



Ricerca di Sistema elettrico

Smart City Platform Specification Semantic Level 1.0

A. Brutti, A. Frascella, N. Gessa

SMART CITY PLATFORM SPECIFICATION SEMANTIC LEVEL 1.0

A. Brutti, A. Frascella, N. Gessa (ENEA)

Settembre 2018

Report Ricerca di Sistema Elettrico

Accordo di Programma Ministero dello Sviluppo Economico - ENEA

Piano Annuale di Realizzazione 2017

Area: Efficienza energetica e risparmio di energia negli usi finali elettrici e interazione con altri vettori energetici

Progetto: Sviluppo di un modello integrato di smart district urbano

Obiettivo: Piattaforma ICT per la gestione dello Smart District

Responsabile del Progetto: Claudia Meloni, ENEA

Indice

INTRODUZIONE.....	4
SEMANTIC LEVEL	5
1 DESCRIZIONE DELL'ONTOLOGIA E MODELLO SEMANTICO	6
2 REGOLE DI NAMING PER I CONCETTI DELL'ONTOLOGIA	16
3 CONSULTAZIONE DELL'ONTOLOGIA.....	18
4 INTERROGAZIONE DELL'ONTOLOGIA	21
5 INSERIMENTO DI NUOVI URBAN DATASET	25
5.1 IL COMITATO DI GESTIONE DELL'ONTOLOGIA	25
5.2 PROCEDURA PER LA RICHIESTA D'INSERIMENTO.....	25
5.3 MODULO PER LA RICHIESTA DI INSERIMENTO.....	26

Introduzione

Questa specifica tecnica è parte delle **Smart City Platform Specification (SCPS)**
<http://smartcityplatform.enea.it/specification/>

Questo documento è la specifica **Smart City Platform Specification (SCPS) Semantic Level**
<http://smartcityplatform.enea.it/specification/semantic/>

Semantic Level

Le Specifiche del **Semantic Level** (o “Livello Semantico”) definiscono la semantica comune a cui aderire per permettere che lo scambio di dati (UrbanDataset) tra una Smart City Platform e le Solution verticali sia non ambiguo e quindi permetta di raggiungere una comunicazione interoperabile tra i sistemi.

Il core del Livello Semantico è l’**ontologia**: essa definisce gli Urban Dataset disponibili e ne rappresenta il riferimento principale, dato che tali definizioni, per ragioni di flessibilità, non possono essere documentate in maniera univoca dagli schemi definiti all’Information Level.

Per tale ragione l’ontologia è conservata e gestita in maniera centralizzata rispetto a una o più Smart City Platform per divenire, in questo modo, punto condiviso di convergenza semantica.

Per lo stesso motivo, il suo essere accentratore delle definizioni semantiche degli Urban Dataset, rende l’ontologia anche un nodo critico, la cui modifica e aggiornamento devono essere fatti con attenzione, attraverso un processo rigoroso e controllato.

Per tale ragione le specifiche definiscono:

- un Comitato di gestione dell’ontologia;
- una procedura formale per sottoporre la richiesta d’inserimento al Comitato;
- un template per sottoporre la richiesta

L’ontologia è un **file OWL** e può essere navigato e consultato tramite programmi esterni (siano essi desktop come Protégé o visualizzatori on line come WebVOWL). Inoltre, per facilitarne l’interrogazione, viene fornita una libreria software che permette di consultarla senza dover conoscere SPARQL.

In questo livello, inoltre, si definiscono anche le **Code List** che associano liste di codici utilizzate dagli Urban Dataset al loro significato. Queste Code List sono rappresentate con il formato standard OASIS genericode¹. Si noti come il modulo di richiesta di nuovi Urban Dataset contenga anche un foglio per l’inserimento di nuove code list.

¹ **OASIS**. Code List Representation (Genericode) Version 1.0. [Online] 2007. <http://docs.oasis-open.org/codelist/cs-genericode-1.0/doc/oasis-code-list-representation-genericode.html>

1 Descrizione dell'ontologia e modello semantico

L'ontologia pone centralmente il concetto di **UrbanDataset (UD)** e lo definisce come sottoclasse di **Entity** della Prov Ontology²

Con tale concetto si riferisce ogni dato, aggregato o meno, che un contesto applicativo è in grado di elaborare a partire dai dati raccolti nello Smart District. Questo è il nodo nevralgico di tutta la comunicazione e l'informazione principale che deve essere scambiata all'interno dell'infrastruttura.

Ontologie già esistenti, definite e testate in altri contesti, sono state utilizzate per associare, quando possibile, ai concetti e dati di Urban Dataset altre informazioni quali, per esempio, le unità di misura utilizzate per i valori raccolti dai sensori. In questo modo si hanno diversi vantaggi. Uno di questi è l'uso di una risorsa già testata e che quindi ha superato la validazione dovuta all'uso di tale strumento in un'altra applicazione. Un altro vantaggio è l'uso di un riferimento già noto ad altri sistemi automatici. L'utilizzo di una rappresentazione comune di un dominio permette di condividere anche tra umani la conoscenza di dominio facilitando la comprensione della struttura dei dati e favorisce il riuso di tali conoscenze anche in altre applicazioni e in contesti diversi.

Di seguito sono elencate le varie parti (che verranno riprese e chiarite anche più avanti) che compongono l'ontologia e che ruotano attorno al concetto principale di **UrbanDataset**. Il tutto è accompagnato da grafici che ne mostrano le relazioni (in tali immagini le parti in grassetto sono state definite per l'occasione mentre le altre sono pezzi di ontologie importate che completano e aiutano la sua definizione).

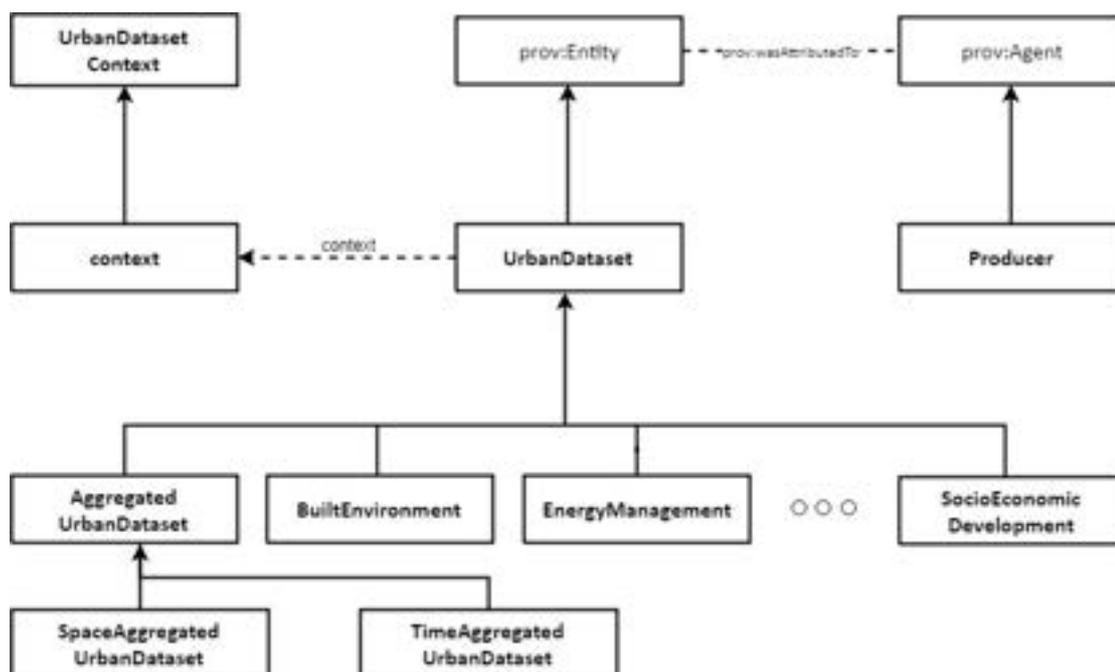


Figura 1. Schema di alcuni dei principali concetti relativi agli Urban Dataset

- **UrbanDatasetContext**: ogni Urban Dataset ha un suo contesto che serve per descrivere meglio le informazioni e per caratterizzarle, ad esempio posizione geografica a cui fanno riferimento i dati, lingua usata per le descrizioni, tempo di creazione e fuso orario. Le informazioni di un contesto sono modellate come proprietà.

² Raccomandazione del W3C che fornisce un insieme di classi, proprietà e restrizioni per rappresentare le informazioni generate da sistemi diversi. **W3C**. PROV-O: The PROV Ontology. [Online] 2013. <https://www.w3.org/TR/prov-o/>

- **Property:** un Urban Dataset, quando viene generato, viene associato ad altre informazioni (sia specifiche che di contesto) che lo descrivono, come il tempo di creazione, la posizione del sensore da cui è prodotto, la sua descrizione o il suo identificativo e la versione. Tali informazioni sono definite come proprietà dell'Urban Dataset o di un contesto.
- **Producer:** un aspetto importante di un set di dati è la sua provenienza, cioè chi è il fornitore dei dati. PROV Ontology, introdotta in precedenza, in questo caso svolge questa funzione. Nel caso specifico, *Producer*, sottoclasse di "Agent", (Figura 51) descrive l'ente a cui è demandata l'attività di creazione del dato. Come proprietà per collegare un UrbanDataset con il suo produttore può essere usata *wasAttributedTo* dell'ontologia PROV-O.
- **DataType:** per poter fare riferimento a quantità fisiche misurabili, si è deciso di sfruttare e importare l'ontologia OM che definisce le unità di misure del sistema internazionale (all'interno della classe **unit**). Per gestire ed elencare i tipi di dati si è invece creata la classe **DataType**. Esempi di istanze della classe DataType sono: integer, string, double.
- **ApplicationContext:** oltre alle specifiche proprietà, all'Urban Dataset è anche associato il contesto applicativo a cui si riferisce (es. SmartBuilding per il caso delle anomalie in un edificio) e un campo per indicare il produttore del dato.

