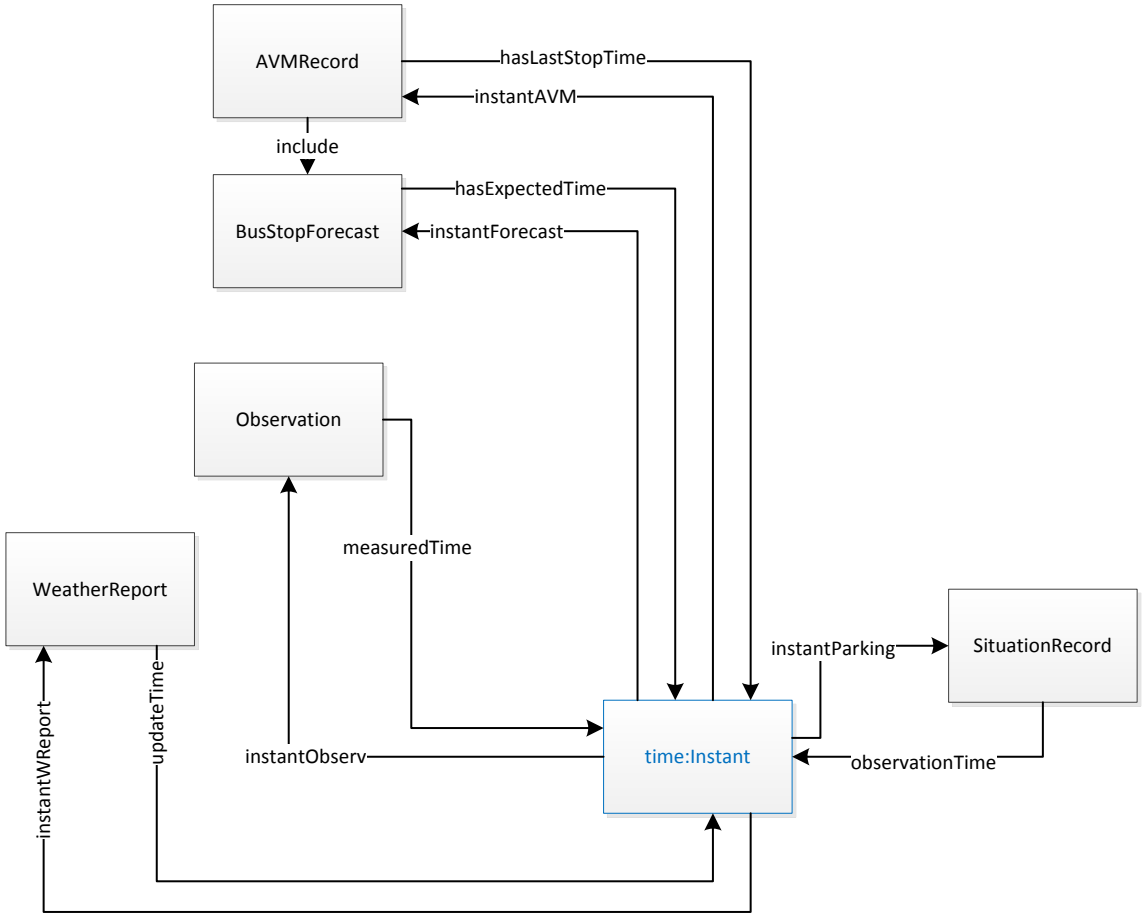
The first part shown in the figure is focused on the real-time data related to parking. The TransferService class, in fact, is connected to the CarParkSensor class, which represents the sensor installed in a given parking and which will be linked to instances of the SituationRecord class, which represent the state of a certain parking at a certain instant; the first link, ie the one between Transferservice class and CarParkSensor class, is realized through two invrse ObjectProperty, "observe" and "isObservedBy", while the connection between CarParkSensore class and SituationRecord class, is performed via the reverse ObjectProperty, "relatedto" and "hasRecord". The class SituationRecord allows to store information about the number of free and occupied parking spaces, in a given moment (also recorded) for the main car parks in Tuscany region.

The second part of the received data in real time, concerns the weather forecast, available for different areas (and thus connected to the class Municipality), thanks to LAMMA. This consortium will update each municipality report 1 or 2 times a day and every report contains forecast of 5 days divided into range, which have a greater precision (and a higher number) for the nearest days until you get to a single daily forecast for the 4th and 5th day. This situation is in fact represented by the WeatherReport class connected to the WeatherPrediction class via the ObjectProperty "isComposedOf". Municipality class is instead connected to a report by 2 reverse ObjectProperty: "RefersTo" and "has".

