Agenzia nazionale per le nuove tecnologie,
l'energia e lo sviluppo economico sostenibile

MINISTERO DELLO SVILUPPO ECONOMICO

Definizione dell'Ontologia degli Urban Dataset e di relativi strumenti Software

Michela Milano, Federico Chesani

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Report RdS/PAR2016/002

DEFINIZIONE DELL'ONTOLOGIA DEGLI URBAN DATASET E DI RELATIVI STRUMENTI SOFTWARE

Michela Milano, Federico Chesani (DISI – Università di Bologna)

Settembre 2017

Report Ricerca di Sistema Elettrico

Accordo di Programma Ministero dello Sviluppo Economico - ENEA
Piano Annuale di Realizzazione 2016
Area: Efficienza energetica e risparmio di energia negli usi finali elettrici e interazione con altri vettori energetici
Progetto: D.6 Sviluppo di un modello integrato di smart district urbano
Obiettivo: D6a Piattaforma ICT per la gestione di smart districts
Responsabile del Progetto: Claudia Meloni, ENEA

Il presente documento descrive le attività di ricerca svolte all'interno dell'Accordo di collaborazione Definizione dell'Ontologia degli UKAI e di relativi strumenti Software"
Responsabile scientifico ENEA: Nicola Gessa
Responsabile scientifico UNIBO: Michela Milano

Indice

Sommario

Il presente documento è uno dei risultati del secondo anno del progetto del MiSE per lo sviluppo di un modello integrato di Smart District Urbano nel Piano Annuale di Realizzazione ENEA 2015 sulla Ricerca di Sistema Elettrico. L'idea centrale è la realizzazione di una piattaforma software in grado di raccogliere dati e dataset di un distretto urbano al fine di creare servizi che sfruttando queste informazioni possano incrementare l'efficienza e la qualità della vita di una Smart City.

Una delle azioni chiave per raggiungere tale risultato è la descrizione delle informazioni provenienti dai diversi ambiti organizzativi di un distretto come un edificio, un singolo appartamento o una strada. Per facilitare e uniformare lo scambio di informazioni è stata ipotizzata la realizzazione di uno strumento in grado di descrivere le informazioni da scambiare secondo una semantica condivisa. A tal fine è stata proposta la realizzazione di una ontologia sulla base delle tecnologie del web semantico, che permetta una uniformità di ricerca dei dati e una organizzazione uniforme delle conoscenze al fine di semplificare lo scambio degli stessi.

Questo documento descrive il lavoro svolto per raggiungere tale risultato. Si è partiti dall'analisi di iniziative internazionali che si sono focalizzate su ambiti simili, si è proposto un modello base ed estendibile di ontologia con cui poter descrivere i dataset e la loro composizione, infine si sono proposte query di ricerca delle informazioni all'interno dell'ontologia. Tali query sono servite da base per sviluppare una libreria software che permettesse l'interrogazione della stessa ontologia.

1 Introduzione

La keyword “Smart City” viene spesso usata per indicare una delle misure prioritarie per affrontare la problematica energetico-ambientale propria di una città, ovvero il luogo in cui si concentra il maggiore consumo di risorse energetiche poiché vi si concentra l’attività insediativa, produttiva e di massimo impatto sull’ambiente.

L’approccio Smart City consiste, quindi, nel raggiungimento di traguardi di abbattimento dei consumi energetici (in primo luogo elettrici) molto più consistenti di quelli ottenuti finora attraverso strategie basate essenzialmente sulla sostituzione di componenti con altri “a maggiore efficienza energetica”. Il principio organizzativo da utilizzare è quello del “resource on demand”. Tale approccio richiede però una tecnologia di sistema avanzata che coinvolge e integra: i) una sensoristica urbana e sistemi di interazione per comprendere esattamente la necessità dell’utente, ii) sistemi di trasmissione e raccolta integrata dei dati (cloud urbani), iii) sistemi a elevata intelligenza, per analisi, diagnostica, elaborazione e ottimizzazione che fondono dati provenienti da diversi canali informativi, e infine iv) servizi urbani capaci di adattare la risposta.

Il principale obiettivo di una Smart City sta nella capacità di mettere insieme gli elementi energetico-ambientali e quelli di carattere sociale come la consapevolezza energetica, la partecipazione e coesione sociale e la qualità della vita attraverso l’uso di tecnologie e di applicazioni e sfruttando l’interconnessione tra reti, ottenendo lo sviluppo di “servizi innovativi multifunzionali” partendo dalle conoscenze messe a disposizione degli enti. Ovvero attraverso la pubblicazione di dataset contenenti informazioni utili allo sviluppo di servizi grazie all’integrazione di dati precedentemente separai e non pubblici. Quindi allo sviluppo di nuovi servizi dalla creazione di nuova conoscenza derivante dall’integrazione delle diverse sorgenti.

Il presente lavoro si inserisce nel progetto D.6 - SVILUPPO DI UN MODELLO INTEGRATO DI SMART DISTRICT URBANO. Il principale obiettivo del progetto D.6 consiste nello sviluppo di un modello di “distretto urbano intelligente” che coniughi aspetti tecnologici e aspetti sociali, finalizzati al miglioramento dei servizi erogabili ai cittadini in quanto più efficienti dal punto di vista energetico e funzionale.

Nel distretto è prevista infatti l’implementazione di tecnologie e metodologie per la raccolta e distribuzione di informazioni per garantire questo approccio. La soluzione proposta prevede inoltre una combinazione tra tecnologie, modelli di business e coinvolgimento dei cittadini in un approccio innovativo di rigenerazione urbana.

L’attività si focalizza sullo sviluppo integrato di infrastrutture pubbliche urbane, sistemi per la modellazione e gestione della rete energetica del distretto (Smart District), sistemi centralizzati per l’analisi dei dati provenienti dalle abitazioni ed edifici con feedback all’utente per orientarlo a un uso consapevole (Smart home e Smart building) e sistemi di supporto alle decisioni per la valutazione del rischio del patrimonio edilizio e delle infrastrutture.

Obiettivo finale dell’attività è lo sviluppo di un modello di Smart District come distretto urbano intelligente in cui tutti i servizi di quartiere siano gestiti in maniera ottimale, sinergica e interoperabile. Grazie alla definizione e utilizzo di specifiche standard e tecnologie open, si agevolerà la replicabilità dei modelli sviluppati come primo step di una roadmap per la realizzazione delle Smart City; di fatto si realizzerà uno strumento a servizio delle amministrazioni locali e dei cittadini per evitare il locked-in dei vendors.

Uno degli obiettivi del progetto è lo sviluppo della Smart Platform in grado di raccogliere e integrare tra loro i dati dai diversi ambiti applicativi: Building Network, Lighting, Smart Home Network, sicurezza delle infrastrutture e coinvolgimento dei cittadini. La Piattaforma ICT deve essere interoperabile e aperta per la gestione delle infrastrutture fisiche di distretto attraverso l’integrazione di applicazioni verticali relative ai servizi di distretto. La Piattaforma ICT, denominata Smart Platform, è un livello software orizzontale, trasversale a tutte le applicazioni verticali da cui riceve i dati. Tali dati devono essere elaborati, forniti e resi fruibili dai diversi attori che interagiscono con il distretto (gestori, amministratori comunali, utenti).

All'interno di questo task si sviluppa il lavoro condotto dal Dipartimento di Informatica – Scienza e Ingegneria (DISI) dell'Università di Bologna. Il lavoro svolto consiste nell'analisi dei dati ed elaborazione delle informazioni per poter definire uno strato di interoperabilità semantica che ha come scopo quello di semplificare l'implementazione di algoritmi intelligenti facilitando la selezione e aggregazione dei dati provenienti dai diversi ambiti applicativi.

Dal momento che i dati che confluiscono sulla piattaforma possono provenire da fornitori diversi che gestiscono parti diverse dell'infrastruttura e di raccogliere informazioni provenienti da varie attività della città o del distretto e depositati in luoghi diversi, è stato previsto l'uso di uno strato semantico di interoperabilità basato su di un'ontologia. Questo è un requisito fondamentale per evitare di legare le amministrazioni cittadine, utenti ideali della piattaforma ICT di distretto, a tecnologie proprietarie di singoli fornitori. Come abbiamo detto, una Smart City può definirsi tale se gestisce e integra efficacemente informazioni che provengono da diversi ambiti applicativi della città. La combinazione di queste informazioni è spesso limitata dal modo in cui sono rappresentate tali informazioni e ai concetti a cui si riferiscono non necessariamente interpretati in maniera univoca.

Lo scopo di un'ontologia è proprio quello di appianare tali differenze al fine di facilitare l'analisi delle informazioni permettendo una più facile elaborazione automatica, e in questo caso, al fine di progettare un'infrastruttura che limiti i tempi di sviluppo e al tempo stesso faciliti il lavoro di estensione dell'ontologia stessa per favorire una implementazione futura di nuovi servizi da integrare nella piattaforma della Smart City.

Nel seguito del documento sono descritte le attività svolte. Nel capitolo 2 verranno analizzati i modelli di dataset utilizzati per la distribuzione delle informazioni sulla piattaforma. Nel capitolo 3 si analizzeranno alcuni progetti esistenti che si occupano di modellare e implementare concetti legati al mondo Smart City e si analizzeranno i risultati di alcuni di questi progetti. Nel capitolo 4 verrà presentata l'ontologia definita a partire dal lavoro svolto nel precedente anno di progetto. Nei capitoli 5 e 6 verranno introdotte le operazioni di interrogazione definite per accedere ai dati contenuti nell'ontologia assieme alla descrizione della libreria software definita per integrare più facilmente l'ontologia in altri prodotti software. Nell'ultimo capitolo verranno tratte le conclusioni del lavoro svolto.

2 Analisi degli Urban Dataset forniti da ENEA

Con Urban Dataset ci si riferisce a collezioni di dati (che possono andare da insiemi di dati puntuali prelevati dai sensori, eventualmente aggregati dall'applicazione verticale di dominio, fino a quelli sintetizzati sotto forma di indicatori) che raccolgono informazioni di una città. Come ad esempio il numero di parcheggi liberi/occupati in una particolare area o la presenza di persone in un edificio o il consumo energetico di un appartamento.

Di conseguenza, gli Urban Dataset non sono altro che il frutto di una elaborazione (proiezione, astrazione, aggregazione, validazione) del contenuto informativo raccolto dai sensori del particolare contesto applicativo. L'elaborazione del contenuto ha lo scopo di:

- ottenere dati derivati;
- integrare le osservazioni con ulteriori fonti complementari;
- realizzare estensioni spaziali o proiezioni temporali;
- verificare/validare la qualità dei dati allo scopo di migliorarne l'affidabilità.

La misura e osservazione consiste nell'identificazione sistematica e nella rappresentazione organica e accessibile delle seguenti categorie di dati:

- caratteristiche statiche delle infrastrutture di raccolta dati (es.: geometria, posizione, vincoli di operatività, ecc.);
- caratteristiche dinamiche delle infrastrutture (es.: stato di salute, durata dell'operatività, ecc.);
- dati raccolti dalla rete di sensori (es.: dati grezzi come singolo passaggio di un veicolo con timestamp, tipologia dei veicoli inquadrati in una telecamera, assorbimento istantaneo di corrente rilevato da uno smart meter posizionato a monte di un gruppo di lampioni di illuminazione pubblica, ecc.).

Dal momento che i dati che confluiscono sulla piattaforma possono provenire da fornitori diversi che gestiscono parti diverse dell'infrastruttura e raccolgono informazioni provenienti da varie attività della città o del distretto, è stato previsto l'uso di uno strato semantico di interoperabilità basato su di un'ontologia. Questo è un requisito fondamentale per evitare di legare le amministrazioni cittadine, utenti ideali della piattaforma ICT di distretto, a tecnologie proprietarie di singoli fornitori. Come abbiamo detto, una Smart City può definirsi tale se gestisce e integra efficacemente informazioni che provengono da diversi ambiti applicativi della città. La combinazione di queste informazioni è spesso limitata dal modo in cui sono rappresentate tali informazioni e ai concetti a cui si riferiscono non necessariamente interpretati in maniera univoca.

Lo scopo di un'ontologia è proprio quello di appianare tali differenze al fine di facilitare l'analisi delle informazioni permettendo una più facile elaborazione automatica, e in questo caso, al fine di progettare un'infrastruttura che limiti i tempi di sviluppo e al tempo stesso faciliti il lavoro di estensione dell'ontologia stessa per favorire una implementazione futura di nuovi servizi da integrare nella piattaforma della Smart City.

Per raggiungere l'obiettivo di definire un'ontologia organica e funzionale, si è partiti dall'analisi di alcuni Urban Dataset in corso di definizione da parte di ENEA e dei formati di trasmissione. Nel corso del capitolo si provvederà a descrivere le loro caratteristiche e a fornire alcuni esempi.

2.1 Urban Dataset e informazioni associate

Per definire l'ontologia si è partiti da alcuni esempi di Urban Dataset forniti da ENEA e derivati dalle loro analisi a riguardo. In particolare sono stati analizzati i seguenti casi:

- *Smart Building Anomalies*: anomalie relative a una particolare causa (nel caso specifico riguardava la presenza di luci accese in assenza di persone in un edificio monitorato su di un periodo di tempo definito);

- *Energia Attiva F1*: lettura del valore di consumo di un contatore elettrico per una particolare fascia di consumo, la fascia F1 appunto;
- *Parking Monitoring Street Period*: numero medio di parcheggi liberi in una data via in un intervallo di tempo definito;
- *Evento sismico*: rilevamento delle informazioni di un evento sismico.

Come si può notare, le informazioni monitorate tramite gli Urban Dataset sono molto varie sia per ambito di applicazione che per la natura delle informazioni. Nonostante la loro varietà, alcune informazioni utili per contestualizzare l'Urban Dataset sono necessariamente comuni. Le altre informazioni, che riguardano caratteristiche di ciascuno specifico Urban Dataset, sono comunque strutturate in maniera comune in modo da mantenere una struttura uniforme per una più semplice comprensione delle informazioni sia da parte di in essere umano che in maniera automatizzata.

Le informazioni comuni ai diversi Urban Dataset riguardano i dati sul produttore delle informazioni, dati relativi alle informazioni contenute nell'Urban Dataset, le coordinate spaziali per associare il dato raccolto a uno specifico punto all'interno dell'area cittadina su ci si sta effettuando la raccolta dati.

Le informazioni specifiche di ogni singolo Urban Dataset riguardano i dati che effettivamente vengono raccolti sul campo e sono quindi specifiche per ogni Urban Dataset. In particolare, vengono definite, per ognuno di essi, il nome, la descrizione, il formato dei dati e l'unità di misura usata nel rilevamento.

Di seguito sono analizzate le caratteristiche del formato usato per la trasmissione delle informazioni allo scopo di capirne meglio la struttura e infine sono mostrati alcuni esempi reali.

2.2 *Il formato usato per la trasmissione degli Urban Dataset*

ENEA prevede di utilizzare formati XML e JSON per la trasmissione dei dati, entrambi basati su un modello dati di riferimento. Per comprendere come organizzare l'ontologia partiremo dall'analizzare le informazioni presenti in esso.

Il modello dati astratto definito per l'Urban Dataset si compone delle seguenti parti:

- ***Specification***: contenente le informazioni che descrivono l'Urban Dataset utilizzato (ad es. il riferimento alla specifica a cui aderisce e le grandezze che lo compongono);
- ***Context:*** fornisce le informazioni che contestualizzano i valori trasmessi (ad es. il fuso orario dei timestamp);
- ***Values***: dati rilevati sulle grandezze che compongono l'Urban Dataset, raggruppati in righe.

Elemento	Descrizione	Occorrenze
Specification	Proprietà specifiche dell'Urban Dataset.	**1..1**
Context	Informazioni di contesto	
Values	Grandezze che compongono l'Urban Dataset.	**1..1**
Line	Singolo gruppo di grandezze.	**1..n**

Il modello astratto è rappresentato attraverso tabelle composte dalle seguenti colonne:

- Elemento: etichetta scelta per un'informazione contenuta nell'Urban Dataset;
- Descrizione: descrizione testuale dell'informazione;
- Occorrenze: numero di ripetizioni ammesse (cardinalità) per l'informazione e indicatore di obbligatorietà quando valore minimo >0;

- Tipo: tipo di dato ammesso per l'informazione (ad es. intero, stringa, ...);

Il colore delle righe delle tabelle indica il livello di aggregazione delle informazioni:

- riga con sfondo bianco: indica una informazione elementare (es.: "data", "temperatura", ...) o, se il nome dell'elemento è preceduto dal simbolo @, una informazione che qualifica un'altra informazione elementare;
- riga con sfondo grigio: indica una informazione aggregata, composta da più informazioni elementari (es.: "periodo", "source", ...).

Si consideri, ad esempio, il seguente frammento estratto da una riga del blocco *Context*:

coordinates		Coordinate wgs84 dell'epicentro della rete applicativa su cui sono stati rilevati i valori trasmessi.	0..1
	@format	Formato wgs84 in cui sono espresse le coordinate (opzioni possibili: 'GMS', 'GM', 'G').	0..1
	longitude	Longitudine.	1..1
	latitude	Latitudine.	1..1
	altitude	Altitudine.	0..1

La sua interpretazione è la seguente: il blocco "Context" può contenere un elemento "coordinates" che contiene i dati che identificano l'epicentro della rete applicativa su cui sono stati rilevati i valori trasmessi; questo è opzionale ("coordinates " ha cardinalità minima uguale a 0) e non può essercene più di uno (massima cardinalità uguale a 1) . I dati che occorre indicare sono:

- opzionalmente, il formato wgs84 in cui sono espresse le coordinate
- longitudine e latitudine (obbligatori poiché "longitude" e "latitude" hanno cardinalità minima=1)
- opzionalmente, l'altitudine.

Il blocco **Specification** dell'Urban Dataset è composto dalle informazioni indicate nella seguente tabella:

Elemento		Descrizione	Occorr.
specificationReferences		Insieme di riferimenti che consentono l'identificazione della specifica dell'Urban Dataset utilizzato.	1..1
	@version	Versione della specifica.	0..1
	id	Codice identificativo della specifica.	1..1
	@schemeID	Identificativo dello schema di identificazione secondo il quale è stato definito l'identificatore.	0..1
	name	Nome associato all' Urban Dataset.	1..1
	uri	URI che identifica la specifica (può essere un urn indicate il dominio della specifica, o un url).	1..1

propertyDefinition	Questo elemento consente di inserire, all'interno del messaggio stesso, la descrizione di una delle grandezze che compongono l' Urban Dataset. *Per la struttura del blocco, si veda la sezione propertyDefinition.* *Il blocco deve essere ripetuto per ciascuna grandezza che compone l' Urban Dataset.*	0..n

Come evidenziato precedentemente, da questa tabella si evince, quindi, quali sono le proprietà presenti per ogni Urban Dataset e come dovranno essere specificate le informazioni raccolte tramite il singolo Urban Dataset. In particolare ciascuna proprietà specifica deve essere definita attraverso la descrizione proposta dal gruppo propertyDefinition e quindi definire i campi richiesti.