Meccanismi di aggregazione

Un Urban Dataset può essere catalogato in funzione del suo livello di aggregazione temporale o spaziale. Questo permette sia di chiarire meglio il significato dell'Urban Dataset di per effettuare delle ricerche mirate su determinate categorie.

Tali livelli di aggregazione potranno assumere i seguenti valori

- per il livello di aggregazione spaziale:
 - **not applicable:** nel caso il concetto di aggregazione spaziale non abbia senso per lo specifico Urban Dataset;
 - **item:** se i dati si riferiscono al particolare sensore o, più in generale, oggetto, al livello di aggregazione più basso (per esempio, nel caso dell'illuminazione stradale, al singolo palo);
 - **facility:** se i dati si riferiscono all'intera struttura monitorata (per esempio, l'edificio, nel caso di Smart Building, lampioni di una via, per l'illuminazione stradale, ecc.);
 - **facilityAggregation:** se si riferisce a un insieme di strutture aggregate fra loro (per esempio un gruppo di Smart Building monitorati da uno stesso Aggregatore);
 - **city:** se i dati si riferiscono all'intera infrastruttura cittadina (per esempio l'intera rete di illuminazione pubblica di una città)
 - **region:** se ci si riferisce a infrastrutture che si estendono oltre la singola città, a un'intera zona/regione geografica (per esempio, impianti di depurazione delle acque)
 - **country:** se ci si riferisce ad infrastrutture che si estendono all'intera nazione (per esempio la rete di trasporto dell'energia elettrica)
- per il livello di aggregazione temporale:
 - **not applicable:** nel caso il concetto di aggregazione temporale non abbia senso per lo specifico Urban Dataset;
 - **static:** per Urban Dataset anagrafici, che definiscono le caratteristiche statiche di una struttura (per esempio localizzazione spaziale di un edificio)
 - **instantaneous:** se ci si riferisce a dati istantanei (per esempio temperature istantanea misurata da un sensore)

- **average**: se ci si riferisce a valori medi (per esempio il consumo medio di elettricità in un determinato intervallo di tempo, in un edificio)
- **total**: per valori totali (per esempio il consumo totale medio di elettricità in un determinato intervallo di tempo, in un edificio)
- **forecast**: per far riferimento a valori previsti e quindi futuri (per esempio previsioni meteo)

A tale scopo sono state definite le classi **SpaceAggregatedUrbanDataset** e **TimeAggregatedUrbanDataset**: un Urban Dataset con diversa aggregazione è definito come istanza di queste sottoclassi e riferisce, attraverso una proprietà, l'aggregazione usata e definita come istanza della classe **Aggregation** (Errore. L'origine riferimento non è stata trovata.).

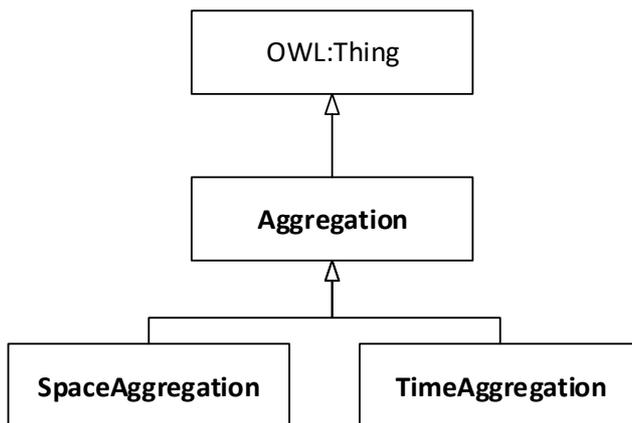


Figura 2. Schema dei concetti per definire il tipo di aggregazione

Proprietà degli Urban Dataset

Le singole proprietà dei diversi **UrbanDataset** sono a loro volta delle istanze di sottoclassi di **Property** (Figura 3). Si noti che queste proprietà rispecchiano il contenuto che dovranno avere le implementazioni sintattiche degli Urban Dataset (siano esse in sintassi XML o JSON) per rappresentare i dati che devono essere scambiati e, in tal senso, ne forniscono il riferimento *normativo*.

Property ha due sottoclassi:

- **ContextProperty**: raggruppa tutte le istanze di proprietà usate per descrivere il contesto (es. coordinate, lingua, tempo in cui è stato raccolto e inviato il dato, ecc.);
- **UrbanDatasetProperty**: raggruppa tutte le istanze di proprietà usate per descrivere proprietà specifiche degli Urban Dataset (es. il numero di anomalie, il numero di parcheggi occupati, l'assorbimento energetico medio, ecc.).

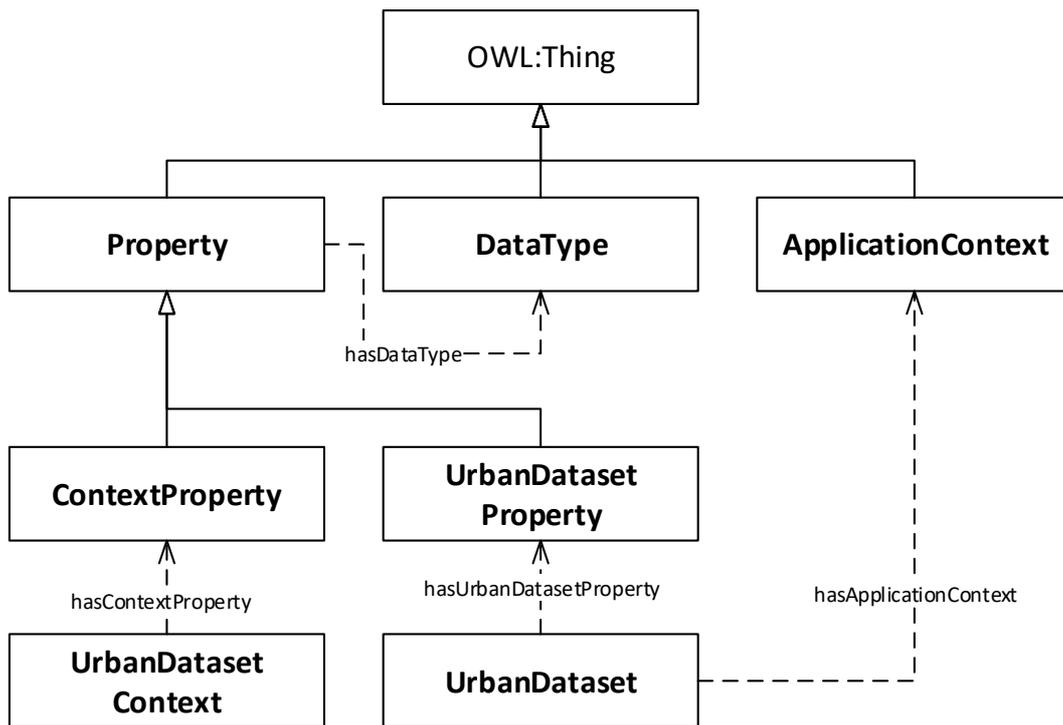


Figura 3. Schema dei principali concetti relativi alle proprietà degli Urban Dataset

L'ontologia descrive anche proprietà composte, ovvero è possibile che una proprietà sia composta a sua volta da altre proprietà. Ciò è descritto semplicemente creando un'istanza di proprietà che ha uno o più riferimenti ad altre istanze di proprietà tramite le object property *hasContextProperty* e *hasUrbanDatasetProperty*.

Si è pensato inoltre di creare una classificazione da associare alle proprietà che ne chiarisca il significato e ne faciliti l'individuazione e la ricerca. Per farlo si sono guardate alcune classificazioni esistenti, in particolare riguardanti i KPI cittadini e i contesti applicativi.

Una delle più complete è risultata essere quella proposta dalla ISO 37120 (ISO, 2018), che, oltretutto, ha il vantaggio di essere riconosciuta come standard internazionale. La lista delle classi delle proprietà create partendo da questo riferimento è:

Tabella 1 - Classificazione dei concetti associati alla proprietà

• Economy
• Education
• Energy
• Environment
• Finance
• FireEmergencyResponse
• Governance
• Health
• Recreation
• Safety
• Shelter

• SolidWaste
• Telecommunication
• Transportation
• UrbanPlanning
• Wastewater
• WaterSanitation

Andando a confrontare la lista delle proprietà presenti nell'ontologia, si nota che, nonostante la completezza di questa classificazione, alcune classi mancano per nostri scopi e sono state dunque aggiunte. Si tratta di:

- Identifier: per le proprietà che contengono un qualche tipo di identificatore (compresi i nomi)
- Description: per le descrizioni testuali
- Format: per i formati
- Time: le proprietà legate al tempo (come durata, timestamp, ecc.)
- Weather: le proprietà legate a previsioni o rilevazioni meteo.

Inoltre, per migliorare la ricerca dei concetti, per alcune delle precedenti classi, è stato necessario creare delle sottoclassi (per esempio Energy è stata divisa in Electric e Thermal).

Tale categorizzazione potrà essere usata, per esempio, per ricercare nell'ontologia, tramite l'interfaccia grafica di navigazione, le proprietà al fine di verificare se esiste già ciò che mi serve.