The third part of the Real Time data concerns the sensors placed along the roads of the region, which allow to make different detection related to traffic situation. Unfortunately, the location of these sensors is not very precise, it is not possible to place them in a unique point thanks to coordinate, but only to place them within a toponym, which for long-distance roads such as FI- PI-LI road, it represent a range of many miles. Sensors are divided into groups, each group is represented by SensorSiteTable class and each instance of the class SensorSite (that represent a single sensor) is connects to its group through the ObjectProperty "forms" and, as mentioned earlier, each instance of SensorSite class can be connected only to the Road class (through the ObjectProperty "installedOn"). Each sensor produces observations, which are represented by instance of Observation class and these observations can belong to 4 types, ie they can be related to the average velocity (TrafficSpeed subclass), or related to the car flow passing in front of the sensor (TrafficFlow subclass), related to traffic concentration (TrafficConcentration subclass), and finally related to the traffic density (TrafficHeadway subclass). The classes Observation and Sensor are connected via a pair of reverse ObjectProeprty, "hasproduced" and "measuredBy".

The fourth part of RealTime macroclass concerns the AVM systems installed on most of ATAF busses, and it is mainly represented by two classes, AVMRecord and BusStopForecast: the first class mentioned represents a record sent by the AVM system, in which, as well as information on the last stop done (represented by the "lastStop" ObjectProperty that connects the classes AVMrecord to BusStop), GPS coordinates of the vehicle position, and the identifiers of vehicle and line, we also find a list of upcoming stops with the planned passage time; this list have a variable length and it represents instances of the BusStopForecast class . This latter class is linked to the class BusStop through "atThe" ObjectProperty so as to be able to recover the list of possible lines provided on a certain stop (AVMRecord class is in fact also connected to Line class via "concern" ObjectProperty).

Finally, the last macroclass, called Temporal Macro class, is now only "sketchy" within the ontology, and it is based on the Time ontology (http://www.w3.org/TR/owl-time/) but also on experience gained in other projects such as OSIM. It requires the integration of the concept of time as it will be of paramount importance to be able to calculate differences between time instants, and the Time ontology comes to help us in this task.



We define fictitious URI #instantForecast, #instantAVM, #instantParking, #instantWreport, #instantObserv to following associate them to the identifier URI of a resource referred to the time parameter, ie respectively BusStopForecast, AVMRecord, SituationRecord, WheatherReport and finally Observation.

The fictitious URI #instantXXXX, will be formed as concatenation of two strings: for example, in the case of BusStopForecast instances, it will be concatenate the stop code string (which allows us to uniquely identify them) and the time instant in the most appropriate format. Is necessary to create a fictitious URI that links a time instant to each resource, to not create ambiguity, because identical time instants associated with different resources may be present (although the format in which a time istant is expressed has a fine scale) .

Time Ontology is used to define precise moments as temporal information, and to use them as extreme for intervals and durations definition, a feature very useful to increase expressiveness.

Pairs of ObjectProperties have also been defined for each class that needs to be connected to the class Instant: between classes Instant and SituationRecord were defined the inverse ObjectProperties “instantParking” and “observationTime”, between classes WeatherReport and Instant, “instantWReport” and “updateTime” ObjectProperties have been defined, between classes Observation and Time there are the reverse ObjectProperties “measuredTime” and “instantObserv”, between BusStopForecast and Time we can find “hasExpectedTime” and “instantForecast” ObjectProperties, and finally, between AVMRecord and Time, there are the reverse ObjectProperties “hasLastStopTime” and “instantAVM”.

The domain of all ObjectProperties with instantXXXX name is defined by elements Time:temporalEntity, so as to be able to expand the defined properties not only to time instant, but also to time intervals.

DataProperties of main classes

Inside the ontology a lot of DataProperties, for which there was the certainty of not being able to use the equivalent values defined on other ontologies, have been defined.

We analyze below, for the main classes in which they were defined, these properties.

Within the class PA were defined only 2 DataProperties, foaf name that represents the name of the public administration represented by the considered instance, and its unique identifier present in the regional system, dtc:Identified, from DublinCore ontology. More information about PA can be found through the link with the Service class, where in fact there are more details that would otherwise be redundant. The Resolution class, whose instance as seen above are the municipality resolutions, has some DataProperties for the definition of an Id, dtc:Identified, the year of approval, SiiMobility:year, and some other property that is coming from the ontology DublinCore dct:subject (the resolution object), dct:created (the date on which the PA has resolved) and foaf:page that represents the URL to which the resolution is available online.

Each instance of the Route class is uniquely identified using the DataProperty dtc:Identified where it is stored the toponym identifier for the entire regional network, consisting of 15 characters and defined according to the following rule: RT followed by ISTAT code of the municipality to which the toponym belongs (6 characters), followed by 5 characters representing the sequential from the character value of the ISTAT code, and finally the letters TO. The roadType instead, represents the type of toponym, for example:

- Chiasso, Corso, Largo, Località, Piazza, Piazzale, Piazzetta, Via, Viale, Vicolo, Viottolo, Viuzzo etc.