Il modello dati prevede la possibilità di definire sia **grandezze elementari** (ad esempio: magnitudo di un evento sismico, latitudine, longitudine, consumo elettrico di una unità abitativa, consumo termico di una unità abitativa), sia **grandezze aggregate**, ovvero grandezze composte da più grandezze elementari (ad esempio: epicentro di un evento sismico, che è composto da latitudine e longitudine, o consumo totale di una unità abitativa, che è composto da consumo elettrico e consumo termico).

La definizione di una **grandezza elementare** deve essere data dalle seguenti informazioni:

propertyDefinition	Struttura del blocco in caso di **grandezza elementare**.	
propertyName	Nome della grandezza.	1..1
propertyDescription	Descrizione della grandezza.	0..1
dataType	Tipo di dato con cui viene espresso il valore della grandezza.	1..1
unitOfMeasure	Unità di misura in cui è espresso il valore.	0..1

La definizione di una **grandezza aggregata** deve fornire le seguenti informazioni:

propertyDefinition	Struttura del blocco in caso di **grandezza aggregata**.	0..n
propertyName	Nome della grandezza.	1..1
propertyDescription	Descrizione della grandezza.	0..1
subPropertyRef	Riferimento (nome) di una grandezza che compone la grandezza definita in questo blocco. *Anche la grandezza riferita deve essere definita, seguendo la struttura propertyDefinition.*	2..n

Il blocco **Context** contiene informazioni generali di contesto non specifici per i diversi Urban Dataset:

Elemento	Descrizione	Occorr.
timeZone	Fuso orario di qualsiasi marca temporale presente nel documento.	1..1
timestamp	Istante di generazione di questo Urban Dataset (ovvero il momento in cui i dati sono stati aggregati).	1..1
coordinates	Coordinate wgs84 del centro geometrico della rete applicativa su cui sono stati rilevati i valori trasmessi.	1..1
@format	Formato wgs84 in cui sono espresse le coordinate (opzioni possibili: 'GMS', 'GM', 'G').	0..1
longitude	Longitudine.	1..1
latitude	Latitudine.	1..1
altitude	Altitudine.	0..1
language	Lingua in cui sono espressi i campi testo dell'istanza.	0..1
producer	Dati del sistema (Piattaforma/Soluzione) che ha prodotto i dati rilevati.	1..1
id	Identificatore univoco del sistema.	1..1
@schemeID	Identificativo dello schema di identificazione secondo il quale è stato definito l'identificatore.	0..1

In particolare si può notare come sia necessario sempre indentificare un Urban Dataset con l'informazione di geolocalizzazione dei valori.

Il blocco **Values** è composto di una o più righe dove a ognuna riga corrisponde a una combinazione dei valori associati da un Urban Dataset in un particolare momento. Questo vuol dire che per ogni riga dovranno essere presenti tutti i gli elementi definiti nel campo propertyDefinition. La presenza di più righe vuol semplicemente dire che si stanno trasferendo più istanze dello stesso Urban Dataset. La possibilità di avere più righe all'interno di questo file ha il solo fine di poter trasmettere più dati contemporaneamente me non ha un valore descrittivo, motivo per cui esula dagli obiettivi dell'ontologia. Le righe, quindi le istanze dell'Urban Dataset, devono contenere anche informazioni diverse da quello definite in propertyDefinition e che sono relative alla singola istanza di Urban Dataset. La definizione è presente nella seguente tabella:

Elemento	Descrizione	Occorr.
id	Numero riga.	0..1

description		Descrizione testuale dei valori rilevati.	0..1
timestamp		Data e ora (marca temporale) di generazione dei dati riportati in questa riga.	0..1
coordinates		Coordinate wgs84 dell'oggetto a cui sono riferiti i dati rilevati indicati in questa riga. *Se questo elemento è presente, specifica ulteriormente la posizione del dato rispetto alle coordinates del blocco Context.*	0..1
	@format	Formato wgs84 in cui sono espresse le coordinate (opzioni possibili: 'GMS', 'GM', 'G').	0..1
	longitude	Longitudine.	1..1
	latitude	Latitudine.	1..1
	altitude	Altitudine.	0..1
period		Periodo durante il quale sono stati rilevati i dati. *Nel caso di grandezza istantanea, indicare solo start_timestamp e end_timestamp coincidono.*	0..1
	start_ timestamp	Marca temporale indicante l'inizio del periodo.	1..1
	end_ timestamp	Marca temporale indicante la fine del periodo.	1..1
property		Grandezza che compone l'Urban Dataset. *Per la struttura del blocco, si veda la sezione property.* *Il blocco deve essere ripetuto per ciascuna grandezza che compone l'Urban Dataset.*	1..n

Il modello dati stabilisce come definire sia i valori delle grandezze **elementari**, sia i valori delle grandezze **aggregate**.

Una **grandezza elementare** deve essere espressa conformemente alla seguente struttura:

property	Struttura del blocco in caso di **grandezza elementare**.	

	name	Nome identificativo della grandezza rilevata.	1..1
	value	Valore rilevato.	1..1

Una **grandezza aggregata** deve essere espressa conformemente alla seguente struttura:

property		Struttura del blocco in caso di **grandezza aggregata**.	
	name	Nome identificativo della grandezza rilevata.	1..1
	property	Grandezza elementare che compone la grandezza aggregata.	2..n
	name	Nome identificativo della grandezza elementare.	1..1
	value	Valore rilevato.	1..1

Nella sezione seguente sono mostrati alcuni esempi di rappresentazione dei dati secondo questo schema.

2.3 Analisi di alcuni esempi

In questa sezione verranno osservati alcuni casi di esempio basati su esempi di raccolta dati dal campo o elaborati da dispositivi sul campo e organizzati come Urban Dataset.

Il primo esempio presentato mostra un ipotetico invio di informazioni di anomalia relativo a un edificio. Come si può notare all'interno del tag *specification* sono presenti i campi:

- specificationRef: usato per fornire una descrizione di base per l'Urban Dataset;
- propertyDefinition: tanti campi di questo tipo quanti sono i singoli dati che compongono questo specifico Urban Dataset

Ognuno dei *propertyDefinition* ha a sua volta quattro campi:

- propertyName: indicante il nome associato a quella proprietà;
- propertyDescription: dove è definita una descrizione testuale della proprietà;
- dataType: tipo di dato usato per rappresentare l'informazione;
- unitOfMeasure: unità di misura del dato (se si tratta di un dato numerico a cui è associata una unità di misura)

```
<UrbanDataset xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="smartcityplatform:enea:information:xml:schemas:main:urbandat
aset ../../SCPS%20Formati%20UKAI/scps-urbandataset-schema-1.0.xsd"
    xmlns="smartcityplatform:enea:information:xml:schemas:main:urbandataset">
    <specification>
        <id>UK000000</id>
        <name>Smart Building Anomalies</name>
        <uri>http://ontologia.esempio.it/UD123456789</uri>
        <properties>
            <propertyDefinition>
                <propertyName>BuildingID</propertyName>
                <propertyDescription>Identificatore del palazzo monitorato e
generatore di anomalie.
            </propertyDescription>
```

```
                <dataType>string</dataType>
                <unitOfMeasure>adimensionale</unitOfMeasure>
            </propertyDefinition>
            <propertyDefinition>
                <propertyName>BuildingName</propertyName>
                <propertyDescription>Etichetta associata al palazzo monitorato e
generatore di anomalie.
            </propertyDescription>
                <dataType>string</dataType>
                <unitOfMeasure>adimensionale</unitOfMeasure>
            </propertyDefinition>
            <propertyDefinition>
                <propertyName>CauseID</propertyName>
                <propertyDescription>Causa a cui si riferiscono le anomalie
individuate dalla piattaforma smart building, espressa come codice.
            </propertyDescription>
                <dataType>int</dataType>
                <unitOfMeasure>adimensionale</unitOfMeasure>
            </propertyDefinition>
            <propertyDefinition>
                <propertyName>CauseDescription</propertyName>
                <propertyDescription>Causa a cui si riferiscono le anomalie
individuate dalla piattaforma smart building, espressa come testo.
            </propertyDescription>
                <dataType>int</dataType>
                <unitOfMeasure>adimensionale</unitOfMeasure>
            </propertyDefinition>
            <propertyDefinition>
                <propertyName>HighAnomalies</propertyName>
                <propertyDescription>Numero delle anomalie individuate con
priorità high.
            </propertyDescription>
                <dataType>int</dataType>
                <unitOfMeasure>adimensionale</unitOfMeasure>
            </propertyDefinition>
            <propertyDefinition>
                <propertyName>MeanAnomalies</propertyName>
                <propertyDescription>Numero delle anomalie individuate con
priorità mean.
            </propertyDescription>
                <dataType>int</dataType>
                <unitOfMeasure>adimensionale</unitOfMeasure>
            </propertyDefinition>
            <propertyDefinition>
                <propertyName>LowAnomalies</propertyName>
                <propertyDescription>Numero delle anomalie individuate con
priorità low.
            </propertyDescription>
                <dataType>int</dataType>
                <unitOfMeasure>adimensionale</unitOfMeasure>
            </propertyDefinition>
        </properties>
    </specification>
```

Come mostrato in questo esempio, i dati presenti fanno riferimento a una serie di anomalie che si sono verificate all'interno di uno specifico edificio; l'edificio è, a sua volta, identificato attraverso una di queste proprietà dell'Urban Dataset. Il tipo di anomalia è identificato da un codice, la descrizione da una stringa di testo e le altre proprietà sono dei valori numerici che riportano il numero di anomalie con quelle caratteristiche.

In questo esempio le etichette relative alle anomalie (low, high e mean) fanno riferimento alla logica con cui sono individuate all'interno del sistema di monitoraggio. In particolare viene adottata una logica fuzzy per tenere conto delle incertezze nella individuazione di anomalie dovute alla trasformazione di un input continuo (come può essere l'assorbimento energetico di un edificio) a un output discreto (come appunto la presenza o meno di un'anomalia). La trattazione della logica di funzionamento dell'elaborazione dei dati non è comunque parte di questo documento e si rimanda ad altri per la trattazione dettagliata della logica utilizzata a riguardo.

Passiamo ad analizzare la seconda parte dell'esempio, in cui sono presenti le istanze di questo particolare Urban Dataset. L'esempio continua di seguito:

```
<context>
    <timeZone>UTC</timeZone>
    <timestamp>2017-09-07T13:00:00</timestamp>
    <coordinates>
        <longitude>12.5</longitude>
        <latitude>41.9</latitude>
    </coordinates>
    <language>IT</language>
    <producer>
        <id schemeID="SCP">Smart Building Casaccia</id>
    </producer>
</context>
```

In questa seconda parte sono presenti le informazioni di contesto generiche per ogni Urban Dataset dove sono associati informazioni riguardanti la posizione, il produttore, il tempo e la lingua delle informazioni testuali.

L'ultima parte dell'esempio:

```
<values>
    <line id="1">
        <description>Anomalie negli ultimi tre anni.</description>
        <timestamp>2017-08-02T11:22:15</timestamp>
        <period>
            <start_ts>2014-08-02T00:00:00</start_ts>
            <end_ts>2017-08-02T00:00:00</end_ts>
        </period>
        <property name="BuildingID">
            <val>1</val>
        </property>
        <property name="BuildingName">
            <val>F40</val>
        </property>
        <property name="CauseID">
            <val>1</val>
        </property>
        <property name="CauseDescription">
            <val>Luci accese in assenza di persone</val>
        </property>
        <property name="HighAnomalies">
            <val>301</val>
        </property>
        <property name="MeanAnomalies">
            <val>115</val>
        </property>
        <property name="LowAnomalies">
```

```
                <val>280</val>
            </property>
        </line>
        <line id="2">
            <description>Anomalie negli ultimi tre anni.</description>
            <timestamp>2017-08-02T11:22:15</timestamp>
            <period>
                <start_ts>2014-08-02T00:00:00</start_ts>
                <end_ts>2017-08-02T00:00:00</end_ts>
            </period>
            <property name="BuildingID">
                <val>4</val>
            </property>
            <property name="BuildingName">
                <val>F66</val>
            </property>
            <property name="CauseID">
                <val>1</val>
            </property>
            <property name="CauseDescription">
                <val>Luci accese in assenza di persone</val>
            </property>
            <property name="HighAnomalies">
                <val>55</val>
            </property>
            <property name="MeanAnomalies">
                <val>54</val>
            </property>
            <property name="LowAnomalies">
                <val>88</val>
            </property>
        </line>
    </values>
</UrbanDataset>
```

In questo caso i tag *line* sono due a indicare che sono presenti due istanze dell'Urban Dataset in questo esempio. Altro dettaglio importante è la presenza di altre quattro proprietà non definite dalla *propertyDefinition*: timestamp, description, coordinates e period come definito dalle specifiche viste in precedenza nella descrizione del blocco Values e che sono relative alle specifiche istanze di questo particolare Urban Dataset e hanno lo scopo di aggiungere informazione di contesto alle istanze.

Di seguito presentiamo un nuovo esempio di Urban Dataset di altro tipo. Questo esempio descrive un Urban Dataset il cui scopo è informare del numero medio di parcheggi liberi in una specifica via. A esso sono associate informazioni sul produttore, la posizione e le specifiche proprietà che caratterizzano le istanze. Anche in questo caso, per rispettare le specifiche, sono presenti due blocchi principali identificati dai tag *specification*, *context* e *values*.

```
<UrbanDataset xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="smartcityplatform:enea:information:xml:schemas:main:urbandat
aset ../../SCPS%20Formati%20UKAI/scps-urbandataset-schema-1.0.xsd"
    xmlns="smartcityplatform:enea:information:xml:schemas:main:urbandataset">
    <specification>
        <id>UK00001</id>
        <name>Smart Building Records</name>
        <uri>http://ontologia.esempio.it/UD1</uri>
    <properties>
        <propertyDefinition>
```

```
            <propertyName>BuildingID</propertyName>
            <propertyDescription>Identificatore del palazzo monitorato e
generatore di anomalie.
        </propertyDescription>
            <dataType>string</dataType>
            <unitOfMeasure>adimensionale</unitOfMeasure>
        </propertyDefinition>
        <propertyDefinition>
            <propertyName>BuildingName</propertyName>
            <propertyDescription>Etichetta associata al palazzo monitorato e
generatore di anomalie.
        </propertyDescription>
            <dataType>string</dataType>
            <unitOfMeasure>adimensionale</unitOfMeasure>
        </propertyDefinition>
    </properties>
    </specification>
```

Questo esempio mostra il caso della raccolta di informazioni statiche, di tipo anagrafico, di edifici. Infatti i campi dell'Urban Dataset sono un identificativo dell'edificio e un nome associato.

A questi dati vanno aggiunti le informazioni di contesto come ad esempio la posizione dell'edificio stesso e il fornitore del dato:

```
    <context>
        <timeZone>UTC</timeZone>
        <timestamp>2017-09-07T13:00:00</timestamp>
        <coordinates>
            <longitude>12.5</longitude>
            <latitude>41.9</latitude>
        </coordinates>
        <language>IT</language>
        <producer>
            <id schemeID="SCP">Smart Building Casaccia</id>
        </producer>
    </context>
```

All'interno del campo *values* sono elencate le istanze e i valori ed esse associate e descritte nella *specification* sotto forma di diverse *line*. In ogni *line* sono elencate le diverse proprietà, assieme a i valori associati, definiti nella sezione *specification*, ovvero 'BuldingID' e 'BuildingName'.

```
     <values>
        <line id="1">
            <timestamp>20017-08-03T00:00:00</timestamp>
            <property name="BuildingID">
                <val>1</val>
            </property>
            <property name="BuildingName">
                <val>F40</val>
            </property>
        </line>
    </values>
</UrbanDataset>
```

Dai due esempi mostrati si può quindi provare a trarre qualche analisi più generale. Da quello che si vede, il dominio applicativo delle informazioni non influenza il modo con cui vengono rappresentate le informazioni, ovvero non c'è un legame tra dominio e modalità di rappresentazione. Questo permette di definire una struttura unica per la rappresentazione dei dati che favorisce sia l'inserimento che la ricerca

delle informazioni sulla base di conoscenza dal momento che non è necessario differenziare le operazioni sulla base degli ambiti applicativi.

Nel capitolo successivo ci soffermeremo ad analizzare le ontologie esistenti che si occupano di rappresentazione delle informazioni in ambito Smart City.

3 Ontologie per Smart City

L'architettura di riferimento ipotizzata per la Piattaforma ICT prevede, tra le altre cose, un approccio modulare e un set di specifiche aperte. Lo scopo di tali richieste è quello di fornire soluzioni replicabili e sostituibili. Per permettere che i dati prodotti possano essere interpretabili da processi che elaborano automaticamente le informazioni, è necessario che tali dati siano strutturati e definiti senza ambiguità. Per raggiungere tale risultato, l'uso di una ontologia che definisca tali concetti è ritenuto un passaggio fondamentale. Inoltre, diverse ontologie sono state definite per descrivere informazioni inerenti agli ambiti di una Smart City.

In questo capitolo discuteremo delle loro caratteristiche allo scopo di estrarre le caratteristiche desiderate per il nostro scopo.

3.1 Ontologie e rappresentazione delle informazioni

Col termine ontologia ci si riferisce a un sistema per descrivere il modo in cui diversi schemi vengono combinati in una struttura dati contenente tutte le entità rilevanti e le loro relazioni in un dominio. Un'ontologia per il Web è costituita da una tassonomia, da un insieme di relazioni fra gli elementi della tassonomia e da un insieme di regole di inferenza. La tassonomia definisce una classificazione gerarchica di oggetti. Classi, sottoclassi e relazioni fra entità sono strumenti molto potenti. Si può esprimere un grande numero di relazioni fra entità assegnando proprietà a classi e permettendo alle sottoclassi di ereditarle. Per esempio, uno studente può essere definito come un tipo di persona e una matricola può essere definita come applicabile solo a oggetti di tipo studente.

In questo modo è possibile combinare informazioni presenti su database diversi memorizzati con nomi diversi, ma che in realtà identificano la stessa categoria logica. Idealmente, una macchina, grazie all'utilizzo di un'ontologia comune, sarebbe in grado di scoprire e individuare questa uguaglianza di significati per qualsiasi database incontrasse.

Tuttavia possono sorgere problemi se si cerca di aggregare informazioni provenienti da sorgenti diverse che fanno uso di ontologie diverse. Questo problema può essere risolto attraverso l'uso di relazioni di equivalenza in una o entrambe le ontologie.

In ambito di Web Semantico, i concetti sono identificati univocamente con gli URI (Uniform Resource Indicator), mentre i vocabolari dei termini sono utilizzati per descrivere le relazioni tra le risorse. Queste descrizioni rappresentano lo schema dell'informazione e forniscono le indicazioni utilizzate dai Linked Data per rappresentare l'informazione. I vocabolari, a loro volta, sono descritti sotto forma di Linked Data.

Resource Description Framework (RDF) è una specifica del W3C rilasciata nel 1999. Progettato per l'elaborazione di metadati web, RDF fornisce una piattaforma di interoperabilità tra applicazioni web che scambiano informazioni comprensibili dalle macchine. Il framework facilita l'elaborazione automatica delle informazioni e può essere utilizzato in diverse aree di applicazione: nella ricerca di risorse (*resource discovery*), per migliorare le capacità dei motori di ricerca, per descrivere il contenuto e le relazioni disponibili in un particolare sito web, pagina o digital library, per facilitare condivisione e scambio di conoscenza tramite gli agenti intelligenti, nella valutazione del contenuto, nella descrizione di collezioni di pagine che rappresentano un singolo documento *logico*, nella descrizione del copyright di pagine web, per esprimere le preferenze di un utente, per la gestione della privacy di un sito.