Ad ogni proprietà è stato associato il **DataType**, ovvero il tipo del dato usato per rappresentare l'informazione, ovvero un numero intero, un numero reale, una string o un timestamp, tramite la object property *hasDataType*. Istanze di DataType sono per esempio: **double**, **integer**, **string**. Oltre a questo è stata prevista l'associazione con un'unità di misura che qualifichi il dato. A questo scopo è stata usata l'ontologia OM (Ontology of units of Measure) descritta meglio in seguito.

Contesti applicativi degli Urban Dataset

Un UrbanDataset contiene un'aggregazione di dati provenienti dalle piattaforme locali della Smart City (per es. applicazioni per la gestione dell'illuminazione cittadina, per il traffico, ecc.) e perciò tali informazioni sono ascrivibili a un particolare contesto applicativo. Per questo motivo è stata prevista nell'ontologia l'esistenza di una proprietà che metta in relazione un'istanza di UrbanDataset con un contesto applicativo. Per fornire un'idea, in Figura 4 sono definite le istanze (indicate con testo sottolineato) che al momento un ApplicationContext può assumere attraverso la relazione *hasApplicationContext*. Inoltre, è specificato dall'ontologia che un UrbanDataset può essere associato esclusivamente a un solo ApplicationContext esplicitato tramite la dicitura *only*.

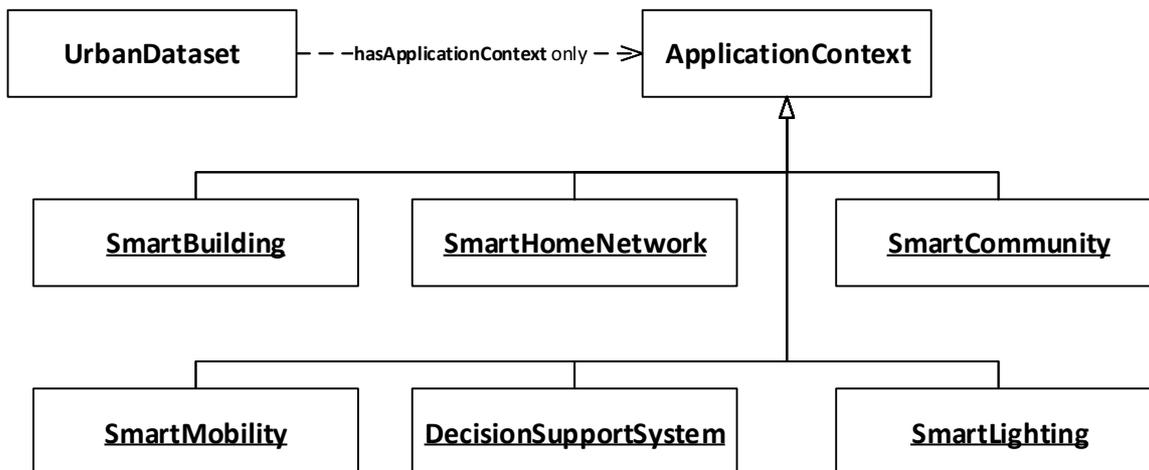


Figura 4. Associazione tra UrbanDataset e ApplicationContext

Uso di ontologie esterne

L'ontologia definita utilizza altre due ontologie esterne per una maggiore integrazione e compatibilità con l'esterno. La prima è PROV-O (Figura 5) che, come già detto, ha come scopo l'indicazione della provenienza di un dato. In particolare vengono utilizzate le classi:

- **prov:Entity**: una entità è un qualcosa di fisico, digitale, concettuale o altro con alcuni aspetti fissi; le entità possono essere reali o immaginarie;
- **prov:Agent**: un agent è qualcosa che possiede una qualche forma di responsabilità per un'attività, per l'esistenza di un'entità o per l'attività di un altro agent.

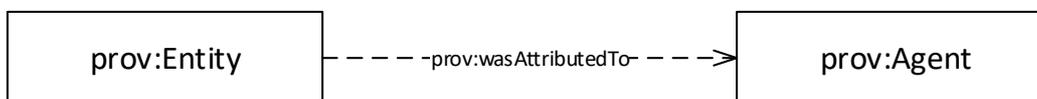


Figura 5. Uso di PROV-O

Le unità di misura sono rappresentate come istanze in modo che sia possibile associarle alle istanze di UrbanDataset. L'ontologia individuata che meglio soddisfa questa condizione è OM, che prevede un sistema di organizzazione delle unità di misura multiple e sottomultiple.

Essa contiene un'estesa lista delle unità di misura, del Sistema Internazionale (SI) e non, definite come istanze della classe **unit**. Ha adottato un sistema che permette di definire multipli e sottomultipli per dieci delle diverse unità di misura fornendo una struttura per aggiungere anche multipli e sottomultipli non ancora presenti ma possibilmente utili in futuro per specifiche unità di misura. Utilizza per fare ciò una sottoclasse della classe unit detta **PrefixedUnit** e associa a ogni istanza di questa classe il prefisso usato (es. kilo o milli) e l'unità di misura base del SI. Le istanze dei prefissi sono definite staticamente come istanze della classe *Prefix*.

In Figura 6 è mostrata la struttura completa dell'ontologia OM. Nel nostro caso non viene usata tutta, ma solo la parte relative alle singole unità di misura, ovvero le istanze della classe Unit.

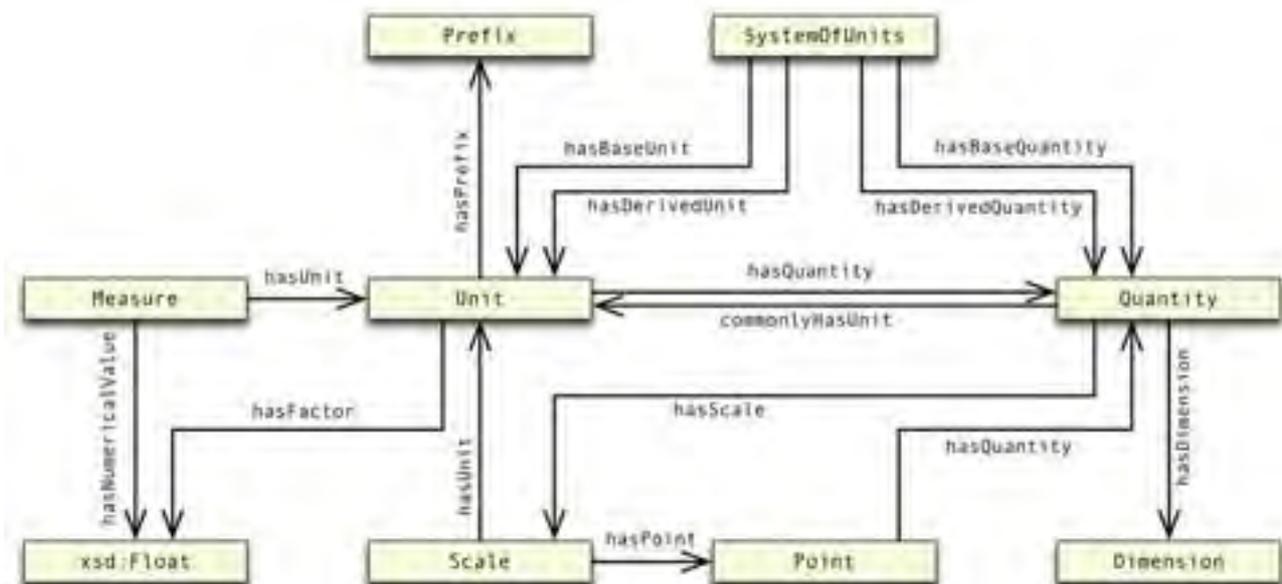


Figura 6. Struttura delle classi dell'ontologia OM

Informazioni sui produttori dei dati

I produttori dei dati possono essere diversi, dato che le diverse piattaforme, sorgenti dei dati, i possono essere gestite da enti diversi. Diventa necessario tenere traccia di chi è il responsabile dei diversi Urban Dataset ed è quindi utile avere informazioni dettagliate al riguardo. A questo scopo esiste già una ontologia, il cui nome è FOAF [17], definita a partire dal formato vCrad con lo scopo di gestire e descrivere le informazioni di contatto di persone e organizzazioni. In questo modo, alle istanze della classe Producer possono essere aggiunte informazioni strutturate già comprensibili ai sistemi in grado di analizzare il formato FOAF.

Code List

In generale, un flusso di informazioni è composto da tre tipologie di dati:

- dati fattuali (per es. i valori assunti dalle proprietà di un Urban Dataset)
- testi (per es. la descrizione di un Urban Dataset)
- codici (per es. l'unità di misura delle proprietà degli Urban Dataset).

È buona pratica vincolare a dei vocabolari controllati i valori che possono essere assunti dagli elementi del flusso che ricadono nella terza tipologia. Tali vocabolari, infatti, possono essere pensati come insiemi predefiniti di valori in cui ogni valore rappresenta una semantica concordata tra le parti che lo usano; essi favoriscono innanzitutto l'interoperabilità fra le applicazioni che li adottano e inoltre, supportando il multilinguismo, consentono a utenti di comunità diverse di meglio cogliere il significato dei valori che rappresentano (un esempio in **Errore. L'origine riferimento non è stata trovata.**).