RDF Schema (RDFS, noto anche come RDF vocabulary description language) è una estensione semantica per RDF introdotta con lo scopo di descrivere le ontologie. Fornisce un meccanismo di base per descrivere gruppi di risorse collegate tra loro e le relazioni tra le stesse risorse. È in grado di definire concetti e relazioni, e di definire vocabolari di termini per modellare specifici domini attraverso il linguaggio RDF stesso.

RDFS è basato su un set di classi e proprietà di default con cui è possibile specificare altre classi e proprietà e specificare, così, un vocabolario completo. Il vocabolario costruito deve definire tutti i possibili concetti e tutte le relazioni tra i concetti, con lo scopo di poter costruire una struttura adatta a descrivere correttamente l'ambito informativo che si vuole modellare.

RDF Schema è uno strumento che consente (in modo molto basilare) di definire vocabolari RDF, tuttavia in alcuni casi può essere utile utilizzare altri linguaggi che forniscono qualche capacità in più. Alcune di queste capacità sono state raccolte all'interno di Web Ontology Language (OWL). Si tratta di un altro importante pezzo per la costruzione del Web Semantico.

OWL è una raccomandazione W3C per la creazione di ontologie da utilizzare per di descrivere il significato semantico dei dati. OWL presenta strumenti più espressivi di RDFS, di cui ne costituisce una estensione, per esprimere le informazioni semantiche.

Tale formato è di fatto quello più utilizzato per la rappresentazione delle ontologie, le quali hanno avuto una forte spinta con la diffusione dei concetti di web semantico e dei formati che sono sopraggiunti in quel periodo sotto la spinta del W3C. Per questo motivo tutte le ontologie analizzate usano questo modello di rappresentazione così come il prodotto di questo lavoro è stato rappresentato secondo tale specifica.

Il lavoro che la Piattaforma ICT deve svolgere è quello di mediatore delle informazioni. In particolare, deve rendere efficiente lo scambio di informazioni tra i diversi contesti applicativi che si interfacciano con essa. In questo punto può sorgere un problema di interoperabilità. L'uso di differenti linguaggi, paradigmi e strumenti software causano limitazioni all'interoperabilità e al potenziale riuso del software. A lungo termine questo conduce alla perdita di tempo nel dover riscrivere il software. Infatti se i differenti contesti applicativi operano attraverso formati diversi, possono sopraggiungere problemi riguardanti l'estrapolazione di dati dai differenti formati. In una situazione del genere renderebbe necessario del lavoro aggiuntivo per le piattaforme dei diversi contesti applicativi al fine di poter interpretare i formati degli altri gestori. La situazione diventerebbe ancora più grave se il formato di un contesto dovesse cambiare a seguito del cambio di un gestore dei dispositivi del contesto.

Un modo per affrontare questo problema è attraverso la riduzione o l'eliminazione della confusione terminologica e con una conoscenza e una terminologia condivisa attraverso un framework unificante fra le diverse parti interagenti che funga da base per l'interoperabilità tra sistemi realizzati con differenti modelli, paradigmi e linguaggi di programmazione. Porterebbe benefici dal punto di vista dell'affidabilità, in quanto una rappresentazione formale renderebbe automatico il controllo della consistenza e quindi il software più affidabile. Una conoscenza condivisa può essere utile nel processo di identificazione dei requisiti e definizione di una specifica per sistemi IT; può essere utile anche per la definizione formale delle entità importanti e delle loro correlazioni. Questa rappresentazione formale può essere un componente riusabile in un sistema software [1].

Una ontologia può essere utilizzata come base comune, come framework unificante per risolvere i problemi descritti in precedenza. Essa giocherebbe un ruolo molto importante nella comunicazione fornendo disambiguazione delle definizioni dei termini utilizzati dentro i software. Un'ontologia fornisce una specifica dichiarativa del sistema software che permette di ragionare per cosa il sistema e progettato piuttosto per come il sistema supporta una funzionalità. Un'altra caratteristica importante è la sua riusabilità. Un'ontologia fornisce una libreria di classi per modellare problemi facile da riutilizzare. L'ontologia stessa può riutilizzare parti definite da altre ontologie e la sua diffusione renderebbe ancora più facile l'integrazione e la comparazione di dati prodotti, ad esempio, dalle piattaforme ICT di due diverse Smart City [1].

L'uso di un'ontologia favorisce l'interoperabilità e l'integrazione in ambienti aperti e grandi e permette la scalabilità dei servizi [2]. Ad esempio, nel caso in cui si sentisse la necessità di aggiungere un nuovo contesto applicativo, questo potrebbe essere integrato molto più facilmente semplicemente rispettando le specifiche dell'ontologia limitando i costi e la complessità che ci sarebbero stati altrimenti [3]. Un problema simile ci potrebbe essere nel caso si dovessero ricontrattare i dati prodotti dai fornitori che si occupano dei contesti. In questo caso la ridefinizione dei dati sarebbe molto più semplice. L'integrazione, nel resto della piattaforma, dei nuovi dati sarebbe molto più immediato perché la struttura non sarebbe cablata nel codice della piattaforma.

L'utilizzo di tecnologie standard per la definizione dell'ontologia, come quelle definite dal World Wide Web Consortium (W3C), permettono di avere disponibili tutta una serie di protocolli e linguaggi da usare per la

comunicazione. Tali strumenti facilitano lo sviluppo di nuove applicazioni, rendono immediata e automatizzabile la verifica di aderenza al formato di comunicazione e favoriscono la scoperta automatica della struttura dei dati e del significato degli stessi attraverso l'organizzazione dei concetti in una gerarchia e il collegamento con informazioni collegate, rendendo automatica anche la generazione della documentazione relativa.

Un ulteriore vantaggio che può apportare l'uso di un'ontologia è la possibilità di definire, in maniera semplice, regole per generare nuove informazioni in maniera autonoma con degli appositi strumenti software. Infatti esistono diversi meccanismi e linguaggi che permettono di definire regole per ottenere nuove informazioni a partire dai dati memorizzati nella base di conoscenza. Nel caso di una Smart City, sarebbe possibile, ad esempio, calcolare informazioni di consumo in una particolare zona. In dettaglio, ricevendo dai diversi Contesti Applicativi informazioni di consumo elettrico, si possono selezionare i dati relativi a un distretto in uno specifico intervallo temporale, ed eventualmente applicando in automatico regole di conversione delle unità di misura nel caso i dati ne facciano uso di diverse, al fine di valutare l'efficienza del distretto. Ad esempio, raggruppando le informazioni di consumo dei lampioni della smart lighting con quelle degli edifici dei verticali smart building e smart home con eventuale futura integrazione delle informazioni di consumo delle colonnine di ricarica per auto elettriche della città del domani. Tale valutazione potrebbe essere comparata con altre informazioni presenti in un database, eventualmente strutturato anch'esso secondo i formati del web semantico, dell'amministrazione cittadina al fine di normalizzare la valutazione dell'efficienza usando come riferimento il numero di abitanti della zona o il numero di edifici e abitazioni. Con l'obiettivo di definire degli indici di performance cittadini e comparare i risultati con i diversi distretti della città stessa.

Il vantaggio che offre l'ontologia in questa logica è proprio la semplificazione del processo di aggiunta di nuove logiche anche in seguito alla messa in opera dell'intero progetto dovuta alla loro natura dichiarativa e non essendo cablate nel codice della piattaforma. Le regole possono essere definite in qualunque momento futuro in base a necessità che possono presentarsi e che non sono necessariamente prevedibili. Inoltre possono essere modificate, riviste ed eliminate. Questo permette di venire in contro alla natura dinamica di una Smart City fornendo un indubbio vantaggio.

La ricerca sull'aspetto della definizione delle regole e dei loro meccanismi è ancora in una fase evolutiva piuttosto dinamica, l'attività di ricerca intensa in materia spinta anche dal W3C con lo scopo di integrare le tecnologie di base del web semantico con sistemi per la rappresentazione e la gestione di sistemi di regole che affrontino problemi e casi d'uso sempre di maggiore interesse per i progettisti software e in modo più efficiente, ci permette di essere ragionevolmente ottimisti sulla centralità che può avere in futuro un'ontologia all'interno di un sistema software come quello che ci si propone di realizzare all'interno di questo progetto.

3.2 Specifiche della Smart City Platform

Una Smart City è un prodotto molto complesso che comporta l'analisi e la convergenza di informazioni provenienti da ambiti molto diversi tra loro. Per quanto riguarda questo progetto, si sono ipotizzati diversi ambiti di azione, detti verticali. La classificazione che si è deciso di utilizzare per organizzare questi ambiti di azione è la stessa definita nell'iniziativa IES-City in una collaborazione tra ENEA e NIST e mostrata in tabella 1. Gli Urban Dataset, prodotti dai diversi fornitori che si interfacceranno con la piattaforma, saranno, oltre che relativi ai diversi ambiti, anche molti e diversi tra loro. Ci si aspetta, quindi, di ricevere una grande varietà e quantità di dati.

Sono stati, inoltre, previsti diversi casi d'uso per descrivere il comportamento della piattaforma ed ipotizzare il comportamento del sistema. I casi d'uso previsti sono di tre tipologie:

- Inter-Solution: tali casi d'uso descrivono il flusso dei dati da una Solution di un contesto applicativo verticale a un altro, attraverso la Smart City Platform. Ne è un esempio l'interazione tra il sistema di supporto alle decisioni per le infrastrutture critiche e gli Smart Building con lo scopo di permettere allo Smart Building di migliorare le logiche di gestione energetica sulla base delle condizioni meteo attuali e previste;

- One-Way: in cui fa da ricevente dei dati e non fa da tramite tra diversi verticali. Ad esempio per raccogliere informazioni sul consumo energetico dell'illuminazione pubblica;
- Particolari: situazioni di funzionamento non formalizzate dove verticali inviano dati generici, o per abilitare la comunicazione diretta tra due verticali.

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 & Em.Res.	City surveillance and crime prevention Communication

Tabella 1 – Categorizzazione delle applicazioni sulla base del lavoro svolto nell'ambito di IES-City

La piattaforma, quindi, agisce come tramite tra i diversi verticali garantendo anche diverse modalità e frequenza di raccolta. Ad esempio, l'invio dei dati può essere effettuato al momento della variazione delle informazioni, ad intervalli prefissati anche molto diversi a seconda dello scenario di funzionamento oppure su richiesta esplicita del sistema o di un operatore. Come può essere il caso della gestione energetica di un edificio in cui la frequenza di aggiornamento dei dati richiesta per le condizioni meteo correnti è maggiore di quella relativa alle previsioni meteo.

Alle informazioni che vengono scambiate sono associate molte altre informazioni statiche che le descrivono meglio. Per evitare di trasferire continuamente questa descrizione e per contenere una descrizione condivisa delle informazioni e del loro significato, si è previsto l'uso di una base ontologica che contiene questi riferimenti e che può essere interrogata per recuperarli quando necessario. In questo modo è possibile associare facilmente informazioni anagrafiche e di descrizione delle informazioni ai messaggi ed anche ridurre la dimensione delle informazioni scambiate sulla piattaforma.

Per effettuare la raccolta di informazioni si prevede di interfacciare i diversi verticali con un sistema di comunicazione che permetta l'invio, la memorizzazione e l'interrogazione dei dati. In questo modo, le informazioni dei diversi verticali possono essere condivise facilmente e utilizzate per combinarle in nuove informazioni da usare per analizzare il funzionamento della Smart City e, eventualmente, implementare politiche reattive e predittive per decidere quali azioni eseguire al fine di gestire situazioni critiche o non desiderate. Dal momento che si avrà a che fare con una consistente mole di dati, è stato previsto l'utilizzo di sistemi database molto performanti per la raccolta dei dati.

Un'alternativa sarebbe potuta essere quella di implementare uno store in formato RDF interrogabile dall'esterno e che venisse alimentato con i dati e le informazioni provenienti dai diversi verticali e organizzate secondo l'ontologia definita. Il database, così strutturato, avrebbe avuto lo scopo di fungere da punto di accesso unificato delle informazioni strutturate in modo organico. Questo sistema avrebbe permesso di ottenere facilmente e automaticamente le informazioni desiderate e di garantire l'integrazione con dati provenienti dai vari verticali su base semantica al fine di analizzare in modo automatico la struttura dei dati e recuperare le informazioni di loro interesse e fornire servizi avanzati unendo la conoscenza proveniente dai vari ambiti.

Tale soluzione, però, deve scontrarsi con le capacità di elaborazione garantiti dai sistemi che si basano su grafi ed eseguono interrogazioni SPARQL, che sono tendenzialmente minori rispetto ai sistemi SQL o anche altri NoSQL ottimizzati per grosse moli di dati e/o solo per semplici operazioni di accesso. Nel nostro caso, infatti, la velocità di accesso ai dati è uno dei fattori chiave per completare le richieste. Per questo motivo si è preferito scorporare la parte che si occupa di fornire e ricevere i dati da quella che ne fornisce una descrizione semantica. Per cui si è deciso di separare l'ontologia dal resto e fare in modo che l'endpoint SPARQL fornisca solo la descrizione semantica dei dati e serva da indice per gli stessi.

3.3 Progetti di integrazione di dati di Smart City

Durante la fase di analisi di questo anno si sono esplorate diversi progetti che, attraverso ontologie, cercano di raggiungere un risultato di integrazione e catalogazione dei dati. Lo scopo è di osservare lo stato dell'arte al fine di comprendere come muoversi in questo ambito. Di seguito sono descritte alcune delle soluzioni individuate.

3.3.1 Progetto DIMMER

Il progetto DIMMER [4] integra informazioni sulla topologia di edifici con dati real-time provenienti da sensori e da utenti per analizzare e correlare l'uso degli edifici e fornire informazioni sul comportamento degli stessi sul consumo energetico in tempo reale. Inoltre fornisce libero accesso a diverse informazioni connesse all'energia dell'edificio ad applicazioni client per elaborare informazioni su analisi dei costi, pianificazione e valutazione delle tariffe, identificazione dei guasti e manutenzione. Il risultato atteso dal progetto è una riduzione sia dei consumi che della CO2 emessa sostenendo delle policy di distribuzione dell'energia più efficienti, tenendo conto anche delle reali caratteristiche degli edifici e degli abitanti, e un più efficiente uso dell'energia e della rete di distribuzione basandosi sul comportamento delle persone e sulle loro abitudini.

Il progetto mira quindi a costruire un archivio digitale distribuito di un distretto o una città, in cui diversi attori possono pubblicare e usare informazioni per fornire servizi innovativi. Il progetto individua tre categorie di attori: gli amministratori, i gestori degli edifici e i gestori delle reti energetiche.

L'ontologia definita è molto semplice e definisce una relazione tra servizio fornito da un sensore, edificio e posizione fisica. In sostanza quello che fa è descrivere le funzionalità, gli oggetti e le persone coinvolti e l'ambiente in cui si opera. Sono stati anche modellati gli oggetti fisici in cui si opera per permettere ragionamenti sul risparmio energetico. Lo scopo principale è quello di descrivere in quale sistema sono memorizzate le informazioni e come recuperarle.

La DIMMER Systems Integration Ontology (SIO) è interrogabile attraverso SPARQL e per facilitare l'interrogazione automatica hanno deciso di definire tutti i concetti all'interno di un loro namespace e di linkare da lì altre ontologie.

In realtà l'ontologia principale è molto semplice e limitata. Consiste in poche relazioni e classi con lo scopo di fornire indicazioni sulle sorgenti di dato, ovvero dove si trovano i sensori, in quale edificio di quale distretto e che tipo di osservazione forniscono. Può essere usata come spunto per rappresentare questo tipo di relazioni.

3.3.2 Progetto City Protocol

Il progetto City Protocol [5] è un progetto collaborativo, condotto dalla città di Barcellona, il cui obiettivo è la creazione di un sistema comune per le città e sviluppare protocolli che permettano innovazione attraverso l'interazione tra diversi settori e ambiti della vita cittadina e con il fine di creare un'interconnessione tra le città. Tale ricerca viene intrapresa attraverso la definizione di una piattaforma di interoperabilità per poter affrontare in maniera olistica l'organizzazione e i problemi della città.

Il progetto prevede la definizione di una ontologia fondamentale [6] che definisca una struttura fondamentale della città attraverso la definizione di concetti di base e generali per le diverse tipologie di attività della città sia dal punto di vista infrastrutturale che sociale, economico e ambientale.

In particolare, l'ontologia definita in questo processo contiene i concetti di indicatori della città, che possono riferirsi a infrastrutture, e calcolati a partire da dati raccolti da sensori della città attraverso il sistema di interazione della città e definito come "Information Platform".

Il progetto prevede la definizione di ulteriori ontologie che si focalizzino su settori specifici. In particolare City Anatomy Indicators [7] e Open Sensor Platform [8]. Nella descrizione della Anatomy Ontology [9] è mostrata la struttura degli indicatori.

Tra le varie cose, l'ontologia definisce il concetto di indicatore associato a un'informazione misurabile, a una misura e unità di misura, e a uno scopo. Inoltre, l'indicatore può essere di diverso tipo come lo Structure_indicator riferito a informazioni legate al mondo fisico come le infrastrutture o gli edifici, oppure riferito ad ambiti economici, culturali e governativi.

Tale ontologia deve essere affiancata anche da altre che permettono di descrivere concetti importanti per descrivere la città e le informazioni raccolte. Tra queste troviamo ontologie capaci di descrivere luoghi geografici, i valori numerici usati per le diverse informazioni della città e concetti relativi allo scorrere del tempo. Nel documento "A Foundation Ontology for Global City Indicators" [10], adottato dal progetto City Protocol, sono descritte alcune ontologie atte a questo scopo tra cui Schema.org, che contiene concetti relativi a tipologie di luoghi come `sc:Country`, `sc:City` o `sc:GeoCoordinates` e `sc:GeoShape`, mentre Linkedgeodata.org contiene concetti collegati ad edifici e alla loro tipologia come `gd:neighborhood`, `gd:building`, `gd:hospital` o `gd:airport`. Il progetto GeoNames, invece, fornisce identificatori unici per più di dieci milioni di luoghi in giro per il mondo e realizzati come istanze della GeoNames Onology [11] la quale integra diversi tipi di ontologie tra chi quelle di Schema.org e Linkedgeodata.org.

Viene usata anche un'ontologia; Ontology of units of Measure [12] usata per descrivere le quantità numeriche usate nella misura dei dati rilevati; e PROV [13] che ha come scopo l'indicazione della provenienza di un dato. Affianco a questi concetti sono necessari anche concetti relativi al tempo. A questo scopo esiste OWL-Time [14] che descrive non solo gli istanti di tempo, ma anche le relazioni di ordinamento tra i diversi istanti.

Non ho trovato una definizione dell'ontologia sotto forma di owl che descriva in dettaglio la struttura e l'interazione con le altre ontologie che usa. La sua mancanza è un problema perché rende difficile linkare gli oggetti e l'esplorazione automatica. Inoltre strumenti di editing per le ontologie generano eccezioni nel momento in cui non riescono a importare le ontologie a cui si fa riferimento perché non sono presenti ai link indicati.

3.3.3 Km4City Ontology

Ontologia sviluppata dal DISIT dell'Università di Firenze [15] con lo scopo di integrare le informazioni su trasporti e mobilità con i dataset opendata messi a disposizione dal comune di Firenze a dalla regione Toscana. Essa è costituita da varie macroclassi:

1. Amministrazione: le cui classi principali sono PA, Municipality, Province, Region, Resolution.
2. Stradario: formata dalle classi Road, Node, RoadElement, AdminidtrativeRoad, Milestone, StreetNumber, RoadLink, Junction, Entry, EntryRule e Maneuver.
3. Punti di Interesse: comprende tutti i servizi, le attività, che possono essere utili al cittadino e che quindi quest'ultimo può aver necessità di ricercare e di raggiungere. Sono inoltre inclusi in questa macroclasse le Digital Location e gli eventi programmati (dati Real Time) del comune di Firenze
4. Trasporto Pubblico Locale: formata dalle classi TPLLine, Ride, Route, AVMRecord, RouteSection, BusStopForeast, Lot, BusStop, RouteLink, TPLJunction.
5. Sensoristica: ancora in via di sviluppo. Attualmente sono stati integrati nell'ontologia i dati raccolti da vari sensori installati lungo alcune strade di Firenze e dintorni, e quelli relativi ai posti liberi nei principali parcheggi dell'intera regione; presente anche la parte relativa agli eventi/emergenze.
6. Temporale: macroclasse che punta all'inserimento di concetti legati al tempo (istanti di tempo e intervalli di tempo) all'interno dell'ontologia, in modo da poter associare una linea temporale agli avvenimenti registrati e poter riuscire a fare previsioni.
7. Metadati: macroclasse di triple associate al context di ciascun dataset; tali triple raccolgono le informazioni relative a licenza del dataset, se il processo di inserimento ed elaborazione è automatizzato completamente, il formato della risorsa, una breve descrizione della risorsa e altre info sempre legate alla risorsa stessa e al suo processo di inserimento ed elaborazione.

Le parti relative allo stradario, al trasporto pubblico e ai punti di interesse sono le più dettagliate ed è possibile modellare anche informazioni come le manovre consentite in un particolare punto di una strada, o descrivere la struttura di una rete urbana e i collegamenti tra i vari elementi. Le caratteristiche dei punti di interesse e a ognuna è associata una forma geometrica.

Nella sezione sensoristica vengono descritti i sensori usati, la loro posizione e il tipo di informazione raccolta limitatamente alla situazione del traffico e dei parcheggi, e alle informazioni sul percorso degli autobus.

Dal nostro punto di vista, questa ontologia è troppo legata a uno schema in cui i sensori sono definiti puntualmente. L'architettura dell'ontologia che riguarda la raccolta dei dati si riferisce in maniera specifica ai sensori, al loro tipo e al tipo di informazioni che raccolgono. Non fanno riferimento ad aggregazioni od elaborazioni di alcun tipo di queste informazioni. Inoltre, allo stato attuale la definizione della parte di sensoristica è, anche a detta degli autori, ancora incompleta e in fase di definizione.

3.3.4 Progetto CITYkeys

Il progetto CITYkeys [16] ha come obiettivo il miglioramento nello scambio di informazioni tra i vari soggetti che forniscono servizi all'interno di una smart city. Lo scambio di tali informazioni deve permettere l'implementazione di azioni e procedure tali da rendere più efficiente la città nella gestione dei servizi e delle sue attività.

Il progetto cerca di identificare i KPI (Key Performance Indicator) per aiutare la città a implementare strategie collegando vari servizi o un progetto con obiettivi definiti. Essi definiscono la base per una Smart City per pianificare e misurare il suo progresso, attraverso la valutazione dell'impatto di uno specifico progetto svolto all'interno della Smart City, mostrando il progresso dell'intera Smart City verso gli obiettivi prefissati e determinando come un progetto ha contribuito al raggiungimento degli obiettivi. I KPI sono collegati a una strategia e sono, quindi, critici per il successo dell'esecuzione.

I KPI sono classificati secondo le categorie su cui ha impatto il lavoro in ambito di sostenibilità di una Smart City. Queste categorie sono:

- People: attrattività per gli abitanti e quindi indicatori sulla qualità della vita, la salute e l'educazione;

- Planet: pulizia ed efficienza della città e quindi indicatori sul consumo energetico, cambio climatico, inquinamento e rifiuti;
- Prosperity: equità della società, supporto a soluzioni smart e green, e quindi indicatori su lavoro, performance economica e innovazione;
- Governance: amministrazione e implementazione dei progetti e quindi indici su implementazione di policy per smart city e coinvolgimento della comunità;
- Propagation: potenziale di diffusione ad altre città o contesti dei progetti e quindi indici su replicabilità e scalabilità di una smart city.

Il progetto definisce diverse tipologie di indici: input, processo, output, risultato e impatto. Un indice di input si riferisce alle risorse necessari per eseguire un'attività o eseguire una misurazione (per esempio risorse umane economiche o materiali necessari), un indice di processo si riferisce a quando un'attività pianificata è stata eseguita (per esempio la distribuzione di smart meter), un indice di output aggiunge dettagli al risultato di un'attività (per esempio il numero di smart meter distribuiti), indice di risultato si riferiscono agli obiettivi di un risultato (per esempio la percentuale di case con isolamento termico dopo una campagna di incentivazione all'isolamento termico), infine indice di impatto che misurano la quantità e qualità di risultati a lungo temine di una pianificazione (per esempio la riduzione nell'uso di energia).

Il progetto si sofferma in particolar modo sugli indicatori di impatto. Per esempio, un indicatore può essere la diminuzione di emissioni di gas serra. Il risultato può essere ottenuto in vari modi, come la costruzione di edifici con migliore isolamento termico o con l'immatricolazione di auto elettriche. L'indicatore permette alla città di raggiungere l'obiettivo nel modo che si ritiene opportuno senza prescrivere il modo in cui si debba raggiungere. Focalizzandosi sugli impatti è possibile valutare soluzioni cross-settoriali.

Un grosso limite è che gli indici non sono organizzati sotto forma di ontologia, ma si tratta principalmente di un elenco di indicatori raggruppati per macro aree. Inoltre, le informazioni sugli impatti sono disponibili solo dopo un certo periodo di tempo dall'implementazione del progetto. Inoltre, fattori contestuali possono influenzare l'indice di impatto. Nonostante questo, il valore di impatto è l'unica misura che conta per valutare e raggiungere gli obiettivi prefissati.

Il progetto, inoltre, elenca una raccolta estensiva e dettagliata di indicatori raggruppati per temi e definisce una serie di parametri per effettuare una scelta di quali indicatori scegliere per ottenere una breve lista che copra gli aspetti di interesse.

3.4 Integrazione di ontologie esistenti con la Smart City Platform

Una ontologia è di interesse se, oltre a rappresentare le informazioni di interesse, consente anche un collegamento con altre informazioni che sono connesse a esse. In questo modo la loro descrizione semantica risulta essere più completa e utile al fine di una elaborazione automatica. Per questo motivo si è previsto l'uso di alcune altre ontologie al fine di connotare meglio le informazioni che vengono raccolte dai vari verticali della Smart City.

L'ontologia PROV Ontology (**PROV-O**) [13] fornisce un set di classi, proprietà e restrizioni che possono essere usati per rappresentare e scambiare informazioni di provenienze generate in diversi sistemi e diversi contesti. È possibile creare nuove classi e proprietà per modellare la provenienza di informazioni da differenti applicazioni e domini.

Ad esempio, un aspetto importante di un indicatore è la sua provenienza, cioè da quali dati deriva e come è stato calcolato. La PROV Ontology ha proprio questa funzione. A partire dal generico concetto *Entity* di cui si vuole specificare la provenienza, è affiancati il concetto di *Agent* che rappresenta l'organizzazione che esegue l'azione di creazione. Nel caso specifico, questo concetto è esteso con *DataProvider* che descrive la l'ente a cui è demandata l'attività di creazione del dato.

Al fine di descrivere i fornitori dei dati, la Friend of a Friend Ontology (**FOAF**) [17] può essere usata. Essa permette di definire in modo articolato una descrizione dei fornitori e delle organizzazioni coinvolte nel processo di generazione ed elaborazione degli Urban Dataset.

Dal momento che la raccolta delle informazioni che coinvolgono la Smart City riguardano dati oggettivi misurabili, risulta necessario anche un sistema univoco per la descrizione delle misure effettuate. Esistono diverse ontologie che modellano la raccolta di misurazioni. In particolare è di interesse, per i nostri obiettivi, una rappresentazione organica delle unità di misura. In questo campo sono state valutate in particolare la Quantities, Units, Dimensions and Types Ontology (**QUDT**) [18] e la Ontology of units of Measure **(OM)** [19] entrambe usate per descrivere le quantità numeriche usate nella misura dei dati rilevati.

La QUDT è una ontologia OWL2 sviluppata da NASA e TopQuadrant mentre la OM è sviluppata da VU University in Olanda. Entrambe specificano diverse unità di misura su diversi ambiti. La particolarità di OM è che rappresenta anche multipli e sottomultipli di una unità di misura attraverso la definizione dei prefissi e definendo le unità di misura con prefisso come facente riferimento alla unità di misura base con l'aggiunta del prefisso. La QUDT invece non effettua questo tipo di organizzazione e non è chiara come decida di inserire un multiplo (so sottomultiplo) nello schema, dal momento che non sono tutti presenti. Questa mancanza può causare problemi nel caso di un'annotazione automatica, ovvero nella conversione di unità di misura diverse provenienti da diverse fonti. Un altro problema operativo della QUDT è dovuto al fatto che la versione attualmente disponibile presenta degli errori. Per questo motivo sistemi di analisi delle ontologie non funzionano correttamente nel verificare la correttezza semantica delle regole definite. Per questo e per altri motivi discussi in [20] si preferisce utilizzare OM Ontology per la creazione della nostra ontologia.

3.5 Conclusioni

L'analisi svolta finora è partita dall'esplorazione di ontologie esistenti in ambito Smart City cercando di comprendere le modalità con cui si è cercato di affrontare questo tema in passato. Si sono, inoltre, osservate alcune delle ontologie esistenti con lo scopo di integrarle nella nostra per descrivere argomenti secondari ma utili (per esempio il caso di FOAF) e altre al fine di agganciarle a concetti più generali e che ne permetta il collegamento con altre informazioni esterne (come PROV-O e OM).

Si sono analizzati i progetti CityProtocol, CITYkeys, DIMMER e Km4City. Il primo prevede la definizione di una ontologia per raggruppare e ordinare gli indicatori della città. Il secondo progetto ha come obiettivo l'individuazione di tali indicatori. Il vantaggio di un approccio alla gestione della Smart City attraverso l'uso di indicatori generati da aggregazione di dati permette una maggiore resistenza alle interferenze degli errori di misura, ma non danno garanzia e supporto alla reattività della risposta in quanto tali indici non sono in grado di fornire informazioni per poter reagire a eventi eccezionali, ma solo una valutazione della bontà dell'azione di gestione che l'indice cerca di misurare. Gli indici proposti, in particolare, hanno un'alta inerzia perché sono focalizzati a descrivere l'andamento generale della città e possono volerci anche diverse settimane prima che ci sia un cambiamento sensibile. DIMMER integra informazioni sulla topologia di edifici con dati real-time provenienti da sensori e da utenti per analizzare e correlare l'uso degli edifici e fornire informazioni sul comportamento degli stessi sul consumo energetico in tempo reale. L'ontologia svolge anche un ruolo di indice delle informazioni puntando al database dove sono memorizzati i dati cercati attraverso le query predisposte. Quest'ultimo è probabilmente l'aspetto più interessante, in questo modo l'ontologia è slegata dalla raccolta dati e funge da indice per gli stessi. Un approccio sensato quando si intende gestire un grosso flusso di dati attraverso sistemi ottimizzati per il big data. Km4City è un'ontologia che permette di integrare le informazioni su trasporti e mobilità con i dataset opendata messi a disposizione dal comune di Firenze a dalla regione Toscana.

Dal momento in cui una struttura dove molti dati sono convogliati in un intervallo di tempo ristretto, come può essere il caso in cui tanti sensori raccolgono dati ad alta frequenza, e si hanno molti dati da inserire sui supporti di memoria, si ritiene che una soluzione sia quella di utilizzare l'ontologia come un indice dei dati e delle informazioni disponibili. Lo scopo diventa quello di puntare verso la tabella contenente le informazioni e descriverne il contenuto al fine di facilitare l'uso e l'elaborazione di queste informazioni.

Nel prossimo capitoli si procede descrivendo il lavoro svolto in precedenza e come sono state definite le nuove caratteristiche dell'ontologia.

4 Modello semantico di riferimento

In questo capitolo sono ripresi i concetti sviluppati durante l'anno precedente del progetto e viene presentata la nuova versione tenendo conto delle nuove specifiche.

4.1 Il lavoro svolto in precedenza

Nel documento dello scorso anno, si è descritto il lavoro svolto per l'individuazione di una ontologia comune utile per favorire lo scambio di informazioni all'interno di una piattaforma ICT per una Smart City. Lo scopo di tale piattaforma era la condivisione di informazioni tra i vari ambiti applicativi di una città. La raccolta e la condivisione di informazioni tra i diversi ambiti può essere utile a creare sinergie tali da favorire nuovi e più importanti risultati nell'ambito del risparmio energetico e dell'efficienza di una città. Il vantaggio offerto da un'ontologia in tale ambito è la sua capacità di agire come framework unificante fra le diverse parti interagenti e fungere da base per l'interoperabilità tra sistemi realizzati con differenti modelli, paradigmi e linguaggi di programmazione. Tra le altre cose,

- Garantisce benefici dal punto di vista dell'affidabilità, in quanto una rappresentazione formale renderebbe automatico il controllo della consistenza e quindi il software più affidabile.
- Può essere utile nel processo di identificazione dei requisiti e definizione di una specifica per sistemi IT.
- Favorisce l'interoperabilità e la scalabilità dei servizi in ambienti grandi e aperti e può essere usata facilmente come base per l'applicazione di regole logiche per l'inferenza automatica di nuova conoscenza.

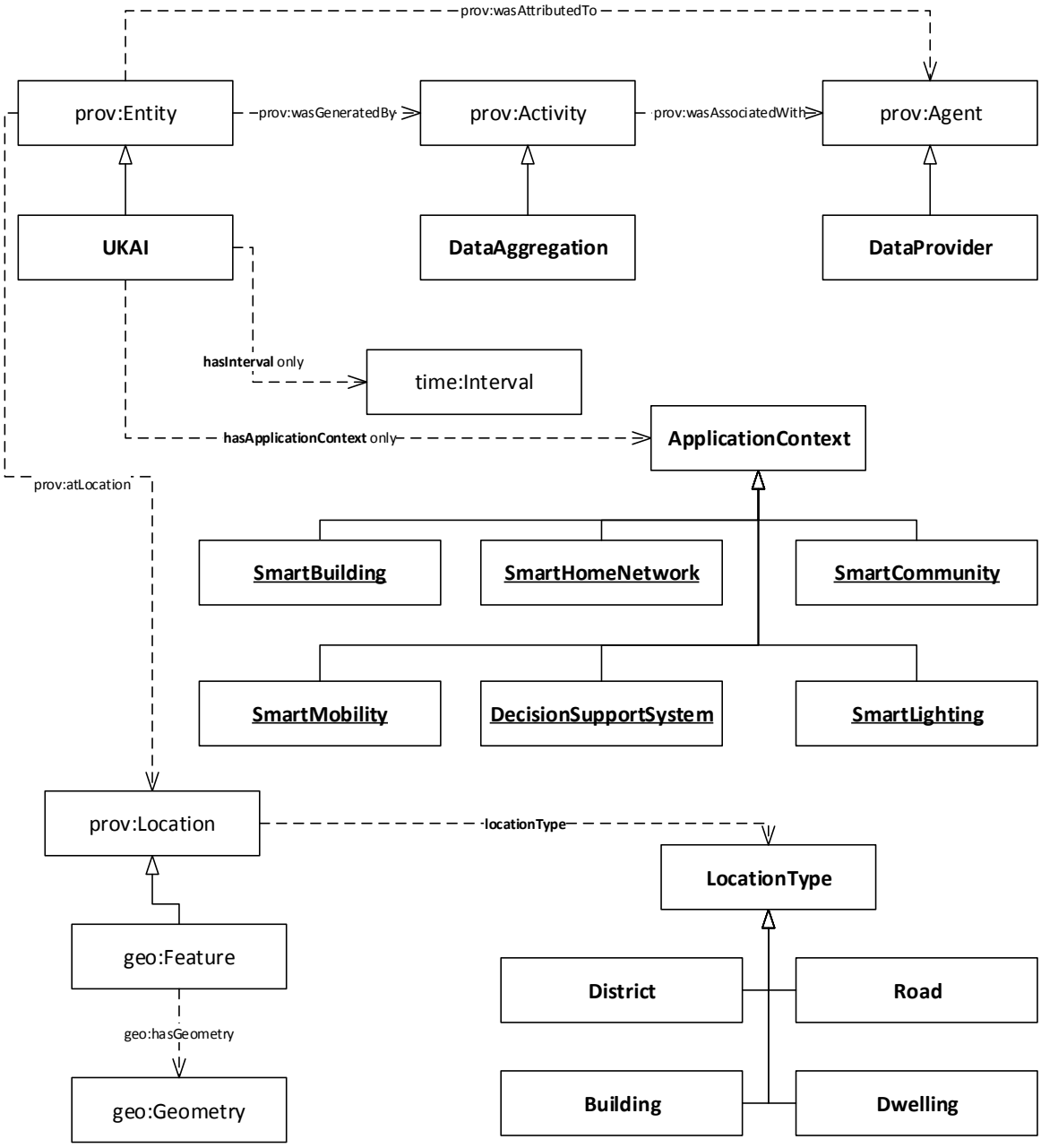
Il lavoro svolto è iniziato osservando le diverse esperienze in ambito Smart City già avviate all'estero e si è cercato di trarre delle linee guida sul lavoro da svolgere. L'idea iniziale era di estendere la mappatura dell'ontologia su tutti i dispositivi di sensing della città e perciò si è partiti dall'esplorazione di ontologie esistenti in ambito Smart City riguardanti dati provenienti da sensori. Si sono osservate diverse delle ontologie esistenti e sono state individuate alcune come possibili candidate per una loro adozione/estensione o da cui trarre ispirazione per la definizione di una nuova ontologia per Smart City adatta ai nostri scopi. La criticità mostrata da questo modello, in cui si opera direttamente da un dato grezzo, riguarda l'eccessiva volatilità di un dato proveniente da un sensore, che potrebbe mostrare dati errati e quindi condurre a eseguire azioni non necessarie, o addirittura dannose, in risposta a tale input. Nel caso di azioni dannose, potrebbero verificarsi conseguenze anche gravi, che possono essere causa di problemi anche legali. Esiste, inoltre, il problema della gestione della rete di sensori. Se la Smart City decidesse di appoggiarsi a un servizio esterno, dovrebbe richiedere un livello di servizio al fornitore e anche definire dei livelli di responsabilità. Attività che possono risultare difficili da implementare.