Tabella 2. Sottoinsieme della Code List "Unit Of Measure" dell'UNECE

Codice	Descrizione italiano	Descrizione inglese
MTR	metro	metre
KWH	kilowattora	kilowatt hour
KWT	kilowatt	kilowatt
INH	pollice	inch

L'istanza di un vocabolario controllato costituisce una **Code List** (o **Lista di Codici**); le Code List sono caratterizzate dalla *versione*, che consente di tracciare le variazioni che, nel corso del tempo, può subire l'insieme dei valori che compongono il vocabolario implementato (per es. aggiunta di valori, modifica al significato di valori esistenti, ecc.).

Le Code List possono essere espresse in modo formale attraverso lo standard **OASIS Genericode**³.

Al fine di favorire l'interoperabilità semantica tra le applicazioni che aderiscono alle specifiche per la Smart City Platform, è stato deciso di definire, a livello di sottospecifica semantica, sia quali degli elementi che compongono un Urban Dataset devono essere vincolati a Code List, sia il vocabolario da utilizzare (quindi le liste di valori e relativa semantica). Per associare una Code List ad una proprietà, o in generale ad un elemento dell'ontologia si è definita la Datatype Property *hasCodeList*, che può assumere come valore l'URL che consente di accedere alla specifica completa (tipicamente un file) della Code List stessa. Ad esempio, la proprietà *BuildingEnergyAnomalyCode* ha valorizzata la Datatype Property *hasCodeList* con <http://smartcityplatform.enea.it/specification/semantic/1.0/gc/BuildingEnergyAnomalyCode.gc>.

Alcune delle Code List definite sono valide per tutti gli Urban Dataset (**Errore. L'origine riferimento non è stata trovata.**); altre sono strettamente legate allo specifico Urban Dataset (

³ **OASIS**. Code List Representation (Genericode) Version 1.0. [Online] 2007. <http://docs.oasis-open.org/codelist/cs-genericode-1.0/doc/oasis-code-list-representation-genericode.html>

Tabella 4). Come nell'esempio precedente, alcune delle Code List indicate nell'ontologia sono state formalizzate anche in formato Genericode (si veda la cartella <http://smartcityplatform.enea.it/specification/semantic/1.0/gc/>).

Tabella 3. Esempi di Code List valide per qualsiasi Urban Dataset

Code List ⁴	Descrizione ⁵	Genericode ⁶	Elemento ⁷	Concetto ⁸	Urban Dataset ⁹
TimezoneCode	Fusi orari	TimezoneCode.gc	timeZone	timezone	Tutti
FormatCode	Formati coordinate geografiche	FormatCode.gc	format	format	Tutti
LanguageCode	Lingue	LanguageCode.gc	language	language	Tutti

⁴ Code List: nome che identifica la Code List

⁵ Descrizione: descrizione della Code List

⁶ Genericode: nome del file genericode che formalizza la Code List (tutti i file sono disponibili all'url: <http://smartcityplatform.enea.it/specification/semantic/1.0/gc/>)

⁷ Elemento: elemento del Modello Dati Astratto per il quale è richiesto l'uso di questa Code List (vedere specifiche sintattiche)

⁸ Concetto: concetto dall'ontologia a cui è associata la Code List. Poiché una Code List può essere associata a più concetti, se ne riporta solo uno a titolo di esempio.

⁹ Urban Dataset: nome dell'Urban Dataset che richiede l'uso della Code List indicata

Tabella 4 – Esempi di Code List valide per specifici Urban Dataset

Code List	Descrizione	Genericode	Concetto
BuildingEnergyAnomalyCode	Codice identificativo dell'anomalia dell'edificio.	BuildingEnergyAnomalyCode.gc	BuildingEnergyAnomalyCode
TreatmentPlantAnomalyCode	Codice identificativo dell'anomalia dell'impianto.	TreatmentPlantAnomalyCode.gc	TreatmentPlantAnomalyCode
DeviceAnomalyCode	Codice identificativo dell'anomalia del device.	DeviceAnomalyCode.gc	DeviceAnomalyCode

2 Regole di naming per i concetti dell'ontologia

Le specifiche definiscono delle regole di naming per costruire i nomi degli Urban Dataset e delle Proprietà. lo scopo è sempre quello di **rendere** più facilmente identificabile il significato degli UD già presenti, questa volta per gli esseri umani. La necessità è nata per evitare i casi di Urban Dataset con identificatori simili ma significati diversi o, viceversa, con significati simili ma identificatori molto diversi.

Regole di naming per gli Urban Dataset:

L' identificatore dell'Urban Dataset viene costruito mettendo in sequenza:

- L'oggetto su cui si sta effettuando la misurazione (per esempio Building se si sta misurando qualcosa nell'edificio, Home, per misurazioni di parametri della casa, ...)
- Un modificatore che specifica se l'Urban Dataset riporta dati aggregati (Aggregated). Nel caso di dati dettagliati (cioè la lista di dati per singoli item, per esempio singoli appartamenti sottoposti a monitoraggio e inviato insieme in un singolo UD) invece non viene specificato nulla perché si ritiene questa sia la situazione di default
- La categoria della tassonomia di Tabella 1 (pag. 9) a cui fa riferimento l'UD
- Se la categoria è identica all'oggetto della misurazione o se si parla di dati anagrafici, questa viene omessa (o si utilizza la categoria di ordine superiore)
- La definizione di ciò che trasporta l'Urban Dataset. Sintetizzare tale definizione in una parola è più semplice nel caso di UD che trasportano un singolo parametro (per esempio le anomalie). La situazione diventa più complicata se i parametri trasportati sono "spuri" (per esempio, come avviene nel caso delle acque, in cui un UD, individuato nei casi studio, trasporta sia il consumo di energia che la portata di acque trattata nel periodo). In questo caso si individua un termine il più possibile rappresentativo del set d'informazioni trasportato (per esempio, nel caso delle acque, si parlerà di Performance e per le condizioni meteo si parlerà di Condition o Forecast).

Inoltre:

- Le parole da utilizzare non devono essere mai troncate (si userà sempre "Condition", "Electric", "Consumption" e mai "Cond", "Elec", "Cons").
- Gli identificatori degli UD non hanno spazi, iniziano con la maiuscola e usano il Camel Case per individuare le diverse parole che lo compongono



Figura 7. Regole di naming per UD

Regole di naming per le Proprietà:

- Gli identificatori delle proprietà non hanno spazi, iniziano con la maiuscola e usano il Camel Case per individuare le diverse parole che lo compongono;

- Gli identificatori delle proprietà relative al consumo (elettrico, termico, di acqua, ecc.) devono terminare con “Consumption”;
- Gli identificatori delle proprietà relative alla produzione (elettrico, termico, di acqua, ecc.) devono terminare con “Production”;
- Gli identificatori delle proprietà della classe Name, finiscono con “Name”;
- Gli identificatori delle proprietà della classe ID, finiscono con "ID";
- Gli identificatori delle proprietà TextDescription finiscono con "Description"
- Gli identificatori delle proprietà indicanti un “Numero di...” finiscono con “Number”
- Gli identificatori delle proprietà associati alla dimensione di superficie finiscono con “Area”
- Gli identificatori delle proprietà associate a righe di una codelist o comunque a codici, finiscono per “Code”
- Le parole da utilizzare non devono essere mai troncate (si userà sempre “Condition”, “Electric”, “Consumption” ecc. e mai “Cond”, “Elec”, “Cons”).
- Identificatori di proprietà troppo generiche rispetto alle esigenze, vengono costruiti a partire le regole di naming viste per gli UD (per esempio “BuildingAnomaliesCode” è costruito tramite le regole degli UD più la regola 7 di questa lista)

3 Consultazione dell'ontologia

L'ontologia è stata formalizzata nel linguaggio OWL e resa disponibile online e scaricabile al link:

<http://smartcityplatform.enea.it/specification/semantic/1.0/ontology/scps-ontology-1.0.owl>

L'ontologia importa alcune ontologie esterne, che vengono caricate dal web. Nel caso l'importazione non andasse a buon fine, devono essere pre-caricate e rese disponibili.

L'ontologia può essere visualizzata attraverso i normali programmi per l'editing dei file OWL. In particolare può essere visualizzata attraverso programmi desktop come Protégè (<https://protege.stanford.edu/>). In questo caso il programma deve prima essere installato sul computer e poi eseguito per poter caricare e visualizzare il file. Protégè può essere scaricato al link <https://protege.stanford.edu/products.php#desktop-protege>.

La figura seguente riproduce la visualizzazione dell'ontologia tramite Protégè.