La ricerca si è quindi spostata sull'analisi di progetti per Smart City che prevedessero una ontologia di supporto e riguardanti indicatori della città. In base a tale analisi si è deciso di spostare l'attenzione sulla definizione di un nuovo tipo di indice che fosse una via di mezzo tra i dati grezzi e indicatori di andamento di progetti.

Si è, pertanto, partiti dall'analisi degli scenari di riferimento su cui operare la raccolta delle informazioni utilizzate della piattaforma. Si sono individuate le caratteristiche comuni che contraddistinguono le informazioni da scambiare e gli interlocutori delle comunicazioni, ovvero la Piattaforma ICT principale e le diverse piattaforme dei singoli contesti applicativi. Tale analisi ha mostrato che esiste una struttura uniforme di comunicazione tra la piattaforma centrale e le piattaforme dei diversi contesti applicativi e ha perciò permesso di individuare una struttura di base comune per l'ontologia da usare al fine di organizzare la struttura delle informazioni.

Dopo di che si è passati all'individuazione dei dati da correlare all'informazione principale, ovvero quelli cheche nel report dello scorso anno erano indicati UKAI (Urban Key Application Indicator) e che nel corso di quest'anno sono stati ribattezzati Urban Dataset, dati urbani con lo scopo di indicare criticità ed efficienza di una Smart City. Infatti, l'utilità di tali dati c'è se essi sono associati a informazioni sul contesto che li ha generati. In particolare sono state individuate come necessarie le informazioni temporali e di localizzazione

dei dati usate per ottenere l'aggregazione che ha generato l'indice UKAI. Oltre a queste, sono state collegate anche altre informazioni sugli autori dell'aggregazione, ovvero il fornitore del servizio alla pubblica amministrazione, e informazioni sulle tecniche usate per effettuare l'aggregazione. Infine, queste informazioni, sono state ricondotte a una ontologia di più alto livello per descrivere la provenienza delle informazioni. In Figura 1 è mostrata una visione d'insieme dell'ontologia com'è stata ipotizzata durante lo scorso anno.



prefix time: <http://www.w3.org/TR/owl-time/#> .
prefix prov: <http://www.w3.org/TR/2013/REC-prov-o-20130430/>
prefix geo: <http://www.opengis.net/ont/geosparql>

Figura 1. Visione completa dello schema di ontologia

L'ontologia qui mostrata descrive soltanto ad alto livello i collegamenti tra i diversi concetti e necessita di una definizione completa di tutte le proprietà necessarie al fine di poter far sì che sia utilizzabile nella pratica. A questo scopo si è ipotizzata una fase di mapping dell'ontologia su di un caso d'uso

ragionevolmente vicino alla realtà che è in corso di definizione all'interno del progetto, ma non ancora definito completamente. La fase di mapping è partita da una serie di informazioni ragionevolmente utili per descrivere un UKAI e si è spostato sull'adattamento della struttura dell'informazione ai vincoli imposti dall'ontologia fin qui definita.

Di seguito è mostrato una parte di un esempio di mapping utilizzando il formato Turtle:

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix time: <http://www.w3.org/2006/time#> .
@prefix : <http://example.com/> .

:UKAI_0013_ad47c
      a :UKAI ;
      :hasApplicationContext :smartHomeNetwork ;
      :UKAI_ID "0013" ;
      rdfs:label "UKAI consumo giornaliero ad abitante" ;
      rdfs:comment "totale dei consumi dentro una smart home network per
                                          un giorno per abitante" ;
      prov:value "2,3"^^xsd:float ;
      :unitOfMeasure "kWh/persona" ;
      :updateFrequency "Giornaliero" ;
      :generatedAt [
            a time:Instant ;
            time:inXSDDateTime "2016-08-15T12:00:24+02:00"^^xsd:dateTime ;
      ] ;
      :hasInterval [
            a time:Interval ;
            time:hasBeginning [
                  a time:Instant ;
                  time:inXSDDateTime "2016-08-14T12:00:00+02:00"^^xsd:dateTime ;
            ] ;
            time:hasEnd [
                  a time:Instant ;
                  time:inXSDDateTime "2016-08-15T12:00:00+02:00"^^xsd:dateTime ;
            ] ;
      ] ;
      prov:atLocation :dataLocation_01 ;
      prov:wasAttributedTo : dataProvider_01 ;
      prov:wasGeneratedBy : dataAggregation_01 .
```

Il messaggio inizia con una serie di indicazioni di prefissi per la definizione del namespace. Si tratta di una particolarità del formato utilizzato che ne facilità la leggibilità per le persone. Dopodiché viene presentato una istanza della risorsa UKAI con tutte le sue proprietà. L'istanza è identificata con un id univoco in modo da poter essere distinta. Tale identificatore potrebbe essere generato a partire da alcune sue proprietà scelte in modo da evitare collisioni di nomi. Ad esempio potrebbe essere una combinazione del suo id con l'hash del timestamp di generazione. Le proprietà sono organizzate secondo la descrizione fornita dall'ontologia definita e da quelle utilizzate per mappare alcuni concetti. In particolare, sono evidenti le rappresentazioni temporali per descrivere il momento di generazione del dato aggregato e l'intervallo di tempo preso in considerazione per l'aggregazione. Tali informazioni sono definite rispettivamente come *Instant* (proprietà *generatedAt*) e *Interval* (proprietà *hasInterval*) composto a sua volta da due *Instant* per definire inizio e fine dell'intervallo.

Naturalmente è presente il valore finale dell'aggregazione effettuata assieme a una indicazione dell'unità di misura. A proposito dell'unità di misura, esistono diverse ontologie che armonizzano i concetti di unità di misura. Andranno valutati in base alle necessità che si presenteranno. L'uso di una ontologia per le unità di misura può permettere anche la comparazione di dati che fanno riferimento a contesti diversi ma che

possono essere correlati come il consumo elettrico dei diversi contesti al fine di effettuare, ad esempio, calcoli di efficienza usando un'aggregazione temporale o spaziale.

Alla fine del file sono presenti tre proprietà che sono collegate a istanze non definite in questo messaggio. Si tratta di istanze che si presuppongono essere già presenti nella base di conoscenza dove il messaggio sarà immagazzinato dalla Piattaforma ICT. Infatti tali istanze fanno riferimento al luogo dove si trova l'abitazione sorgente dell'informazione, al fornitore dell'informazione e gestore di quella specifica infrastruttura e alla descrizione della modalità di aggregazione utilizzata, se necessaria.

Si sono fatte delle assunzioni per le parti non coperte dall'ontologia per giungere a un risultato in grado di essere applicabile in un sistema software reale. In questo modo si è potuto valutare l'efficacia della struttura fin qui definita. In ogni caso, per ottenere un risultato completo è necessaria la completa definizione degli UKAI e delle informazioni che l'infrastruttura deve scambiarsi.

4.2 Le nuove specifiche

Nel corso del primo anno del progetto, è stato definito il concetto di Urban Key Application Indicator (UKAI), quale indice aggregato che riassuma, su un certo periodo di tempo e ambito di spazio, informazioni significative su un aspetto ritenuto importante dal punto di vista della gestione dello Smart District (per esempio un indice di consumo energetico). Nel corso del secondo anno si è deciso di trasformare gli UKAI in dataset di informazioni relative alla Smart City da pubblicare sul web con il nome di Urban Dataset

In particolare (Figura 2):

- Il livello di aggregazione spaziale può andare dal singolo dispositivo (item) all'intera città;
- l'orizzonte temporale può andare dalla misura istantanea effettuata al valore medio sul breve-medio o lungo periodo, fino al valore anagrafico inviato una tantum per identificare una risorsa una volta per tutte;

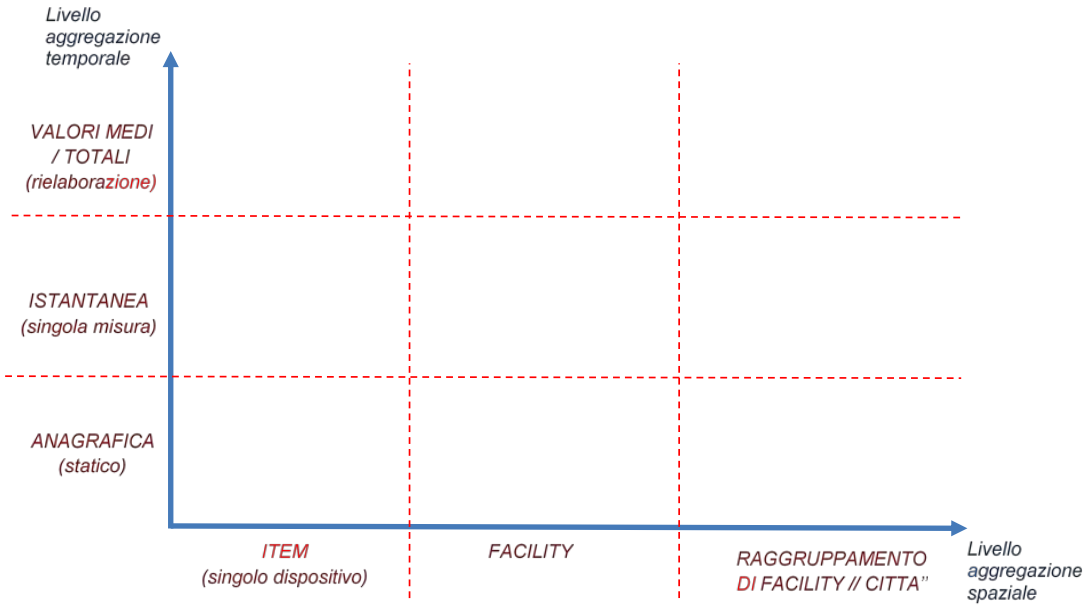


Figura 2 Classificazione degli Urban Dataset in base ad aggregazione spaziale e temporale

L'idea di partenza è di dare alle amministrazioni pubbliche uno strumento per monitorare i servizi affidati a fornitori. Supponiamo, per esempio, che un comune emetta il bando per la gestione dell'illuminazione pubblica. Potrebbe definire degli Urban Dataset per verificare il consumo energetico e i livelli d'illuminazione e richiederne il valore medio sulle ore di accensione, aggregate a livello di strada. Queste dovrebbero essere inviate al comune tramite la piattaforma orizzontale, che esporrà un Web Service Rest a cui inviare tali dati in un determinato formato (per esempio JSON).

Dunque:

- Tramite tali Urban Dataset, il comune potrà verificare che l'azienda rispetti i limiti di consumo energetico senza tenere le strade scarsamente illuminate.
- Al momento della scadenza del contratto di gestione dell'illuminazione pubblica e dei dati relativi, il comune, rifacendo la gara, potrà richiedere, nel bando, all'azienda vincitrice di fornire i dati allo stesso modo: dunque l'amministrazione pubblica potrà riutilizzare la stessa piattaforma, senza ricostruirla da zero.
- Se altre città adottassero lo stesso sistema, la stessa utility potrebbe partecipare al bando, senza dover rifare l'infrastruttura.

Estendendo il ragionamento precedente, se si adottasse il sistema per le varie utility:

- L'amministrazione presenterebbe un'interfaccia unica verso tutti i fornitori.
- Le varie applicazioni potrebbero scambiarsi dati, tramite la piattaforma orizzontale.

Per esempio, se s'implementasse sui lampioni un sistema di monitoraggio del traffico, per regolare la luminosità sulla base delle necessità, questi dati potrebbero essere inviati all'applicazione di monitoraggio del traffico, gestita da un altro soggetto.

Per raggiungere questi scopi, non è sufficiente definire il singolo indice, ma occorre definire anche la rappresentazione ontologica del modello dati e dei vari Urban Dataset che permettano, in modo semplice, di passare da un formato a un altro, di costruire una serie di strumenti che facilitino l'inserimento dei dati nei vari formati e, comunque, di avere una comprensione semantica del singolo Urban Dataset.

Gli Urban Dataset sono, quindi, strutture dati che trasmettono informazioni legate a uno specifico dominio applicativo, pensati per fare da tramite fra applicazioni verticali e l'applicazione orizzontale di gestione della città (Smart City Plaform) o fra diverse applicazioni verticali (per esempio singoli sensori/dispositivi ma anche piattaforme di gestione locale di un dominio che elaborano i dati grezzi e forniscono servizi).

Figura 3 Scambio di Urban Dataset fra applicazioni verticali

Precedentemente si era pensato di fare in modo che l'ontologia contenesse anche le informazioni degli Urban Dataset, cioè tutti i dati generati dai sensori e dai sistemi che elaborano informazioni a partire dai dati dei sensori. Una scelta del genere, data la mole di dati prevista in ingresso sarebbe stata insostenibile da gestire da parte di un sistema basato su tecnologie del web semantico quali RDF e SPARQL. Per questo si è deciso che i dati vengano memorizzati in database esterni all'ontologia. Lo schema di riferimento dello scambio degli Urban Dataset è quello mostrato in Figura 3. Un applicativo verticale usa il formato dati (sia

esso JSON, XML o magari CSV) per inviare verso la Smart City Plaform gli Urban Dataset e questa lo memorizza nel proprio DB e/o lo invia a uno o più applicativi connessi a essa.

Nel precedente schema, gli Urban Dataset possono essere definiti, coerentemente con le definizioni di aggregazione spaziale e temporale:

- A livello di dispositivo/sorgente (detto, in questo progetto, anche livello di campo): questi saranno tipicamente Urban Dataset di Item in Real-Time. Per esempio, il sensore di una Smart Street segnala il numero di macchine parcheggiate in un determinato momento.
- All'interno della piattaforma locale, ovvero i dati aggregati nelle piattaforme locali, per la gestione interna del contesto applicativo (con protezione dei dati a tutela della privacy). Saranno Urban Dataset con aggregazione spaziale a livello di facility, su un tempo medio. Per esempio, il numero medio auto nella fascia oraria 17-18 dei giorni lavorativi in una particolare strada monitorata dalla piattaforma Smart Street.
- A livello dell'Area Smart City Platform, ovvero quei dati elaborati nelle piattaforme locali per descrivere un aspetto a livello di distretto o città: Per esempio, numero medio auto nella fascia oraria 17-18 dei giorni lavorativi in tutte le strade monitorate del distretto dalla piattaforma smart street.

4.3 Descrizione del nuovo modello dell'ontologia

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

È inoltre utile utilizzare ontologie già esistenti, definite e testate in altri contesti, per associare, quando possibile, ai concetti e dati di Urban Dataset altre informazioni quali ad 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 qualcosa di già testato e che quindi ha superato la validazione dovuta all'uso di tale strumento in un'altra applicazione. Un altro vantaggio è dovuto dall'uso di un qualcosa già noto ad altri o ad altri sistemi automatici. In questo modo è possibile che siano stati già definiti strumenti che siano in grado di analizzare quella sottoparte di grafo RDF in maniera automatica. L'utilizzo di una rappresentazione comune di un dominio permette inoltre di condividere anche tra umani la conoscenza di dominio facilitando la comprensione della struttura dei dati e favorendo il riuso di tali conoscenze anche in altre applicazioni e in contesti diversi. Il riuso facilita inoltre non solo la comunicazione e la condivisione, ma anche la manutenibilità del sistema in generale.

Di seguito sono spiegate le varie parti che compongono l'ontologia. 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.

4.3.1 Concetti principali

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* (Figura 4) descrive l'ente a cui è demandata l'attività di creazione del dato.

Un Urban Dataset (Figura 4), quando viene generato, viene associato ad altre informazioni statiche che lo descrivono, come il tempo di creazione, la posizione del sensore da cui è prodotto, la sua descrizione o il suo identificativo. Per questo motivo sono definite come data property informazioni specifiche come il nome, la descrizione, un URI di riferimento e la versione, ovvero la **specificationRef**. Ogni Urban Dataset necessita di altre informazioni di contesto che servono per descrivere meglio le informazioni e per caratterizzarle, ovvero informazioni come produttore, posizione geografica a cui fanno riferimento i dati, lingua usata per le descrizioni, tempo e fuso orario. Ad esempio la lingua delle descrizioni, le coordinate da cui sono stati presi i dati o il timestamp del momento di raccolta dei dati.

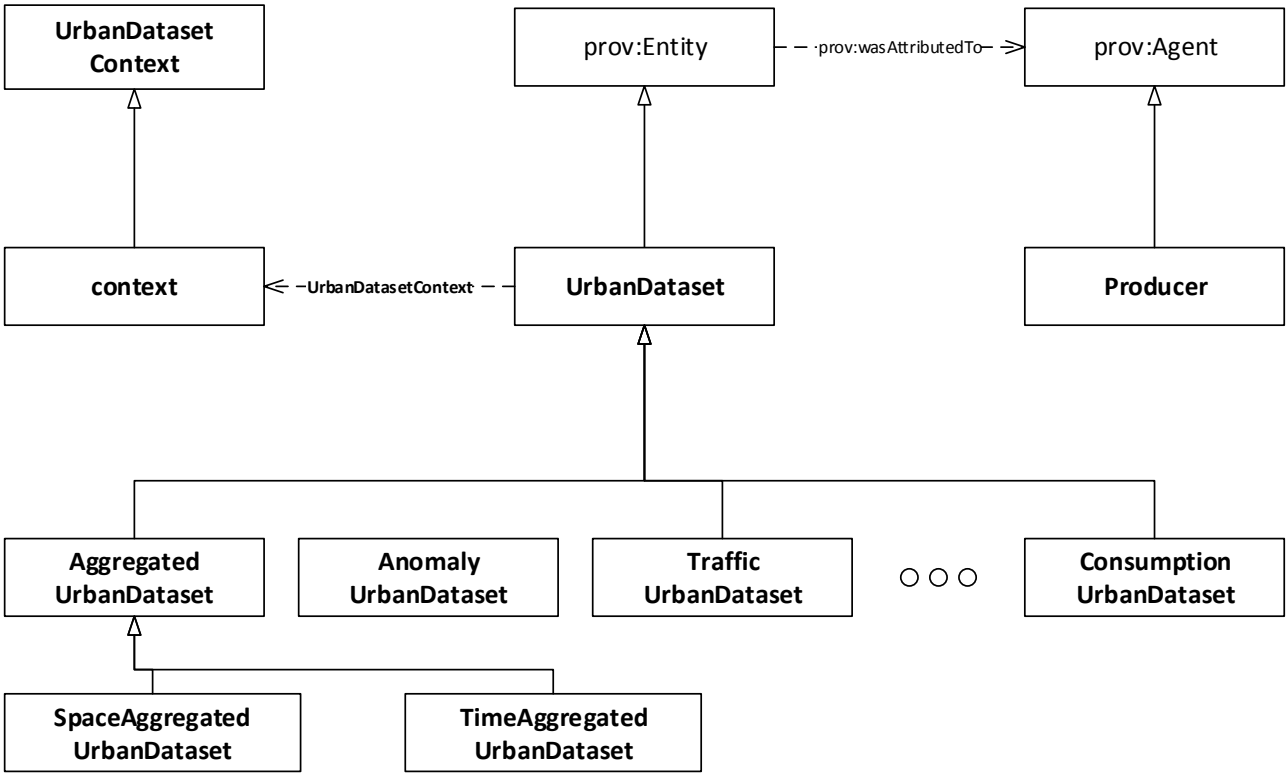


Figura 4 Schema dei principali concetti relativi agli Urban Dataset

L'entità Urban Dataset, definita come sottoclasse di Entity dell'ontologia PROV-O, è collegata alle informazioni centrali, ovvero i valori che fanno parte dei dataset messi a disposizione attraverso la piattaforma. Ad esempio il numero e il tipo di anomalie presentatesi in un intervallo di tempo per l'edificio osservato oppure il numero medio di parcheggi liberi su una particolare strada. Nel caso si faccia riferimento a quantità fisiche misurabile, si è deciso di aggiungere un riferimento alla ontologia OM per collegarla alle unità di misure del sistema internazionale. Oltre alle specifiche proprietà, all'Urban Dataset è anche associato un campo che indica 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. Quest'ultimo, il *Producer*, è una sottoclasse di *Agent* di PROV-O per cui può essere usata, come proprietà per riferire il produttore, wasAttributedTo di PROV-O.

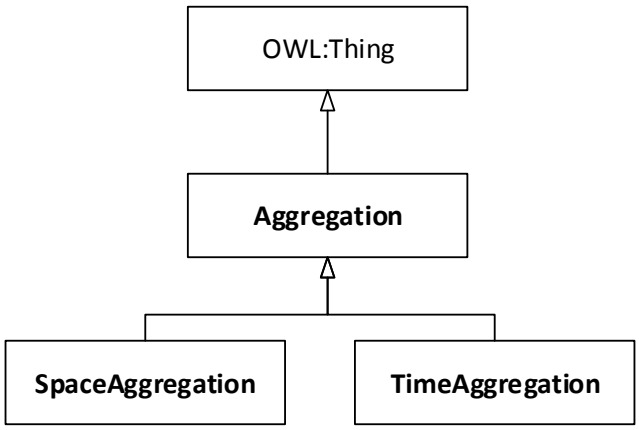


Figura 5 Schema dei concetti per definire il tipo di aggregazione

Uno stesso Urban Dataset può essere aggregato in spazio e tempo in modo diverso. È stata ipotizzata la possibilità di definire dei livelli di aggregazione predefiniti e creare delle istanze diverse di Urban Dataset in funzione delle modalità di aggregazione utilizzate. In particolare sono state definite le classi *Space* e *TimeAggregatedUrbanDataset* (Figura 4), 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* (Figura 5). Ad esempio, se si volesse definire un Urban Dataset che fornisce un livello di aggregazione settimanale dei parcheggi liberi in una strada, si può definire un nuovo Urban Dataset che fa riferimento all'Urban Dataset base e al livello di aggregazione "average" definito come istanza di TimeAggregation come mostrato nell'esmpio seguente:

```
<owl:NamedIndividual
rdf:about="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#AverageFreeParking">
        <rdf:type
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#TrafficUrbanDataset"/>
        <HasUrbanDataset
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#FreeParking"/>
        <context
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#context1"/>
        <hasAggregation
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#average"/>
        <hasApplicationContext
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#SmartMobility"/>
        <hasUrbanDatasetProperty
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#FreeParkNum"/>
        <prov:wasAttributedTo
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#SmartStreet"/>
        <UrbanDataset_ID
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">UrbanDataset
D5b</UrbanDataset_ID>
    </owl:NamedIndividual>
```

Qui sopra è mostrato un estratto di RDF dove si definisce una istanza *AverageFreeParking* in cui si fa riferimento all'aggregazione (proprietà *hasAggregation*) e al tipo (proprietà *type*) di Urban Dataset.

Gli Urban Dataset che sono frutto di aggregazione sono comunque definiti come istanze di Urban Dataset. Viene poi specificato che la classe *TimeAggregatedUrbanDatasetAnomaly*, sottoclasse di *TimeAggregatedUrbanDataset* ha una equivalenza con le istanze che hanno la proprietà *hasUrbanDataset* definita una sola volta con una istanza di *TrafficUrbanDataset* e possiede la proprietà *hasAggregation* definita una sola volta con una istanza di *TimeAggregation*. In questo modo è possibile aggiungere facilmente nuove possibilità di aggregazione che possono valere per tutti gli Urban Dataset già definiti o ancora da definire e avere già una gerarchia pronta per questo uso. Questa logica è stata anche usata per la definizione delle unità di misura con prefissi all'interno dell'ontologia OM che è stata integrata in questa ontologia.

4.3.2 Concetti di supporto

Le singole proprietà a cui sono associati i diversi Urban Dataset sono a loro volta delle istanze di sottoclassi di *Property* (Figura 6). 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.).

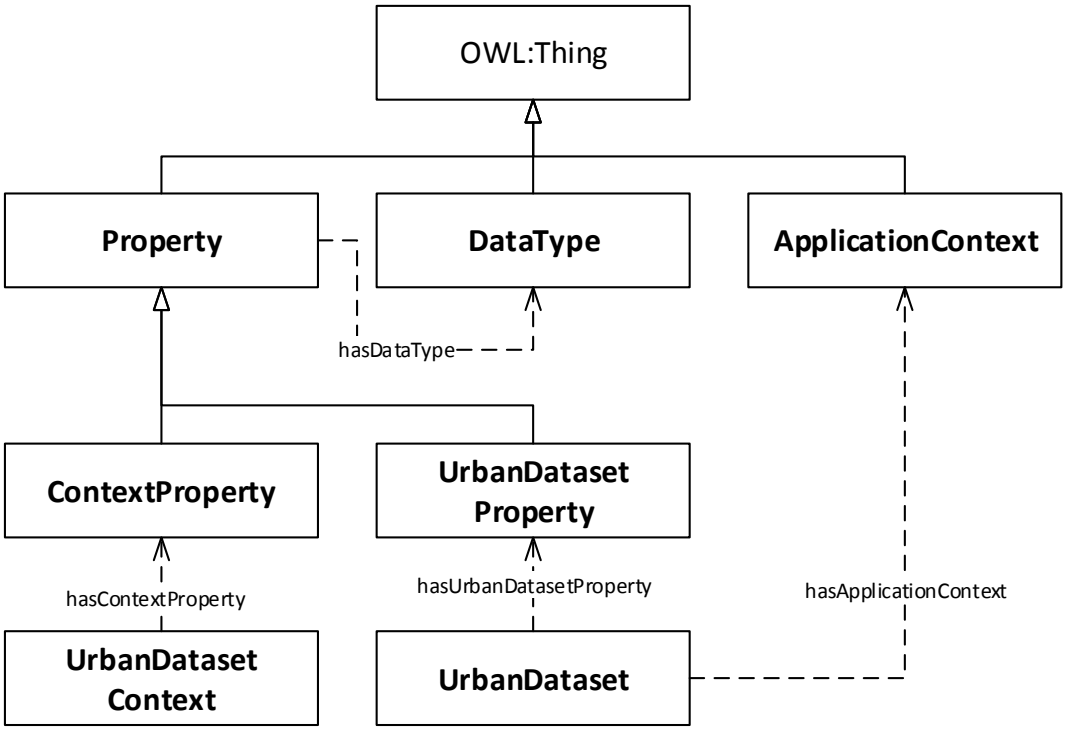


Figura 6 Concetti di supporto

È possibile anche creare proprietà composte, ovvero è possibile che una proprietà sia composta a sua volta da altre proprietà. Ciò è descritto semplicemente creando una istanza di proprietà che ha come uno o più riferimenti ad altre istanze di proprietà tramite le object property *ContextProperty* e *UrbanDatasetProperty*.

A ogni proprietà sono state associate il *DataType*, ovvero il formato del dato usato per rappresentare l'informazione, ovvero un numero intero, un numero reale, una string o un timestamp, tramite la object property *hasDataType*. Oltre a questo è stata prevista l'associazione con una 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.

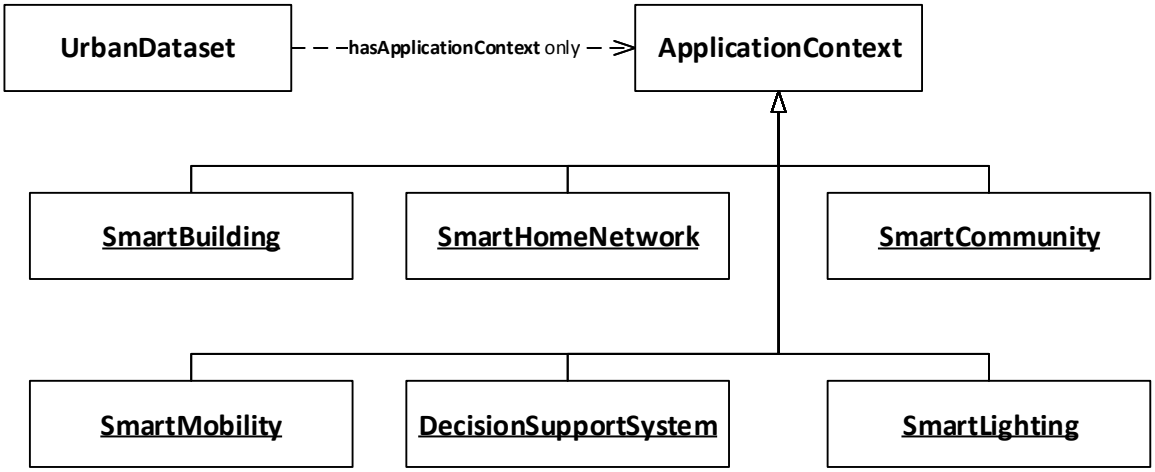


Figura 7 Associazione tra UrbanDataset e ApplicationContext

Un Urban Dataset contiene un'aggregazione di dati provenienti da sensori di tipi specifici e perciò tali informazioni sono ascrivibili a un particolare contesto applicativo. Per questo motivo, è stata prevista l'esistenza di una proprietà che metta in relazione una particolare istanza di *UrbanDataset* con un contesto applicativo. In Figura 7 sono definite le istanze (indicate con testo sottolineato) che 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.*

Di seguito è specificata l'istanza a cui fa riferimento la proprietà FreeParkNum dell'esempio precedente dove si vedono l'unità di misura e il tipo di dato utilizzato:

```
<owl:NamedIndividual
rdf:about="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#FreeParkNum">
    <rdf:type
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#UrbanDatasetProperty"/>
    <om-2:hasUnit rdf:resource="http://www.ontology-of-units-of-
measure.org/resource/om-2/Number"/>
    <hasDataType
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#integer"/>
    <rdfs:comment>numero parcheggi liberi</rdfs:comment>
</owl:NamedIndividual>
```

Di seguito è mostrata una istanza di contesto utilizzata dagli Urban Dataset dove sono mostrate le proprietà di contesto usate:

```
<owl:NamedIndividual
rdf:about="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#context1">
        <rdf:type
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#UrbanDatasetContext"/>
        <hasContextProperty
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#coordinate"/>
        <hasContextProperty
rdf:resource="http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-
ontology#language"/>
</owl:NamedIndividual>
```

4.3.3 Concetti da 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 8) 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 8 Uso di PROV-O

Per le nostre necessità servivano le unità di misura rappresentate come istanze in modo che fosse possibile associarle alle istanze di *UrbanDataset*. Le ontologie individuate che rispettavano questa specifica sono state OM e QUDT. Si è preferito propendere per OM per via del sistema di organizzazione delle unità di

misura multiple e sottomultiple e perché la versione corrente di QUDT presenta degli errori in alcune dichiarazione che causano degli errori all'apertura della stessa con un editor specifico. Per maggiori dettagli sulle differenze tra le due ontologie si rimanda a [20]

L'ontologia usata per specificare le unità di misura è OM. Essa contiene una 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 9 è mostrata la struttura completa dell'ontologia. Nel nostro caso non viene usata tutta ma solo la parte relative alle singole unità di misura, ovvero le istanze della classe Unit.

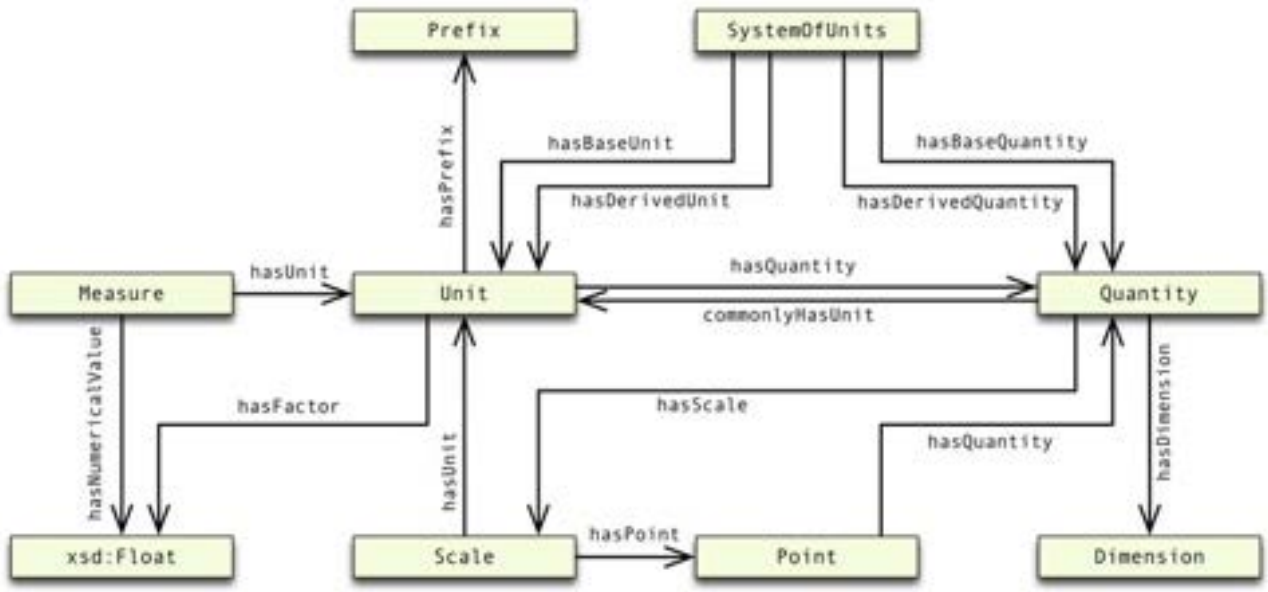


Figura 9 Struttura delle classi dell'ontologia OM

I produttori dei dati possono essere diversi, dato che i diversi sensori possono essere gestiti 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 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.

Nei prossimi capitoli verranno mostrate le query usate per poter accedere ai dati inseriti nell'ontologia.

5 Modalità di inserimento e accesso ai dati dell'ontologia

Un'ontologia può esprimere una regola per cui "se un fornitore ha un codice identificativo e un dato prodotto è associato a una matricola, allora un dato è associato a un produttore". Un programma può dedurre che tutte le richieste di informazioni relative a un sensore devono essere inoltrate per posta all'indirizzo del fornitore. Una macchina non comprende alcuna di queste informazioni ma è in grado di manipolarne i termini in modo da avere un significato per gli utenti umani. Una volta definita l'ontologia, quindi, risulta necessario avere un modo per interrogarla ed estrarre le informazioni che servono. Il modo per fare questo è di renderla disponibile all'interno di un query engine.

Un query engine (motore per esecuzione di interrogazioni) è un sistema in grado di processare una specifica richiesta, detta query, effettuata in un linguaggio standard e utilizzata sia nei database relazionali sia nelle basi di conoscenza di tipo RDF (o linked data). È in grado anche di ricavare informazioni a partire da dati in formato linked e descritti da un'ontologia. Per ottenere quest'ultimo risultato è necessario che il query engine utilizzi un Rule-based Reasoner (ragionatore basato su regole), ovvero un sistema in grado di estrarre nuova informazione a partire da informazioni strutturate applicando esaustivamente un insieme di regole.

In questo capitolo viene descritto il funzionamento del sistema di interrogazione utilizzato per accedere ai dati dell'ontologia.

SPARQL è un acronimo ricorsivo che sta per SPARQL Protocol and RDF Query Language [21]. Si tratta di un linguaggio dichiarativo pensato per interrogare informazioni memorizzate in formato RDF con lo scopo di raccogliere informazioni da basi di conoscenza distribuite sul web. SPARQL è parte integrante del progetto Web Semantico del W3C e, usando le parole di Tim Berners-Lee, "provare a usare il Web Semantico senza SPARQL è come provare a usare un database relazionale senza SQL".

Sia SPARQL che SQL sono entrambi linguaggi dichiarativi usati per esprimere richieste a un query engine. SQL è usato per ricercare informazioni memorizzate in forma di tabelle all'interno di database relazionali. Tramite SQL è possibile selezionare, all'interno di una tabella, una specifica riga e il valore contenuto una delle colonne della tabella.

```
@prefix ab: <http://learningsparql.com/ns/addressbook#> .
@prefix d: <http://learningsparql.com/ns/data#> .

d:i0432 ab:firstName "Richard" .
d:i0432 ab:lastName "Mutt" .
d:i0432 ab:homeTel "(229) 276-5135" .
d:i0432 ab:email "richard49@hotmail.com" .

d:i9771 ab:firstName "Cindy" .
d:i9771 ab:lastName "Marshall" .
d:i9771 ab:homeTel "(245) 646-5488" .
d:i9771 ab:email "cindym@gmail.com" .

d:i8301 ab:firstName "Craig" .
d:i8301 ab:lastName "Ellis" .
d:i8301 ab:email "craigellis@yahoo.com" .
d:i8301 ab:email "c.ellis@usairwaysgroup.com" .
```

Figura 10 esempio di documento RDF tratto da [22]

In un ambiente distribuito ed eterogeneo come il web, dove è possibile che le stesse informazioni siano memorizzate in maniera differente, non sarebbe possibile utilizzare le stesse regole per estrarre le informazioni. Infatti non ci sono regole precise per definire una rappresentazione di una informazione [23].

Come detto, invece, SPARQL lavora usando dati in formato Linked Data, dove l'informazione è rappresentata in forma di grafi. Per poter gestire questa caratteristica, quindi, l'approccio risulta essere basato sul paradigma del pattern-matching. Attraverso l'individuazione delle corrispondenze tra il grafo su

cui si sta eseguendo la query e i vincoli imposti dalla query stessa, vengono estratti i risultati e legati i valori selezionati alle variabili definite nella query.
In Figura 10 è mostrato un esempio di una semplice rubrica tratto da [22]. Se, a partire da questi dati, si volesse sapere qual è l'email di "Craig", come mostrato in Figura 12, sarebbe sufficiente indicare nel campo "WHERE" della query le relazioni che devono essere verificate per poter risalire al risultato che ci interessa.
In questo caso le persone della rubrica sono identificate univocamente da un numero, e a ciascuno di essi sono associate alcune proprietà. Nella query SPARQL, dove il punto interrogativo a inizio parola identifica una variabile, in Figura 11 possieda due proprietà di cui una ha il vincolo di essere legata a un letterale specifico, ovvero il nome della persona di cui si vuol conoscere l'email. Nel campo "SELECT" è indicato quale informazione riportare in output.