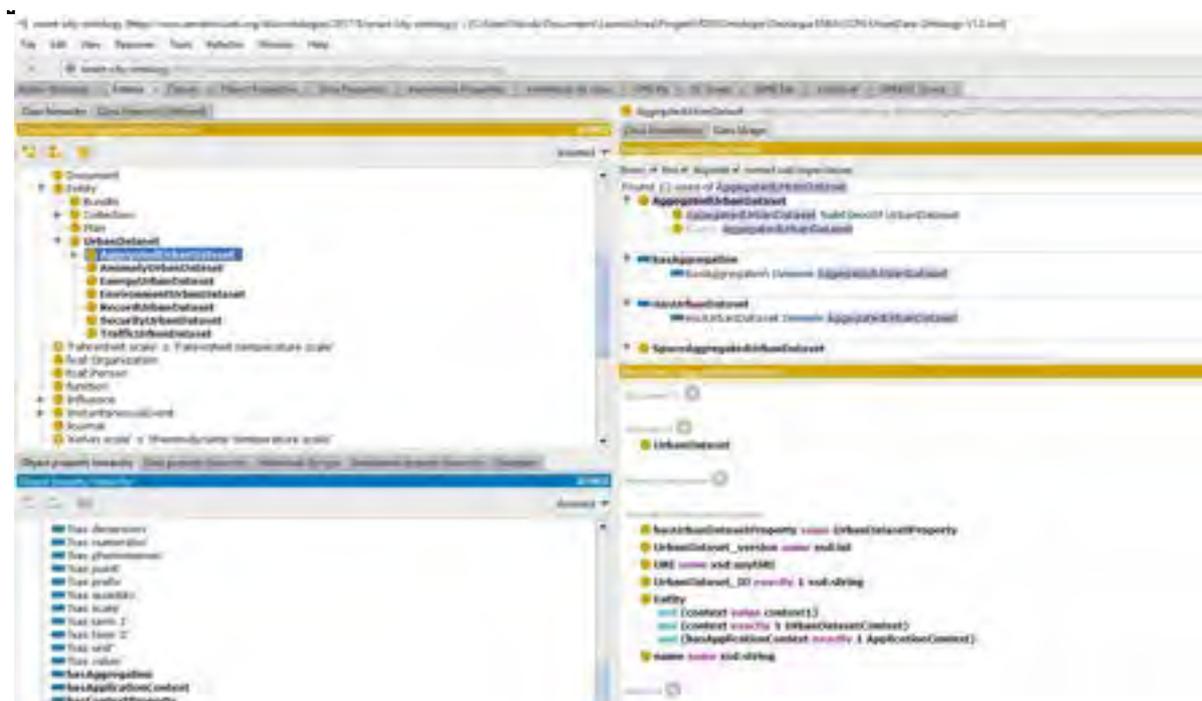


Figura 8. Visualizzazione dell'ontologia con Protégè

Altro modo per visualizzare l'ontologia è ricorrere a visualizzatori online come WebVOWL, disponibile al link <http://vowl.visualdataweb.org/webvowl.html>. In questo caso l'applicazione online consente di indicare dal menù in basso nella pagina web un file locale o un indirizzo web da cui caricare i dati dell'ontologia per poterla poi visualizzare. La Figura 9 riproduce la visualizzazione dell'ontologia tramite WebVOWL.

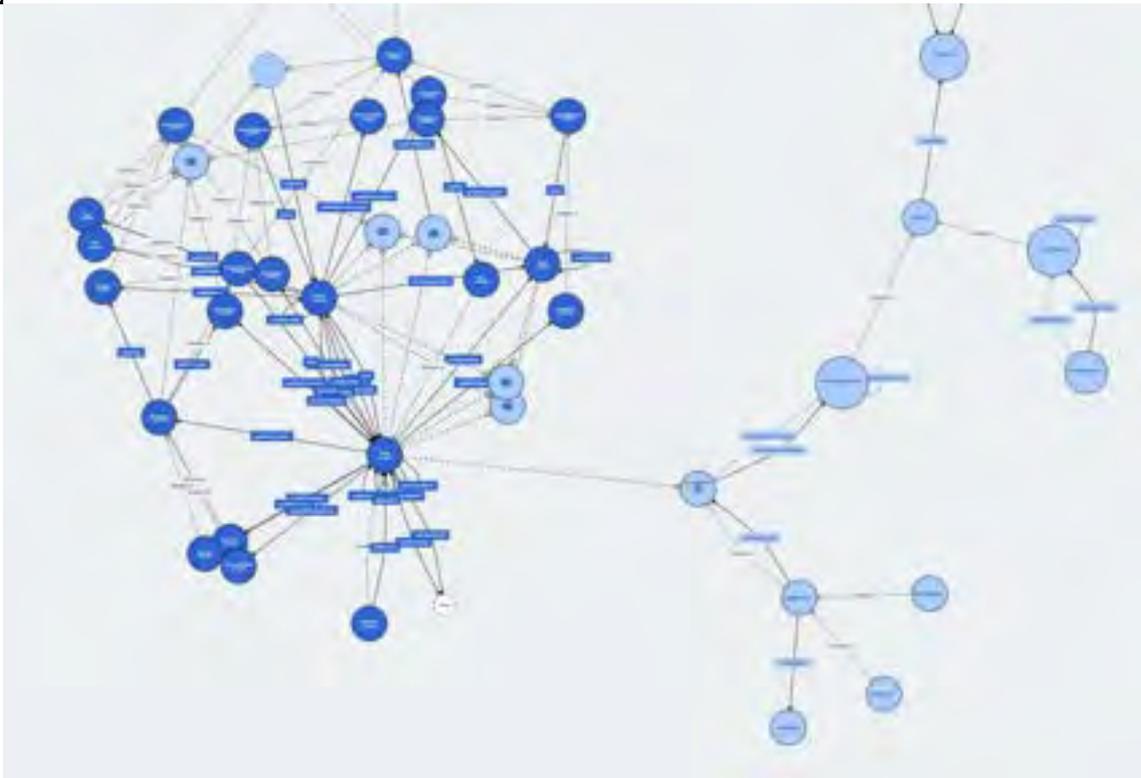


Figura 9. Visualizzazione dell'ontologia con WebVOWL

L'ontologia, in questo caso, è visualizzata come grafo nel quale sono rappresentati graficamente concetti e relazioni fra di essi e dal quale è possibile avere rapidamente una visione di insieme. Selezionando e cliccando sui vari oggetti del grafo, l'applicazione fornisce tutte le informazioni disponibili sull'entità selezionata.

Infine è stato sviluppato un servizio web specifico per facilitare la navigazione e la comprensione dell'ontologia. Il servizio è disponibile al link:

<http://smartcityplatform.enea.it:8080/SCPSWebLibrary/ontologyinfo>

La fig. 9 mostra una delle interfacce di navigazione predisposte per fornire informazioni sugli Urban Dataset. Attraverso varie voci di menù è possibile navigare o fare ricerche all'interno dell'ontologia e andare a estrarre e visualizzare tutte le informazioni in essa contenute.



Figura 9. Interfaccia per la visualizzazione dell'ontologia della SCPS

4 Interrogazione dell'ontologia

Per facilitare e migliorare l'adozione e uso dell'ontologia, oltre ad aver ipotizzato delle query di accesso alle informazioni contenute nell'ontologia, si è provveduto a sviluppare una libreria software che permetta di interagire con l'ontologia senza dover conoscere in maniera fine la sua struttura o il linguaggio di interrogazione. In questo capitolo è mostrata l'architettura della libreria realizzata.

Per interrogare un'ontologia sviluppata con le tecnologie del web semantico è necessario conoscere il linguaggio di interrogazione SPARQL. Dal momento che una stessa interrogazione è molto probabile che sia rivolta più di una volta, ha senso avere delle query predeterminate che possono essere richiamate con un comando più semplice preso da una lista definita a priori.

Questa modalità di funzionamento non obbliga chi vuole accedere alle informazioni contenute nell'ontologia a imparare un nuovo linguaggio di interrogazione e per questo risulta più facile integrare queste query all'interno di software più grandi e con uno scopo diverso ma che necessitino delle informazioni presenti nell'ontologia.

Un altro obiettivo è quello di fornire una copertura completa per l'accesso alle informazioni presenti nell'ontologia. A partire dalla struttura dell'ontologia definita, si è partiti con l'ideazione di una serie di richieste che permettessero di accedere a tutte le informazioni presenti senza conoscere la struttura e da punti di partenza diversi (es. conoscendo il soggetto o l'oggetto di una relazione). La libreria, quindi, deve permettere di recuperare le informazioni conoscendo solo in parte ciò che si vuol sapere e per questo risulta molto importante un buon grado di facilità di esplorazione a partire dalle interfacce fornite.

Un requisito necessario riguarda la possibilità di accedere alle informazioni dell'ontologia in remoto per cui è necessario che la libreria sviluppata sia in grado di raggiungere su http il server che contiene l'ontologia.

Un altro vincolo importante riguarda l'utilizzo di Java come linguaggio di programmazione. Si prevede che questa libreria sia sviluppata per essere integrata all'interno di altre applicazioni. Java è uno dei linguaggi più diffusi e conosciuti perciò per garantire una facilità di utilizzo della libreria si è optato per questa scelta.

Architettura

Partendo dalle caratteristiche appena descritte si è arrivati alla definizione dell'architettura. Per poter recuperare le informazioni presenti nell'ontologia, questa deve essere inserita all'interno di un server che accetti richieste di interrogazione ed essendo tecnologie sviluppate dal W3C, le richieste vengono portate al server attraverso protocollo web http. Per affrontare tale vincolo e ottenere già un supporto alle astrazioni dei linguaggi SPARQL e RDF si è deciso di utilizzare una libreria già sviluppata per tali scopi. A questo proposito si è scelto di utilizzare Apache Jena¹⁰. Questa libreria fornisce una serie di strumenti utili per interagire con un server SPARQL e fornisce un supporto per memorizzare ed elaborare in locale dataset in formato RDF. Il software in questione è sviluppato da diversi anni ed è ancora supportato e in evoluzione. Questo garantisce che il prodotto abbia una buona maturità e affidabilità e una garanzia di evoluzione e supporto futuri sufficiente, oltre che necessaria per l'utilizzo in un sistema complesso e duraturo come quello oggetto di tali specifiche. Un'ulteriore garanzia deriva dalla diffusione che ha Apache Jena, infatti risulta essere una delle più usate in ambiente Java per questo tipo di attività.

La libreria sviluppata, denominata SmartCityOntology Library, si interfaccia, quindi, con Apache Jena fornendo un accesso semplificato alle sue operazioni. Tale accesso è costruito in maniera specifica per l'ontologia definita in precedenza. La

Figura mostra, attraverso la rappresentazione grafica, che la libreria definita ha come compito quello di nascondere la complessità dell'accesso a un sistema SPARQL e per farlo sfrutta le operazioni messe a disposizione da Apache Jena.