```
PREFIX ab: <http://learningsparql.com/ns/addressbook#>

SELECT ?craigEmail
WHERE
{
     ?person ab:firstName "Craig" .
     ?person ab:email ?craigEmail .
}
```

Figura 11 query SPARQL per ricavare l'email di Craig a partire dal grafo in Figura 10

In Figura 12 è riportato il risultato della query in cui si vede che la corrispondenza con la proprietà "ab:email" comporta due risultati come risulta essere dal grafo di origine.

```
-------------------------------
 craigEmail
===============================
 "c.ellis@usairwaysgroup.com"
 "craigellis@yahoo.com"
-------------------------------
```

Figura 12 output della query mostrata in Figura 11

SPARQL è progettato per interrogare e unire differenti sorgenti di dati sul web, per cui non ci sono problemi a interrogare sistemi con sorgenti di dati distribuiti. Contrariamente a SQL, SPARQL e RDF si basano su un modello di dati standardizzato che permette di semplificare l'esplorazione ed estendere i modelli dei dati esistenti in un'ottica di dati aperti.

5.1 Principali modalità di inserimento e accesso ai dati

SPARQL fornisce diversi modi con cui interrogare una base di dati con lo scopo di eseguire diverse azioni o recuperare risultati in forme diverse (DuCharme, 2011):

- *SELECT*: è la principale forma di query usata. Permette di richiedere dati a una collezione di informazioni. Gli elaboratori di query tipicamente mostrano il risultato dell'interrogazione con SELECT sotto forma di tabella dove è associata a una colonna tutte o un sottoinsieme delle variabili utilizzate per individuare i risultati;
- *CONSTRUCT*: ha come output delle triple. Con questo tipo di query si possono estrarre triple dalle sorgenti dei dati senza cambiarle, oppure costruire nuove triple a partire dai dati estratti. Questo permette di copiare creare e convertire dati RDF;
- *ASK*: chiede all'elaboratore di query se lo schema del grafo contenuto nella query descrive o no un insieme di triple all'interno del particolare database. Questo tipo di query sono utili per esprimere regole su condizioni che devono o non devono essere verificate nei dati. Queste regole possono essere utilizzate per automatizzare il controllo della qualità e del processo di analisi dei dati;

- *DESCRIBE*: con questa query si richiede di descrivere una particolare risorsa. L'elaboratore della query determina autonomamente, come descritto nelle specifiche di SPARQL, quali triple utilizzare come risposta scelte tra quelle che hanno la risorsa come soggetto. Le risposte risultano essere inconsistenti tra le varie implementazioni, cosa che rende questo tipo di query poco popolare.

Tutte e quattro le forme di query, come detto, sono in grado di compiere il loro lavoro individuando una corrispondenza tra grafi, anche se presentano i risultati in formati diversi. Di seguito è trattata più approfonditamente sola la forma SELECT anche perché le altre forme risultano essere più rare.

Una query SPARQL è caratterizzata da una struttura composta da diverse parti (DuCharme, 2011) (Harris & Seaborne, 2012):

- *le dichiarazioni dei prefissi*: per abbreviare gli URI delle risorse e per applicare la notazione QName, prima della vera e propria query, viene posta, opzionalmente, la parola chiave "`PREFIX`" seguita dalla stringa che sostituisce l'URI e URI stesso;
- *la clausola di indicazione dei risultati*: di seguito alla parola chiave "`SELECT`" vengono indicate le variabili utilizzate nella query di cui si vuole conoscere il risultato;
- *la definizione del dataset*: tramite le parole chiave "`FROM`" e "`FROM NAMED`" seguite da un URI è possibile specificare uno o più grafi RDF su cui eseguire la query;
- *lo schema della query*: identificata dalla parola chiave "`WHERE`" è presente la parte centrale dalla query. In questa area è possibile specificare uno schema di relazioni che le triple del grafo RDF, che si sta interrogando, devono soddisfare per poter essere selezionate per l'output. È identificato anche come *graph pattern* ed è espresso in formato Turtle;
- *i modificatori del risultato*: l'ultima parte della query prevede degli operatori opzionali, e usabili in concerto, per poter riorganizzare i risultati ottenuti:
    - *LIMIT e OFFSET*: queste parole chiave seguite da un valore numerico permettono di limitare il numero dei risultati da presentare in output nel primo caso, o, nel secondo caso, di scartare i primi risultati;
    - *ORDER BY*: permette di ordinare i risultati in modo crescente o discendente per il valore di una delle variabili della soluzione;
    - *GROUP BY*: permette di aggregare i risultati secondo i valori della variabile specificata. Di conseguenza è necessario definire come aggregare i risultati delle altre variabili portate in soluzione. Inoltre l'aggregazione permette di eseguire altre operazioni sui risultati sui raggruppamenti dei risultati come operazioni sui valori numerici delle variabili (calcolo di valori medio, minimo e massimo), oppure il conteggio delle occorrenze.

```
PREFIX e: <http://example.org/ns/expenses#>

SELECT ?description (SUM(?amount) AS ?mealTotal)
FROM <http://my_expenses/meal>
WHERE
{
     ?meal e:description ?description ;
           e:amount ?amount .
}
GROUP BY ?description
```

Figura 13 semplice query SPARQL di esempio

In Figura 13 **Errore. L'origine riferimento non è stata trovata.**è mostrata una query di esempio, formata da solo due triple, in cui sono mostrate alcune delle cose descritte fin qui. All'interno del grafo RDF "<http://my_expenses/meal>" vengono selezionati gli elementi che contengono le proprietà "e:descritpion" e "e:amount". I risultati vengono aggregati secondo i valori della variabile "?description" e viene calcolata la somma dei valori di "?amount" per cui il valore corrispondente di "?description" è identico. Ovvero, se la descrizione consiste in "colazione", "pranzo", "cena" è possibile sapere la distribuzione delle spese tra i diversi pasti.

Tutti i graph pattern sono espressi usando il formato Turtle di RDF. Le variabili sono indicate dal simbolo"?" e possono essere usate per sostituire qualsiasi elemento di uno statement RDF; anche tutti gli elementi di una tripla possono essere composti da variabili.

I possibili tipi di graph pattern che sono usati in query SPARQL sono (Harris & Seaborne, 2012):

- *Basic Graph Pattern*: è formato da un insieme di triple RDF che devono essere considerate insieme. In questo modo è possibile selezionare specifici rami del grafo RDF. Dopo che è stato trovato un riscontro tra la struttura della query e del grafo RDF interrogato, le variabili sono legate ai valori specifici trovati producendo la soluzione;
- *Group Graph Pattern*: è un insieme di uno o più graph pattern delimitati da parentesi graffe. Tutti i pattern devono essere avere un riscontro sulla stessa parte di grafo per far si che possa essere selezionata per la soluzione;
- *Optional Graph Pattern*: è un insieme di graph pattern opzionali che possono o non possono fare match. È indicato dalla parola chiave OPTIONAL e nel caso non dovesse verificarsi riscontro la soluzione resta invariata lasciando semplicemente le variabili non legate a uno specifico valore;
- *Alternative Graph Pattern*: è costituito da un insieme di due o più graph pattern dove vengono selezionate le soluzioni per ogni pattern preso singolarmente, e dove il risultato è costituito unendo i singoli risultati. Questo graph pattern è identificato dalla parola chiave UNION inserita tra i singoli pattern;
- *Patterns on Named Graphs*: si tratta di un graph pattern che viene eseguito su di uno specifico grafo RDF identificato da un URI. Inoltre il nome del grafo può essere a sua volta identificato tramite una variabile e quindi essere parte della soluzione cercata. Questo tipo di pattern è identificato dalla parola chiave GRAPH.

È possibile specificare diversi tipi di schemi più complessi anche combinando tra loro semplici schemi.

5.2 Definizione di query SPARQL per l'accesso ai dati

A partire dall'ontologia precedentemente definita, sono state ipotizzate una serie di query per l'accesso alle informazioni in essa contenute.
Si è partiti col definire le query per accedere alle descrizioni degli Urban Dataset. In particolare si sono ipotizzate le query, mostrate di seguito, per recuperare la lista delle singole sottoclassi di Urban Dataset, la lista delle istanze e le proprietà di ciascuna di esse.
La query seguente serve per recuperare le sottoclassi della classe UrbanDataset:

```
prefix owl:<http://www.w3.org/2002/07/owl#>
prefix :<http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#>

select ?s
where {
      ?s rdfs:subClassOf :UrbanDataset.
}
```

Come si può vedere, la query inizia con la definizione di due prefissi usati nella query vera e propria. Nel campo *where* è presente un'unica tripla dove il soggetto (la nostra incognita) deve essere una classe che sia sottoclasse di *UrbanDataset*.

Il risultato è qualcosa di simile alla tabella seguente:

http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#AggegatedUrbanDataset
http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#AnomalyUrbanDataset
http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#EnergyUrbanDataset
http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#EnvironmentUrbanDataset
http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#SecurityUrbanDataset
http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#TrafficUrbanDataset

In questo caso si suppone che siano state definite alcune sottoclassi e che come risultato venga mostrata la lista di tutte quelle definite nell'ontologia al momento della richiesta.

La query seguente serve per recuperare le istanze della classe *TrafficUrbanDataset*:

```
prefix owl:<http://www.w3.org/2002/07/owl#>
prefix :<http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#>

select ?s
where {
      ?s a :TrafficUrbanDataset,
           owl:NamedIndividual.
}
```

La classe *UrbanDataset* non ha istanze. In questo esempio viene mostrata la query per una delle sottoclassi. Essa è composta da due triple con medesimo soggetto e predicato ma diverso oggetto, ovvero è definito che la nostra incognita sia della classe *TrafficUrbanDataset* e anche della classe *owl:NamedIndividual* che identifica le istanze di una classe. Il risultato sarà, parimenti, una lista di URI, in questo caso delle istanze della classe *TrafficUrbanDataset*.

Per visualizzare le proprietà possedute da una istanza di Urban Dataset è stata prevista la seguente query:

```
prefix owl:<http://www.w3.org/2002/07/owl#>
prefix :<http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#>

select ?p ?o
where {
      :InstantaneousFreeParking ?p ?o
}
```

Anche in questo caso è stata usata una query composta da una sola tripla. Tale richiesta elenca tutte le coppie proprietà oggetto associate all'istanza. Di seguito è mostrata la tabella di output della richiesta:

p	o
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#TrafficUrbanDataset
http://www.w3.org/1999/02/22-rdf-syntax-	http://www.w3.org/2002/07/owl#NamedIndivid

ns#type	ual
http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#HasUrbanDataset	http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#FreeParking
http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#context	http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#context1
http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#hasAggregation	http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#instantaneous
http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#hasApplicationContext	http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#SmartMobility
http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#hasUrbanDatasetProperty	http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#FreeParkNum
http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#UrbanDataset_ID	"UrbanDataset D5b"^^<http://www.w3.org/2001/XMLSchema#string>
http://www.w3.org/ns/prov#wasAttributedTo	http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#SmartStreet

Come si può notare, a sinistra sono elencati gli identificativi delle proprietà, come indicato anche dalla prima riga dove sono presenti le indicazioni delle variabili della query a cui vanno associati i valori sottostanti, a destra i valori che assumono.

Oltre alle query predisposte per recuperare le informazioni relative agli Urban Dataset, sono state predisposte anche per gli altri concetti introdotti quali *Property*, *DataType*, *Aggregation*, *ApplicationContext*, *Producer* e *UrbanDatasetContext*. I meccanismi con cui sono realizzati sono simili a quelli esposti finora per recuperare istanze e sottoclassi.

Discorso analogo ma inverso è quello di sapere qual è la superclasse di una data classe. Ad esempio sapere di quale classe *TrafficUrbanDataset* è sottoclasse:

```
prefix owl:<http://www.w3.org/2002/07/owl#>
prefix :<http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#>

select ?o
where {
      :TrafficUrbanDataset rdfs:subClassOf ?o
}
```

Questo esempio è esattamente inverso al primo esempio e permette di ricostruire la struttura delle classi partendo dall'inverso. Stessa cosa può essere applicata alle istanze per ricostruire la classe di cui sono istanze e per sapere a quale soggetto è associata una particolare coppia predicato – oggetto.

Discorso simile può essere fatto per le proprietà, ovvero conoscere il ruolo che svolge una proprietà all'interno dell'ontologia. Per fare questo sono state previste delle query che partendo dal nome della proprietà permettono di ricavare i soggetti a cui sono applicate e quale associazione è presente tra soggetto e oggetto. Per esempio:

```
prefix owl:<http://www.w3.org/2002/07/owl#>
prefix :<http://www.semanticweb.org/disi/ontologies/2017/8/smart-city-ontology#>