¹⁰ <https://jena.apache.org/>

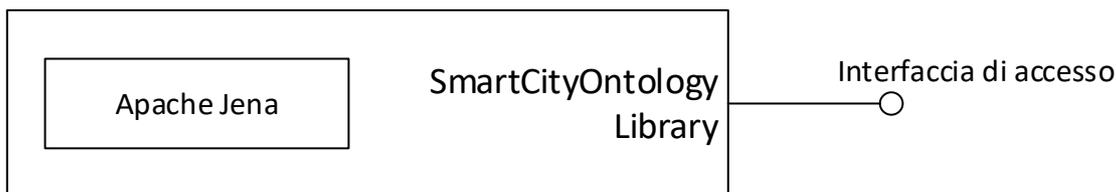


Figura 10. Architettura della libreria

La libreria deve svolgere una funzione piuttosto semplice, ovvero prevedere un set di query pronte da usare per interrogare la base di dati. Un approccio in cui si preparano solamente delle query dentro dei metodi potrebbe essere veloce all’inizio ma difficile da estendere. Per cui si è pensato di organizzare la sua architettura su più livelli in modo da poter riusare le funzionalità dei livelli sottostanti per creare più facilmente interrogazioni e funzionalità più complesse.

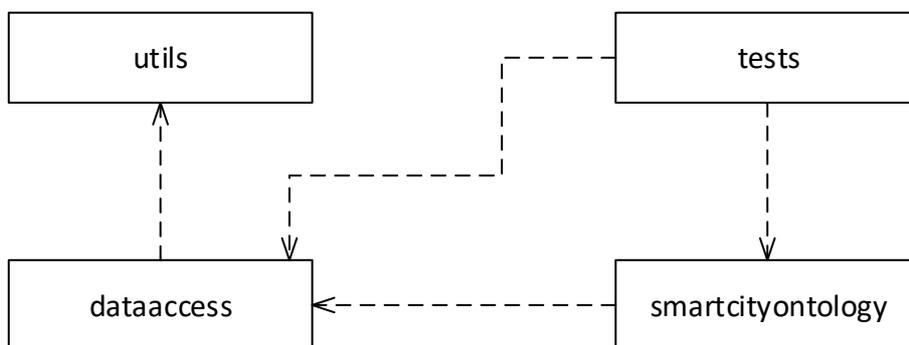


Figura 11. Architettura della libreria

In Figura 11 è mostrato il diagramma dei 3 package della libreria:

- nel package **utils** sono state inserite le informazioni per la configurazione della libreria stessa (classe *Configs*, che legge il file *config.properties*) e sono stati creati degli scheletri di query per costruire dinamicamente l’interrogazione (classe *PresettetQueries*);
- nel package **dataaccess** sono state inserite le classi per effettuare la connessione al sistema remoto (classe *EndpointConnection*) e per costruire interrogazioni che, appoggiandosi sulle funzionalità del package *utils*, recupera le informazioni di una generica ontologia a partire da parametri specifici. A questo scopo sono presenti le due classi *PropertyWrap* e *ResourceWrap*. In particolare, in questo package sono presenti metodi per poter ricostruire la struttura dell’ontologia e per questo potrebbe essere utilizzata anche per altre ontologie diverse dalla nostra. Naturalmente questo è ottenuto nascondendo della complessità ma allo stesso tempo limitando i gradi di libertà;
- il package **smartcityontology**, infine, è quello che contiene le operazioni per recuperare le informazioni specifiche per l’ontologia definita in precedenza. Tali operazioni sono possibili attraverso la classe *OntologyQuery* che consiste di metodi statici che operano sull’ontologia SCPS-UrbanDataset (esempi di questi metodi sono: *getUrbanDatasetInstancesSubClasses*, *getUrbanDatasetPropertyInstances*, ...). Sono stati implementati anche una serie di test automatici per verificare con il comportamento della libreria sia corretto.

Al momento dell’uso della libreria viene creato in automatico la connessione verso il server definito all’interno di un file di configurazione. Naturalmente è possibile modificare il server remoto anche durante l’utilizzo, ma è possibile interrogare solo un server alla volta. Questa scelta si è basata sul fatto che, essendo pensata per una specifica ontologia, si presuppone che venga usata per interrogare l’ontologia presente su un particolare server che fornisce la funzione di indice per i dataset.

Un'altra scelta è stata quella di avere un sistema che costruisce le query utilizzando più livelli per rendere più facile l'estensione nel caso l'ontologia dovesse subire delle modifiche come delle aggiunte di nuovi concetti o una variazione della struttura.

Sono state previste anche delle query che permettono di limitare il numero di risultati recuperati dal server. Si possono creare filtri, attraverso la libreria o da parte dell'utilizzatore, sui dati recuperati dal server in caso non sia necessaria la lista completa. Questo, però, potrebbe causare dei problemi di sovraccarico del server e rallentare il recupero delle informazioni. Infatti, nel caso in cui la query dovesse ottenere come risposta una lista molto lunga di dati, causerebbe un rallentamento nel recupero e nell'invio degli stessi, per cui si è riportato anche nella documentazione la preferenza a richiedere che sia il server stesso a limitare l'output, invece che il software che integra la libreria. In questo modo si può ottenere un carico minore sul server e riuscire a gestire più richieste contemporanee.

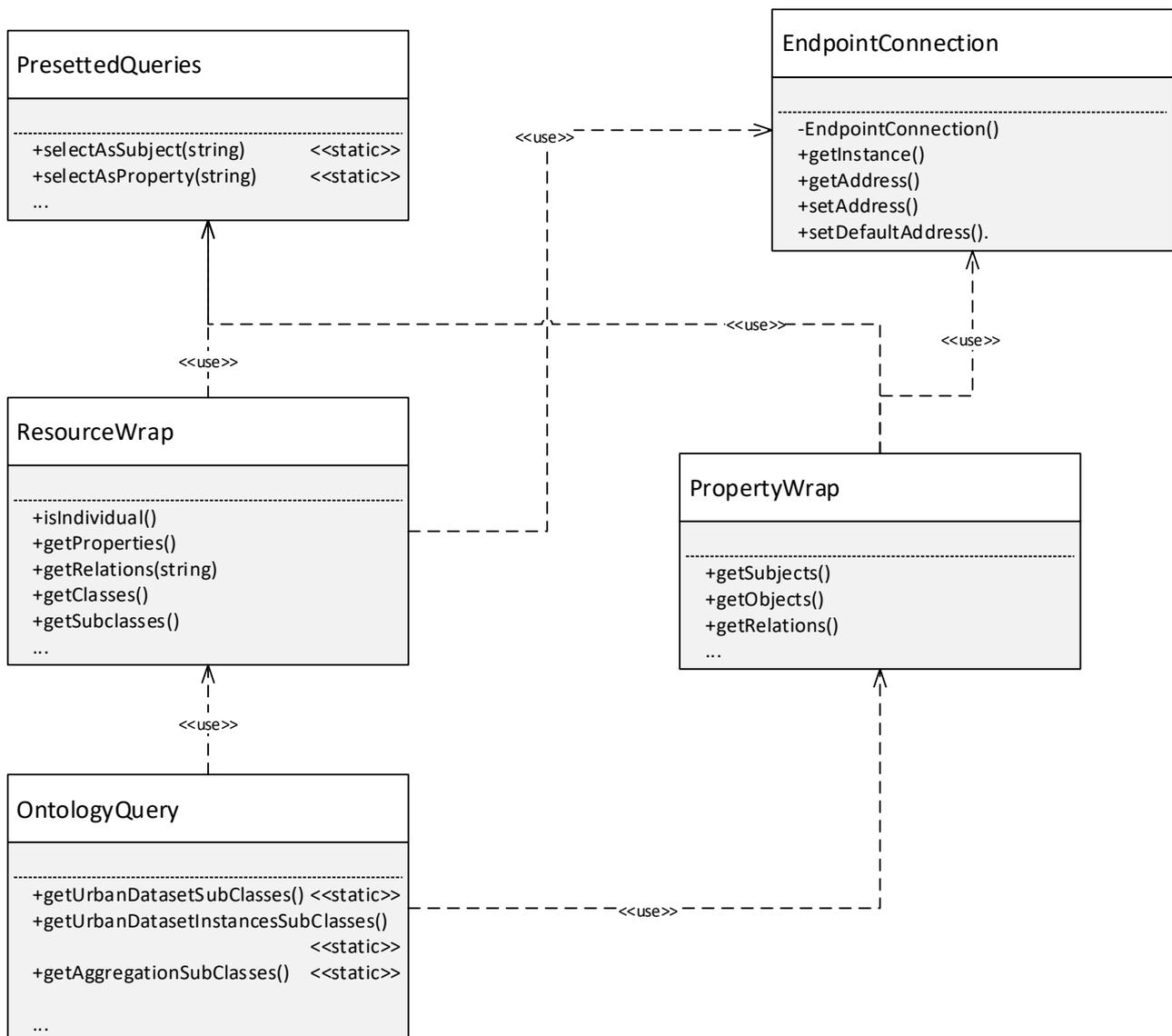


Figura 12. Diagramma delle classi

In Figura è mostrato un diagramma parziale delle classi della libreria. Sono mostrati solamente alcuni dei metodi presenti. Di seguito sono elencate e descritte - alcune classi

- *OntologyQuery*: definisce le query principali specifiche per l'ontologia, utilizzabili attraverso dei metodi get che consentono di estrarre particolari dati

- *ResourceWrap* e *PropertyWrap* sono delle classi pensate per fungere da wrapper rispettivamente per le classi *Resource* e *Property* di Apache Jena. Aggiungono metodi per verificare se una risorsa è una istanza o meno, recuperare le sue sottoclassi o le sue istanze e le relazioni in cui sono coinvolte all'interno dell'ontologia. Queste due classi mantengono il riferimento al server remoto nascondendolo all'esterno. Naturalmente il riferimento al server può essere recuperato dall'esterno per modificare l'indirizzo del server durante l'esecuzione.
- La classe *EndpointConnection* gestisce la connessione alla prima esecuzione prendendo l'indirizzo da un file di configurazione che può quindi essere modificato dall'utente prima dell'esecuzione ed evita di cablare nel codice l'indirizzo del server che gestisce le richieste garantendo flessibilità.