select ?s ?o
where {
```

```
      ?s :hasDataType ?o.
}
```

Il risultato di questa query è la lista delle coppie formate dalle istanze possedenti la relazione *hasDataType* e l'oggetto di tale relazione.

Nel capitolo seguente è mostrato in che modo è stata organizzata la libreria di interrogazione che permettono di richiedere le informazioni ottenute dalle query qui presentate senza dover scrivere il codice SPARQL per ogni nuova richiesta.

6 Libreria di interfacciamento all'ontologia

Oltre ad aver ipotizzato delle query di accesso alle informazioni contenute nell'ontologia, si è provveduto a sviluppare una libreria software che permettesse 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.

6.1 Caratteristiche funzionali

La caratteristica principale che ha una libreria software è quella di nascondere delle complessità all'utilizzatore finale. Ovvero di semplificare l'accesso a delle capacità di elaborazione che altrimenti lo sviluppatore dovrebbe imparare autonomamente a richiedere attraverso l'apprendimento di interfacce di accesso o altri linguaggi. Il nostro caso fa riferimento proprio a quest'ultima situazione. 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, quindi, 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 vuol sapere e per questo risulta molto importante un buon grado di facilità di esplorazione a partire dalle interfacce fornite.

Un obiettivo necessario riguarda la possibilità di accedere alle informazioni dell'ontologia in remoto. I sistemi che accettano interrogazioni SPARQL operano su protocolli web 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 applicazione. Java è uno dei linguaggi più diffusi e conosciuti. Per garantire una facilità di utilizzo della libreria si è optato per questa scelta.

6.2 Descrizione dell'architettura

Partendo dalle caratteristiche appena descritte si è arrivati alla definizione dell'architettura. Per poter recuperare le informazioni presenti nell'ontologia, essa deve essere inserita all'interno di un server 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 Jena [24]. 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 del progetto del MiSE di cui questa attività è parte, da poter essere usata come base per la nostra libreria. 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. Dalla Figura 14 si può notare, 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.

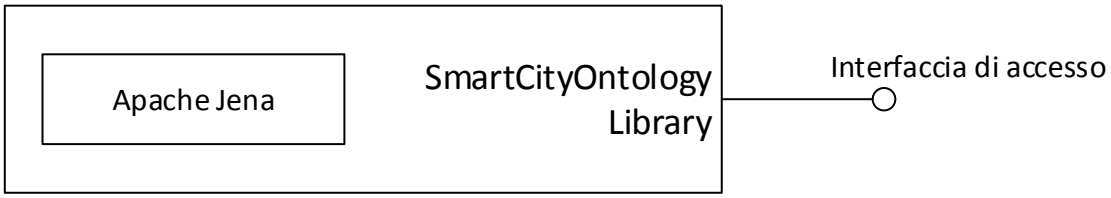


Figura 14 Architettura della libreria

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.

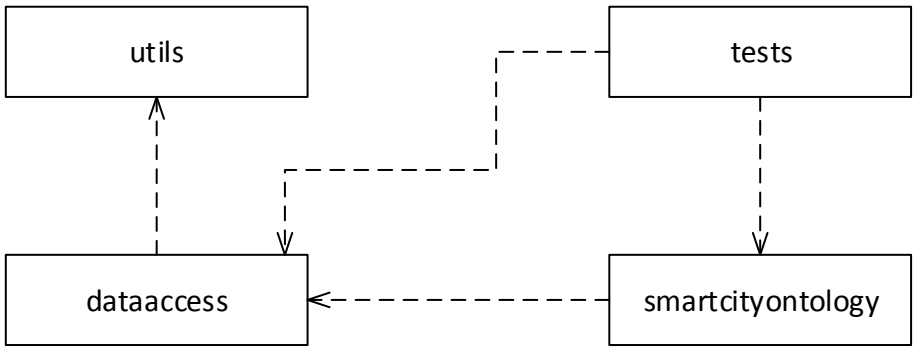


Figura 15 Diagramma dei package

In Figura 15 è mostrato il diagramma dei package. Nel package *utils* sono state inserite le informazioni per la configurazione della libreria stessa e sono stati creati degli scheletri di query per costruire dinamicamente l'interrogazione. Nel package *dataaccess* sono state inserite le classi per effettuare la connessione al sistema remoto e per costruire interrogazioni che, appoggiandosi sulle funzionalità del package utils, recupera le informazioni di una generica ontologia a partire da parametri specifici. 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. Sono stati implementati anche una serie di test automatici per verificare con il comportamento della libreria sia corretto.

Andando nel dettaglio: è prevista una classe che si occupa del collegamento al database remoto attraverso il pattern Singleton. Al momento dell'uso della libreria viene creata 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 è stata fatta per evitare di dover creare una diversa istanza di connessione ogni volta che si accede a un server diverso e per evitare di dover passare come riferimento la classe di connessione a ogni richiesta. 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 in sistema che costruisse le query utilizzando più livelli per rendere più facile l'esenzione nel caso l'ontologia dovese 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. Potrebbe essere semplice effettuare un filtro, attraverso la libreria o da parte dell'utilizzatore, dei dati

recuperati dal server in caso di non necessita della 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.

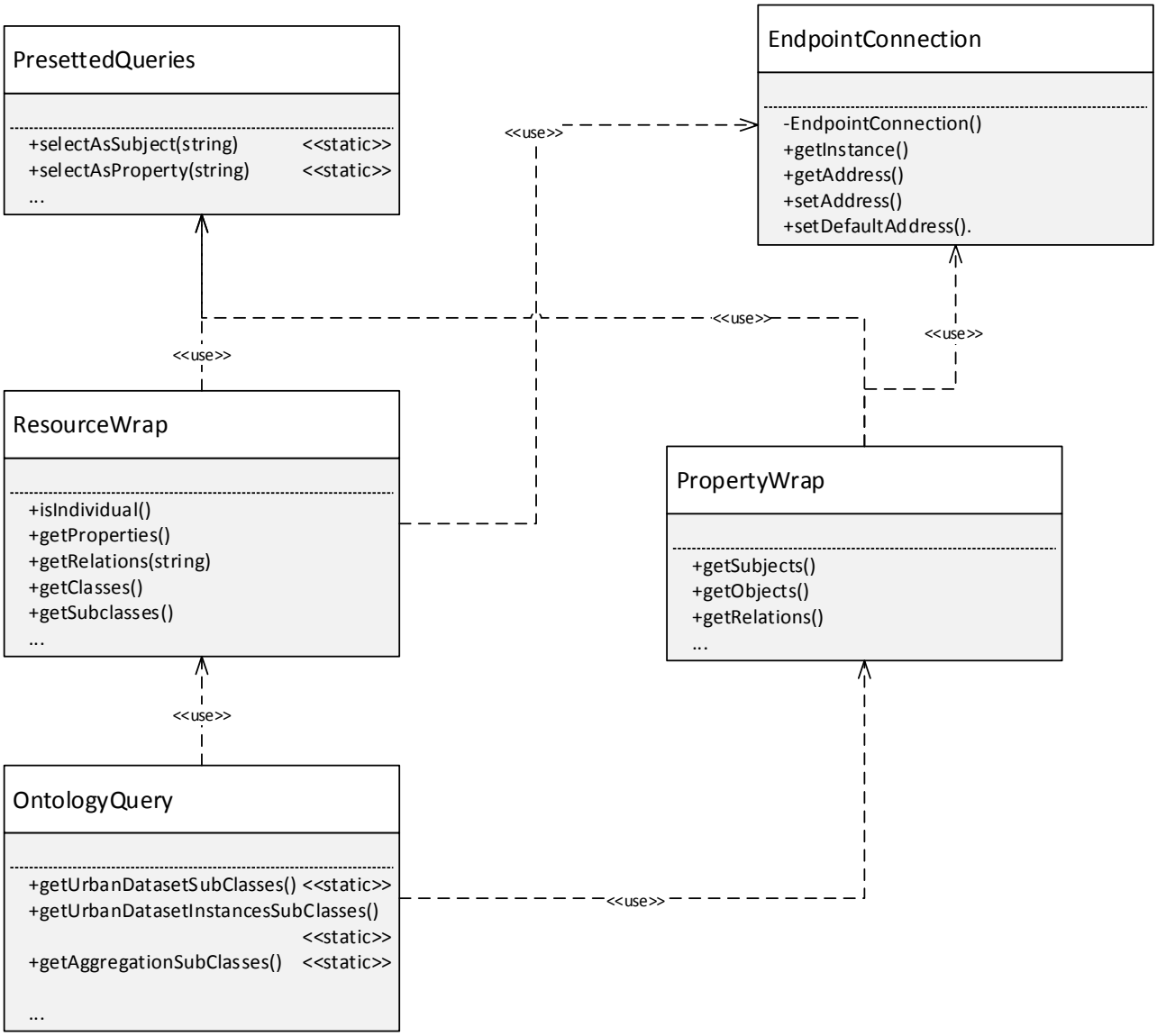


Figura 16 Diagramma delle classi

In Figura 16 è mostrato un diagramma delle classi parziale della libreria. Sono mostrati solamente alcuni dei metodi presenti. Si può notare che le query principali specifiche per l'ontologia sono definite dentro la classe *OntologyQuery*. Le classi *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. Come già detto, 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 che descrive le diverse operazioni, per facilitarne l'uso e l'integrazione all'interno di altri software.

7 Conclusioni

In questo documento si è descritto il lavoro svolto per l'individuazione di una ontologia comune e la progettazione di una libreria di interrogazione per tale ontologia, utili per favorire il reperimento di informazioni all'interno di una piattaforma ICT per una Smart City. Lo scopo di tale piattaforma è la condivisione di informazioni tra i vari ambiti applicativi di una città. La raccolta e la condivisione di informazioni tra i diversi ambiti può essere utile a creare sinergie tali da favorire nuovi e più importanti risultati nell'ambito del risparmio energetico e dell'efficienza di una città. Il vantaggio offerto da un'ontologia in tale ambito è la sua capacità di agire come framework unificante fra le diverse parti interagenti e fungere da base per l'interoperabilità tra sistemi realizzati con differenti modelli, paradigmi e linguaggi di programmazione. Può essere utile nel processo di identificazione dei requisiti e definizione di una specifica per sistemi IT. Favorisce l'interoperabilità e la scalabilità dei servizi in ambienti grandi e aperti e può essere usata facilmente come base per l'applicazione di regole logiche per l'inferenza automatica di nuova conoscenza.

Il lavoro svolto è partito dal lavoro svolto lo scorso anno di progetto e osservando nuove e più specifiche esperienze in ambito Smart City già avviate e si è cercato di trarre delle linee guida sul lavoro da svolgere. L'idea precedente era di utilizzare l'ontologia come punto di raccolta di tutte le informazioni presenti nella Smart City. Analizzando i casi d'uso ci si è resi conto che tale modalità era impraticabile dato l'elevata quantità di dati attesi. Si è deciso quindi di fare in modo che l'ontologia fungesse da indice dai dataset messi a disposizione della città e dei suoi utenti.

La ricerca si è quindi spostata sui progetti DIMMER, CityProtocol, Km4City e CITYkeys. Il primo progetto prevede la definizione di una piattaforma per la gestione di informazioni su edifici e consumi e sfrutta una ontologia come indice per accedere alle diverse informazioni. Il secondo prevede la definizione di una ontologia per raggruppare e ordinare gli indicatori della città. Il terzo prevede lo sviluppo di un'ontologia per descrivere le informazioni e la struttura di una città e dei suoi servizi. Il quarto progetto ha come obiettivo l'individuazione di tali indicatori. Il vantaggio che ha un approccio alla gestione della smart city attraverso l'uso di indicatori generati da aggregazione di dati è quello di permettere una maggiore resistenza alle interferenze degli errori di misura ma non danno garanzia e supporto alla reattività della risposta in quanto tali indici non sono in grado di fornire informazioni per poter reagire a eventi eccezionali, ma solo una valutazione della bontà dell'azione di gestione che l'indice cerca di misurare. Gli indici proposti nel quarto progetto, in particolare, hanno un'alta inerzia perché sono focalizzati a descrivere l'andamento generale della città e possono volerci anche diverse settimane prima che ci sia un cambiamento sensibile. L'approccio di utilizzare un'ontologia come indice per i dataset è sembrato quello più sensato.

In base a tale analisi si è deciso di spostare l'attenzione sulla definizione di una ontologia che permettesse di fornire un punto di accesso comodo per i dataset relativi alla Smart City. Si è cercato di individuare, attraverso i lavori dei diversi gruppi coinvolti, una struttura dei dati minima da premettere una catalogazione e ricerca.

Si è, pertanto, partiti dall'analisi degli scenari di riferimento su cui operare la raccolta delle informazioni utilizzate della piattaforma. Si sono individuate le caratteristiche comuni che contraddistinguono le informazioni da scambiare e gli interlocutori delle comunicazioni, ovvero la Piattaforma ICT principale e le diverse piattaforme dei singoli contesti applicativi. Tale analisi ha mostrato che esiste una struttura uniforme di comunicazione tra la piattaforma centrale e le piattaforme dei diversi contesti applicativi e ha perciò permesso di individuare una struttura di base per l'ontologia da usare al fine di organizzare la struttura delle informazioni.

Dopo di che si è passati all'individuazione dei dati da correlare all'informazione principale, ovvero agli Urban Dataset. Infatti, l'utilità degli dataset c'è se essi sono associati a informazioni sul contesto che li ha generati. Oltre a queste, sono state collegate anche altre informazioni sui produttori, ovvero il fornitore del servizio alla pubblica amministrazione, sull'aggregazione e informazioni sulle tecniche usate per effettuare l'aggregazione. Infine, queste informazioni, sono state ricondotte a una ontologia di più alto livello per descrivere la provenienza delle informazioni. L'ontologia fin qui realizzata descrive soltanto ad alto livello i

collegamenti tra i diversi concetti e necessita di una definizione completa di tutte le proprietà necessarie al fine di poter far sì che sia utilizzabile nella pratica.

In seguito a questo, sono state previste delle query per accedere alle informazioni in essa contenute per ipotizzarne un uso. A partire da queste query, si è sviluppata una libreria software che permette di interrogare l'ontologia senza vincolare l'utente a conoscere il linguaggio di interrogazione. Sono state previste un certo numero di richieste predeterminate e si è sviluppata l'architettura tenendo conto anche delle esigenze di estendibilità della stessa al fine di poter integrare nuove query anche più specifiche in seguito a una prima fase di uso sul campo. In ogni caso, per avere un risultato miglio è indispensabile conoscere come verrà effettivamente utilizzata una volta messa a disposizione, solo in quel caso se ne potranno conoscere bene i limiti e prevedere le necessarie estensioni. Questo ulteriore passo permetterà di valutare ed estendere sia l'ontologia sia la libreria in modo da poter essere utilizzabili in modo efficiente nella versione finale della piattaforma.

8 Riferimenti bibliografici

1. M. Uschold, M. Gruninger, “Ontologies: principles, methods and applications”, The Knowledge Engineering Review, vol. 11 (1996), pp. 93–136.
2. D. Oberle, “How Ontologies Benefit Enterprise Applications”, Semantic Web vol. 5 (2014), pp. 473-491.
3. H.S. Happel, S. Seedorf, "Applications of ontologies in software engineering." Proc. of Workshop on Sematic Web Enabled Software Engineering"(SWESE) on the ISWC. 2006.
4. DIMMER. Disponibile all’indirizzo: http://www.dimmerproject.eu.
5. City Protocol. Disponibile all’indirizzo: http://cityprotocol.org/
6. V Guallart, F Meneses, D Frogheri, D Ibañez, R Rubio, F Giralt, “City Anatomy: A Framework to support City Governance, Evaluation and Transformation”.
7. City Indicators- Supporting Ontologies. Disponiblile all’indirizzo: http://cityprotocol.org/city-indicators-supporting-ontologies/.
8. Open Sensors Platform. Disponibile all’indirizzo: http://cityprotocol.org/open-sensors-platform/.
9. R Rallo, “City Anatomy Ontology”.
10. M. S. Fox, “A Foundation Ontology for Global City Indicators”. Disponibile all’indirizzo: http://www.semantic-web-journal.net/system/files/swj1133.pdf.
11. GeoNames Ontology. Disponibile all’indirizzo: http://www.geonames.org/ontology/documentation.html.
12. Ontology of units of Measure (OM) Disponibile all’indirizzo: http://www.wurvoc.org/vocabularies/om-1.8/
13. Provenance Working Group, “The PROV Namespace”. Disponibile a: https://www.w3.org/ns/prov.
14. OWL-Time ontology. Disponibile all’indirizzo: https://www.w3.org/2006/time.
15. Km4City: Smart City Model and Tools Main Documentation Web Page. Disponibile all’indirizzo: http://www.disit.org/drupal/?q=node/6056
16. CITYkeys. Disponibile all’indirizzo: http://www.citykeys-project.eu/)
17. D. Brickley, L. Miller, “FOAF Vocabulary Specification 0.99”, 2000-2014. Disponibile all’indirizzo: http://xmlns.com/foaf/spec/.
18. Quantities, Units, Dimensions and Types Ontology (QUDT). Disponibile all’indirizzo: http://www.qudt.org/pages/HomePage.html.
19. H. Rijgersberg, M. van Assem, D. Willems, M. Wigham, J. Broekstra, J. Top, “Ontology of units of Measure (OM)”. Disponibile all’indirizzo: https://github.com/HajoRijgersberg/OM#om.
20. H. Rijgersberg, M. van Assem, J. Top: “Ontology of units of measure and related concepts”. Semantic Web Journal (SWJ) 4(1), 3–13 (2013).
21. E. Prud'hommeaux, “SPARQL vs. SQL – Intro”, Disponibile all’indirizzo: http://www.cambridgesemantics.com/semantic-university/sparql-vs-sql-intro.
22. B. DuCharme, Learning SPARQL, O'Reilly, 2011.
23. S. Harris e A. Seaborne, “SPARQL 1.1 Query Language”, W3C, 2013. Disponibile all’indirizzo: https://www.w3.org/TR/sparql11-query/.
24. Apache Jena. Disponibile all’indirizzo: https://jena.apache.org/.

Curriculum Vitae Michela Milano

Laureata in Ingegneria Elettronica presso l'Università degli Studi di Bologna riportando la votazione di 100/100 e lode, il 16 Marzo 1994.
Ha ottenuto il titolo di **Dottore di Ricerca** in Ingegneria Elettronica e Informatica il 30 Giugno 1998.
Dal 1 Luglio 1999 al 30 Giugno 2000 ha usufruito di una **borsa di studio** per lo svolgimento dell'attività di ricerca **post-dottorato** presso il Dipartimento di Ingegneria dell'Università degli Studi di Ferrara.
Dal 1 Luglio 2000 ha ricoperto il ruolo di **Ricercatore Universitario** presso la Facoltà di Ingegneria di Bologna afferendo al Dipartimento di Elettronica, Informatica e Sistemistica (DEIS).
Dal 1 Novembre 2001 ha ricoperto il ruolo di **Professore Associato nel settore concorsuale 9/H1 (ING-INF/05) – Sistemi di Elaborazione delle Informazioni** presso la Facoltà di Ingegneria di Bologna afferendo prima al Dipartimento di Elettronica, Informatica e Sistemistica (DEIS) poi al Dipartimento di Informatica – Scienza e Ingegneria DISI.

Posizione Attuale
Dal 1 Aprile 2016 ricopre il ruolo di **Professore Ordinario nel settore concorsuale 9/H1 (ING-INF/05) – Sistemi di Elaborazione delle Informazioni** presso la Facoltà di Ingegneria di Bologna afferendo al Dipartimento di Informatica – Scienza e Ingegneria presso cui svolge attività didattica e di ricerca, partecipando attivamente a convenzioni e progetti di ricerca.

Attività di Ricerca
L'attività di ricerca di Michela Milano riguarda i sistemi di supporto alle decisioni basati sulla Programmazione a Vincoli e la sua integrazione con tecniche di Programmazione Intera: in particolare, sono stati investigati sia aspetti metodologici sia aspetti applicativi con riferimento a numerose applicazioni quali scheduling, allocazione, cutting e packing, routing, aste combinatorie e recentemente per problemi decisionale e di ottimizzazioni legati allo sviluppo sostenibile e al processo di policy making.
In questo settore Michela Milano ha raggiunto visibilità internazionale e ha collaborazioni con diversi gruppi di ricerca, universitari e industriali.
È membro dei comitati di programma delle maggiori conferenze e workshop del settore e guest editor di diversi numeri speciali di riviste internazionali.
È **Editor in Chief** della rivista Constraints, è **Area Editor** di Constraint Programming Letters e **Area Editor** di INFORMS Journal on Computing.
È editor di cinque libri sull'ottimizzazione ibrida e autrice di più di 130 lavori su riviste e conferenze internazionali. È stata **program chair** di CPAIOR 2005 e CPAIOR 2010, di CP2012 e di CompSust2012. Su tali argomenti Michela Milano ha tenuto numerosi tutorial nelle maggiori conferenze italiane e internazionali quali: AI*IA99, PACLP2000, CP2000, IJCAI2001. Ha tenuto relazioni invitate a CP2013, ICAPS2004, INFORMS2002, INFORMS99, IFORS99 e numerosi seminari e relazioni invitate in centri di ricerca e industrie.
È membro dell'EurAI **Board** (European Association for Artificial Intelligence) e membro del **AAAI Council** (American Association of Artificial Intelligence). È membro dello **Steering Committee di CPAIOR**. È stata membro del Executive Committee della ACP Association of Constraint Programming, ed è membro del Consiglio Direttivo dell'Associazione Italiana per l'Intelligenza Artificiale AI*IA.
Ha partecipato a numerosi progetti di ricerca italiani ed Europei. È stata **coordinatrice** del progetto Europeo FP7 **ePolicy**, Engineering the Policy Making Life Cycle, 2011-2014 e partner del progetto Europeo FP7 **COLOMBO**, Cooperative Self-Organizing System for low Carbon Mobility at low Penetration Rates, 2012-2015, del progetto EU-FP7 **DAREED**: Decision Advisor for Energy Efficient Districts, 2013-2016 e del progetto EU-H2020 **OPRECOMP**: Open Transprecision Computing, 2017-2020. È stata principal investigator del **Google Focused Grant** on Mathematical Optimization and Combinatorial Optimization in Europe nel 2012. Inoltre le è stato assegnato il **Google Faculty Research Award** nel 2016 su integrazione di reti neurali profonde in modelli combinatori.

Curriculum Vitae Federico Chesani

Laureato in Ingegneria Informatica presso l'Università degli Studi di Bologna riportando la votazione di 98/100, il 17 Luglio 2002.
Ha ottenuto il titolo di **Dottore di Ricerca** in Ingegneria Elettronica, Informatica e delle Telecomunicazioni il 12 Aprile 2007.
Dal 1 Gennaio 2007 al 30 Marzo 2012 ha usufruito di alcune **borse di studio**, per lo svolgimento di attività di ricerca post-dottorato, bandite dal CINI (Consorzio Interuniversitario Nazionale per l'Informatica) e dall'Università di Bologna (Dipartimento DEIS).
Il giorno 1 Aprile 2012 ha preso servizio nel ruolo di **Ricercatore Universitario nel settore concorsuale 9/H1 (ING-INF/05) – Sistemi di Elaborazione delle Informazioni** presso la Facoltà di Ingegneria di Bologna, afferendo al Dipartimento di Elettronica, Informatica e Sistemistica (DEIS).
Nel Dicembre 2013 ha ricevuto **l'abilitazione** al ruolo di Professore di Seconda Fascia ("associato") nel settore concorsuale 9/H1(ING-INF/05), e nel Gennaio 2014 ha ottenuto l'abilitazione, sempre per il ruolo di Professore di Seconda Fascia, per il settore concorsuale 01/B1 (INF/01).

Posizione Attuale
Dal 1 Aprile 2012 svolge attività didattica e di ricerca presso la Scuola di Ingegneria e Architettura dell'Università di Bologna, e afferisce attualmente al Dipartimento di Informatica – Scienza e Ingegneria DISI, nel ruolo di Ricercatore Universitario a tempo indeterminato, partecipando attivamente sia a progetti di ricerca, che a progetti di trasferimento tecnologico.

Attività di Ricerca
Federico Chesani ha svolto la sua attività di ricerca prevalentemente nell'ambito dei sistemi esperti e di supporto alle decisioni basati su approcci a regole: in particolare, si è occupato sia di aspetti teorici legati ai sistemi a regole in logiche abduttive per gestire l'assenza di conoscenza e l'integrazione di conoscenza ontologica, sia ad aspetti maggiormente pratici legati all'applicazione di sistemi in presenza di conoscenza incerta e/o probabilistica. In particolare, nell'ambito dei numerosi progetti a cui ha contribuito, ha applicato sistemi a regole per il supporto alle decisioni in ambito sanitario (sia a livello italiano che europeo), "policy making", e manifatturiero/industriale. Nell'ambito di tale attività di ricerca, è co-autore di oltre 60 pubblicazioni, ed è stato invitato a tenere seminari e tutorial nell'ambito di conferenze internazionali.
Federico Chesani collabora attivamente con gruppi di ricerca nazionali e internazionali, e svolge attività di coordinamento e diffusione a livello nazionale e internazionale. E' membro di diversi comitati di programma di conferenze e workshop, e svolge con continuità attività di revisore sia per progetti nazionali ed europei, che per pubblicazioni su riviste scientifiche. Dal 2013 è membro del consiglio direttivo del Gruppo Ricercatori e Utenti Logic Programming (GULP).
Ha partecipato a numerosi progetti di ricerca italiani ed Europei, tra cui il progetto Europeo FP7 **ePolicy (**"Engineering the Policy Making Life Cycle", 2011-2014), il progetto Europeo FP7 **FARSEEING** (2012-2015), il progetto europeo FP5 **SOCS** (2002-2005), i progetti Nazionali PRIN/COFIN/FIRB **MASSIVE**, **SVP**, e **tocai.it**. Attualmente sta collaborando nel progetto Europeo H2020 **PreventIT** (2016-2018).