Infine è stata creata una documentazione (in formato html) che descrive le diverse classi e i metodi della libreria per sapere le operazioni che si possono effettuare, e per facilitarne l'uso e l'integrazione all'interno di altri software.

5 Inserimento di nuovi Urban Dataset

5.1 Il Comitato di gestione dell'ontologia

Al fine di mantenere l'uniformità e la coerenza semantica, fondamentali per l'interoperabilità, anche nel caso di implementazioni autonome e indipendenti della piattaforma, si ritiene importante che l'ontologia sia gestita come risorsa centralizzata.

L'accesso all'ontologia è libero: per cui è possibile scaricarla, modificarla e installarla presso i propri server. In questo caso, però, interoperabilità e uniformità semantica non sono più garantite. Per tale ragione, implementazioni della piattaforma non basate sull'ontologia centrale potrebbero risultare non conformi alla sottospecifica livello semantico.

Manutenzione, gestione e aggiornamento dell'ontologia sono a cura dal **Comitato di Gestione** dell'ontologia della Smart City Platform.

In particolare, tale Comitato ha il compito di:

- formalizzare e seguire la procedura per la richiesta di inserimento di nuovi Urban Dataset
- interfacciarsi con gli utenti che vogliono effettuare tale richiesta
- esaminare le richieste di inserimento di nuovi Urban Dataset
- aggiornare l'ontologia, quando necessario (sia in caso di input esterni, che di decisioni interne)

Il Comitato è costituito da personale del laboratorio DTE-SEN-CROSS di ENEA e del Dipartimento di Informatica - Scienza e Ingegneria (DISI) dell'Università di Bologna.

5.2 Procedura per la richiesta d'inserimento

Al fine di avviare la procedura per la richiesta di un nuovo Urban Dataset, l'utente deve compilare l'apposito modulo (che viene spiegato in dettaglio nel paragrafo successivo) e inoltrarlo al Comitato nelle inviando una mail all'indirizzo smartcityplatform.project@enea.it, con oggetto "Richiesta inserimento nuovo Urban Dataset: [NOME URBAN DATASET]".

L'invio, per ottimizzare la procedura, dovrebbe avvenire in seguito a un'attenta analisi, da parte dell'utente, dell'ontologia esistente al fine di verificare se l'Urban Dataset richiesto non esista già o se siano già definite, in essa, proprietà utilizzabili per costruire il modello del nuovo Urban Dataset¹¹.

Si noti che i nomi delle proprietà da inserire nell'ontologia devono essere scritti **in inglese**, per lasciare valenza internazionale alla stessa.

In seguito alla richiesta il Comitato:

- valuta
 - coerenza dell'Urban Dataset con i fini e con la struttura delle Smart Platform Specification
 - coerenza sintattica delle proprietà richieste
- esamina l'ontologia per verificare che
 - non esista già un Urban Dataset equivalente a quello richiesto
 - alcune fra le proprietà in esso presenti, non siano già presenti nell'ontologia, seppure espresse in modo diverso
 - le eventuali liste di codici richieste, da associare a proprietà dell'Urban Dataset non siano già presenti, seppure parzialmente, nell'ontologia.
- in seguito al precedente esame, decide se:

¹¹ Per esempio, nel caso che esista già la proprietà "electric consumption", è inutile crearne un'altra chiamata "consumption of electricity" da inserire nel proprio Urban Dataset

- inserire l'Urban Dataset, così come è stato richiesto
 - richiedere modifiche alla struttura dell'Urban Dataset richiesto
 - rifiutare, motivando il rifiuto, l'inserimento dell'Urban Dataset. Motivazioni possono essere: la presenza di un Urban Dataset equivalente nell'ontologia, che può essere utilizzato in luogo di quello richiesto, oppure forti incoerenze con le specifiche, che non siano risolvibili tramite delle modifiche
- aggiorna l'ontologia, effettuando l'inserimento dell'Urban Dataset e producendone una nuova versione
 - invia notifica, al richiedente e a tutti gli utenti registrati, della presenza di una nuova release dell'ontologia

La procedura è illustrata nello schema a blocchi della figura successiva:

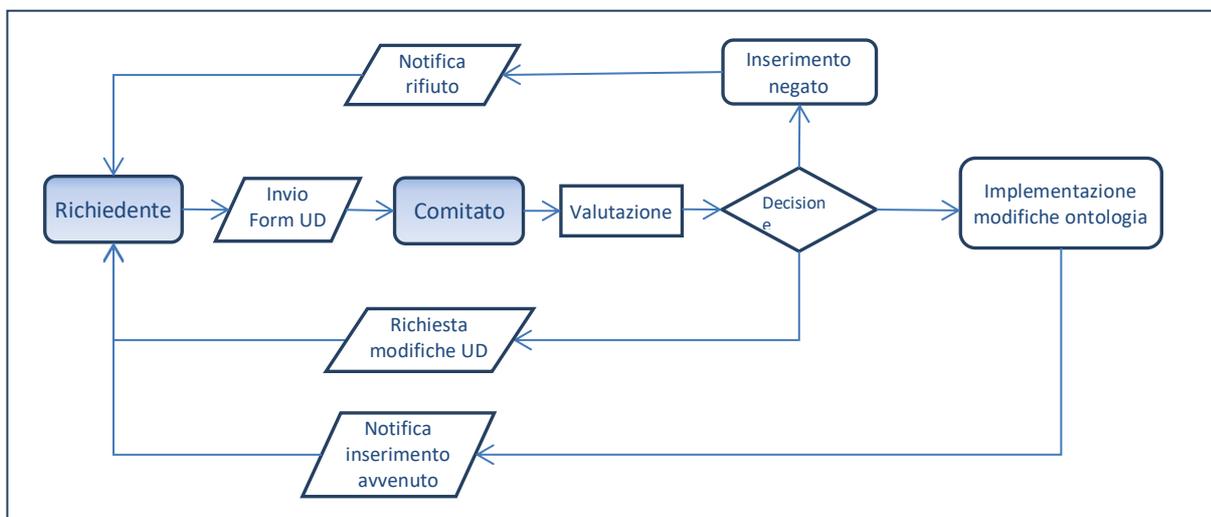


Figura 13. Schema a blocchi della procedura di richiesta di inserimento di un nuovo Urban Dataset

5.3 Modulo per la richiesta di inserimento

Per fornire, al Comitato, la proposta di inserimento di un nuovo Urban Dataset, si suggerisce di utilizzare il modulo SchedaPropostaUD Excel disponibile nella cartella:

<http://smartcityplatform.enea.it/specification/semantic/1.0/update/>

Il modulo è costituito da tre parti:

1. **Caso d'uso:** contiene alcune informazioni che permettono ai valutatori di inquadrare meglio il caso d'uso in cui si colloca l'utilizzo dell'Urban Dataset proposto.

In particolare vanno inseriti, nel presente modulo:

- Il nome del caso d'uso
- L'attore che produce l'Urban Dataset (inteso come tipologia di piattaforma: per esempio Smart Building, Smart Lighting, ecc)
- L'attore che riceve l'Urban Dataset (anche in questo caso inteso come tipologia (per esempio ESCO)
- Il nome dell'Urban Dataset
- Il formato che si intende usare (XML, JSON o altro nel caso si utilizzi un formato diverso, quale CSV)

- La Policy da applicare all'Urban Dataset: sarà trattato come Open Data oppure l'accesso è ristretto (privato)?

Il modulo, disponibile all'indirizzo:

- <http://smartcityplatform.enea.it/specification/semantic/1.0/update/OF224-008-v1-SchedaPropostaUD.xlsx>

è mostrato nella seguente figura.

1. INFORMAZIONI SUL CASO D'USO		
Input	Descrizione	Esempio
Nome del Caso d'uso	Inserire nome del caso d'uso per cui si richiede l'Urban Dataset	Electric consumption
Attore 1 - Produttore	Inserire la tipologia di piattaforma che produce l'Urban Dataset	Smart Home
Attore 2 - Ricevente	Inserire la tipologia di piattaforma che riceve l'Urban Dataset	ESCO platform
Nome Urban Dataset	Inserire il nome dell'UD	
Formato prescelto	Formato o formati prescelti	
Policy dell'UD	Si tratta di Open Data o il suo accesso è limitato?	Open Data / privato

Figura 14. Prima pagina del foglio di calcolo per la richiesta di un nuovo Urban Dataset: descrizione del caso d'uso

2. **Urban Dataset:** il modulo per l'inserimento degli Urban Dataset è quello più complesso e si divide in tre parti: categorizzazione, contestualizzazione e proprietà dell'Urban Dataset.

Nel primo riquadro (**categorizzazione**) vengono inseriti, oltre al nome dell'Urban Dataset, alcuni dati che permettono di categorizzarlo nell'ontologia. Si tratta di:

- Livello di aggregazione spaziale, che può assumere i seguenti valori:
 - **Not applicable:** se il concetto di aggregazione spaziale non è applicabile all'Urban Dataset che si sta definendo
 - **Item:** se i dati si riferiscono al particolare sensore o, più in generale, oggetto, al livello di aggregazione più basso (per esempio, nel caso dell'illuminazione stradale, al singolo palo)
 - **Facility:** se i dati si riferiscono all'intera struttura monitorata (per esempio, l'edificio, nel caso di Smart Building, lampioni di una via, per l'illuminazione stradale, ecc.)
 - **Facility Aggregation:** se i dati si riferiscono a un insieme di strutture aggregate fra loro (per esempio un gruppo di Smart Building monitorati da uno stesso Aggregatore)
 - **City:** se i dati si riferiscono a un'intera infrastruttura cittadina (per esempio l'intera rete di illuminazione pubblica di una città)
 - **Region:** se i dati si riferiscono a infrastrutture che si estendono oltre la singola città, a un'intera zona/regione geografica (per esempio, impianti di depurazione delle acque)
 - **Country:** i dati si riferiscono a infrastrutture che si estendono all'intera nazione (per esempio la rete di trasporto dell'energia elettrica)
- Livello di aggregazione temporale, che può assumere i seguenti valori:
 - **Not applicable:** se il concetto di aggregazione temporale non è applicabile all'Urban Dataset che si sta definendo
 - **Static:** per Urban Dataset anagrafici, che definiscono le caratteristiche statiche di una struttura (per esempio localizzazione spaziale di un edificio)
 - **Instantaneous:** nel caso di dati istantanei (per esempio temperatura istantanea misurata da un sensore)
 - **Average:** per valori medi (per esempio il consumo medio di elettricità in un determinato intervallo di tempo, in un edificio)

- **Total:** valori totali (per esempio il consumo totale medio di elettricità in un determinato intervallo di tempo, in un edificio)
- **Forecast:** valori previsti e quindi futuri (per esempio previsioni meteo)
- Categoria e sottocategoria delle principali applicazioni coinvolte. Viene utilizzata la categorizzazione definita insieme al NIST, nell’iniziativa “IES City”¹²:

Tabella 5. Categorie e sottocategorie di applicazioni, in base al lavoro fatto in IES-City

Categorie:	Sotto-categorie
Built environment	Smart Home
	Smart Building
	Land use and management
Water and wastewater	Water collection and management
	Water distribution
	Water consumption
	Wastewater management
Waste	Citizens engagement
	Collection and segregation
	Waste disposal
Energy	Energy supply
	Energy transmission and distribution
	Energy demand
Transportation	Travel demand/consumption
	Traffic management
	Surveillance
Education	Learning outcomes
	Learning and teaching
	Service management
Health	Health care systems
	Health care delivery
	Communication
Socio-economic development	E-Governance
	Social Innovation and Inclusion
	Economy and Business
Public safety, policy & Emergency Response	City surveillance and crime prevention
	Communication

¹² <https://pages.nist.gov/smartcitiesarchitecture/>

Il modulo di inserimento dati è mostrato nella figura sottostante:

CATEGORIZZAZIONE DELL'URBAN DATASET		
	Valore	Descrizione
Urban Dataset		<i>Inserire il nome</i>
Aggregazione spaziale		<i>Livello di aggregazione spaziale dell'Urban Dataset</i>
Aggregazione temporale		<i>Livello di aggregazione temporale dell'Urban Dataset</i>
Categoria		<i>Categoria principale di applicazioni coinvolte</i>
Sottocategoria		<i>Sottocategoria principale di applicazioni coinvolte</i>

Figura 16- Modulo di inserimento Urban Dataset: categorizzazione

Nel secondo riquadro (**contestualizzazione**) sono previsti una serie di campi che rappresentano proprietà comuni a tutti gli Urban Dataset (Timestamp di generazione, Timezone a cui si riferiscono tutti i timestamp dell'Urban Dataset, Coordinate del centro geometrico del sistema che produce gli Urban Dataset¹³, identificatore univoco del sistema che ha prodotto i dati). Tali campi non possono essere modificati e sono riportati nella scheda solo per rendere il proponente consapevole della presenza di questi dati.

CONTESTUALIZZAZIONE DELL'URBAN DATASET <i>[Dati obbligatori: sempre presenti]</i>				
	Unità di Misura	Raccomandazioni	Tipo	Descrizione
Timestamp	Adimensionale	Esprimere il timestamp come: gg-mm-aaaa hh:mm:ss	Date-Time	Tempo di generazione dell'Urban Dataset
Timezone	Adimensionale	Esprimere il timestamp come offset, espresso come + oppure meno il numero di ore e minuti di distanza dall'ora UTC: +/-hh:mm (ISO 8601)	String	Fuso orario di riferimento per tutti i timestamp del messaggio
Coordinate	Adimensionale	Esprimere come WGS84 (World Geodetic System)	double	Coordinate del centro geometrico del sistema che produce gli Urban Dataset
Producer	Adimensionale		String	Identificatore univoco del sistema che ha prodotto i dati

Figura 17. Modulo d'inserimento Urban Dataset: contestualizzazione

L'ultima parte del modulo (**proprietà**) si divide a sua volta in due riquadri.

Il primo riporta una serie di proprietà comuni agli Urban Dataset (timestamp della riga, coordinate del sistema che ha generato lo specifico set di dati riportati nella riga e periodo a cui fanno riferimento i valori). L'unica modifica che può essere effettuata in tale scheda è la scelta di inserire o meno timestamp e coordinate (entrambi opzionali a questo livello).

¹³ Per esempio, nel caso di una rete di Smart Building, sarebbe il centro geometrico degli insiemi di edifici

PROPRIETA' DELL'URBAN DATASET (Qui vanno definite le righe dell'UD)

Proprietà comuni	Unità di Misura	Raccomandazioni	Formati	Descrizione	Esempio	Inserire?
Timestamp	Date-Time	Esprimere il timestamp come: gg-mm-aaaa hh:mm:ss	String	Tempo di generazione della "riga" (singolo set set di valori) [opzionale]	26-09-2017 00:00:00	
Coordinate	Adimensionale (Coordinate)	Esprimere come WGS84 (World Geodetic System)	String	Coordinate del centro geometrico del sistema che ha prodotto la "riga" (singolo set set di valori) [opzionale]	42.04 ; 12.30	
Start Period	Date-Time	Esprimere il timestamp come: gg-mm-aaaa hh:mm:ss	String	Data di inizio del rilevamento dei dati	25-09-2017 00:01:00	Obbligatoria
End Period	Date-Time	Esprimere il timestamp come: gg-mm-aaaa hh:mm:ss	String	Data di fine del rilevamento dei dati (se dati "lenti", sarà uguale al valore di start)	26-09-2017 00:00:00	Obbligatoria

Figura 18. Prima parte del modulo relativo alle proprietà: proprietà comuni

Il secondo permette l'inserimento delle proprietà specifiche che caratterizzano l'Urban Dataset. I campi da riempire sono:

- Nome della proprietà
- Unità di misura
- Raccomandazioni: qui devono essere inserite raccomandazioni specifiche, relative alla proprietà, quali il riferimento a uno specifico standard per rappresentarli (per es. il WGS84 per le coordinate) o l'uso di liste di codici (nel caso di formato enumeration). Il contenuto di tali liste sarà specificato nei fogli successivi
- Formato dati: scelto (fra quelli standard, quali String, Int, Enumeration ecc.)
- Descrizione
- Esempio
- Annidata: il modello dei dati prevede la possibilità di definire proprietà annidate. Se la proprietà è annidata, qui andrà inserito il nome delle proprietà che si annidano qui dentro (tali nomi devono corrispondere ai nomi altre proprietà definite nella tabella.
- Esiste già nell'ontologia?: è un valore booleano che dice che quella particolarità è già stata definita (in un altro Urban Dataset precedente) e quindi può essere riutilizzata

Il numero di righe predisposte per l'inserimento di nuove proprietà, è limitato a 23, ma può essere aumentato (togliendo la protezione al foglio, che non richiede password e copiando e inserendo altre righe di proprietà), se risulta necessario.

Proprietà specifici	Unità di Misura	Raccomandazioni	Formati	Descrizione	Esempio	Annidata	Esiste già nell'ontologia?
Nome proprietà 1							
Nome proprietà 2							
Nome proprietà 3							

Figura 19. Seconda parte del modulo relativo alle proprietà: proprietà specifiche

3. Liste di codici (code-list)

L'ultima pagina del foglio di calcolo permette di inserire una lista di codici, nel caso sia stata introdotta nella colonna raccomandazioni della pagina Urban Dataset, in corrispondenza di un tipo Enumeration.

All'inizio della pagina si inserisce nome della lista di codici e l'eventuale nome alla colonna codici (se necessario). Quindi è possibile elencare i codici ed esprimerne il significato in italiano e inglese.

Nel caso sia necessario definire più di una lista di codici, basterà copiare la pagina e ripeterla il numero di volte necessaria per esprimere tutte

3 CODE LIST		
<i>Nota: Nel caso sia necessaria più di una lista di codici, replicare questo foglio</i>		
Nome della Code-List:	inserire nome	
Nome generale della colonna Codice	inserire nome	<i>Es. CauseId</i>
Codice	Descrizione Italiana	Descrizione Inglese

Figura 20 – Parte del modulo relativo alle Code List