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

Ministero della Transizione Ecologica

Ricerca di Sistema elettrico

Sviluppo Piattaforma per la Governance dei Dati Urbani Energetici

C. Novelli, G. Santomauro, A. Frascella, A. Brutti, C. Petrovich,
F. Moretti, M. Chinnici, G. Ponti, S. Pizzuti

Report RdS/PTR2020/013

SVILUPPO PIATTAFORMA PER LA GOVERNANCE DEI DATI URBANI ENERGETICI

C. Novelli, G. Santomauro, A. Frascella, A. Brutti, C. Petrovich, F. Moretti, M. Chinnici, G. Ponti, S. Pizzuti

Aprile 2021

Report Ricerca di Sistema Elettrico

Accordo di Programma Ministero dello Sviluppo Economico - ENEA
Piano Triennale di Realizzazione 2019-2021 - II annualità
Obiettivo: Tecnologie
Progetto: Tecnologie per la penetrazione efficiente del vettore elettrico negli usi finali
Work package: Local Energy District
Linea di attività: Sviluppo Piattaforma per la Governance dei Dati Urbani Energetici
Responsabile del Progetto: Claudia Meloni, ENEA
Responsabile del Work package: Claudia Meloni, ENEA

Indice

Sommario

Questo è il rapporto tecnico, parte del WP1 “Local Energy District”, per il progetto “Tecnologie per la penetrazione efficiente del vettore elettrico negli usi finali“, **Linea di Attività 20 (LA 1.20)**, “**Sviluppo Piattaforma per la Governance dei Dati Urbani Energetici**”.

La LA 1.20 fa parte di un percorso, organizzato in tre annualità, pianificato come segue:

1. LA 1.19 (anno 2019): “Progettazione Framework per la Governance dei Dati Urbani Energetici“;
2. LA 1.20 (anno 2020): “Sviluppo Piattaforma per la Governance dei Dati Urbani Energetici”, che è la Linea di Attività descritta in questo report e prevede la progettazione, tramite definizione di un “caso studio pilota”, e l’avvio della fase implementativa della Piattaforma di Governance;
3. LA 1.21 (anno 2021): “Sperimentazione del Framework per la Governance dei Dati Urbani Energetici“.

La definizione del “caso studio pilota” è parte fondamentale dell’intero percorso: utilizzando l’analisi, i requisiti e gli studi compiuti nella prima annualità (2019), ha permesso di ultimare la progettazione e avviare il primo sviluppo della piattaforma agente su scala nazionale nella seconda annualità (2020), la cui implementazione finale e sperimentazione troveranno compimento nella terza annualità (2021).
Il caso studio pilota fornisce, quindi, una descrizione onnicomprensiva dei concetti e dei nuovi componenti necessari per abilitare la comunicazione interoperabile su scala nazionale, che viene dimostrata con l’implementazione di una piattaforma centrale di governance in grado di recuperare i dati urbani dalle piattaforme smart city agenti su scala urbana, tramite l’utilizzo di specifiche e formati condivisi. Il caso studio pilota, in particolare, descrive il flusso dati di uno specifico caso reale, ovvero quello che mette in connessione la “piattaforma urbana SCP Smart Village Casaccia” e la “piattaforma nazionale inter-SCP”.

Nel primo capitolo, introduzione di questo rapporto tecnico, si entrerà nel merito del lavoro svolto, fornendo una panoramica generale, poi si passerà nel secondo capitolo alla descrizione del caso studio pilota, entrando via via maggiormente nel dettaglio delle diverse attività e dei risultati ottenuti, nei successivi capitoli.

1 Introduzione

La Linea di Attività 1.20 (annualità 2020) è lo "Sviluppo Piattaforma per la Governance dei Dati Urbani Energetici" che consiste, in primis, nella realizzazione di un **prototipo di piattaforma agente su scala nazionale per la governance dei dati urbani energetici**, abilitante la comunicazione interoperabile con le piattaforme esistenti di gestione delle singole città/distretti, in ottica Smart City.

Tale "prototipo di piattaforma agente su scala nazionale per la governance dei dati urbani energetici" è stato denominato **inter-SmartCityPlatform (inter-SCP o iSCP)**, in quanto permette la connessione con diverse piattaforme di gestione città/distretti, con un duplice obiettivo:

1. Il monitoraggio su scala nazionale della gestione dei dati urbani energetici;
2. la comunicazione tra piattaforme Smart City di città diverse.

Il monitoraggio centralizzato, su scala nazionale, abilita l'osservazione e il confronto di diverse realtà urbane dal punto di vista energetico, con tutti i vantaggi che questa conoscenza comporterebbe;
la piattaforma iSCP, una volta pronta, potrà essere strumento per abilitare tale osservazione.

In questa direzione, è doveroso ricordare che i laboratori ENEA, dopo aver effettuato la prima analisi del problema, hanno deciso di riutilizzare due importanti risultati del triennio precedente della Ricerca di Sistema Elettrico [8], nell'ambito dell'obiettivo "Piattaforma ICT per lo Smart District":

- le specifiche per l'interoperabilità, *Smart City Platform Specification versione 1.0 (SCPS)* [9],
- il prototipo di piattaforma ICT per lo Smart District / Smart City, *Smart City Platform (SCP)* [10].

Sia le specifiche SCPS che il prototipo SCP sono stati definiti (nel precedente triennio) su scala urbana, in particolare in questo prototipo era totalmente assente la capacità di far comunicare SCP differenti e non era stato definito un formato di rappresentazione dei dati urbani su scala nazionale.

Nel nuovo triennio 2019-2021 si vuole evolvere la singola piattaforma SCP su scala urbana, a un framework agente su scala nazionale che permette la convivenza e il monitoraggio di una moltitudine di SCP, andando ad aggiungere tutte le funzionalità e le risorse necessarie al nuovo ambizioso obiettivo.

Questo report, relativo alla linea di Attività 1.20, descrive i **risultati** raggiunti in questa seconda annualità:

- descrizione completa del **caso studio pilota su scala nazionale**;
- il prototipo di **piattaforma inter-city versione 0.1** [11], secondo l'architettura del framework per la governance dei dati urbani su scala nazionale, per l'integrazione di piattaforme cittadine, con interfacce utente che consentono la gestione e il monitoraggio del traffico dei dati su scala nazionale;
- la rappresentazione e la comunicazione interoperabile di KPI su scala nazionale e dei dati ritenuti più significativi, utilizzando le specifiche **SCPS versione 2.0** [2];
- la metodologia di supporto all'utente per l'**adesione al framework, versione draft**.

Si noti che lo sviluppo dello SCP-Scheduler, inizialmente previsto per la seconda annualità (LA1.20) è stato spostato nella terza annualità (LA1.21); ciò è da attribuirsi alla definizione del caso studio pilota, presentato in questo report, è che è una descrizione propedeutica alla progettazione del componente SCP-Scheduler poiché fornisce tutti i requisiti e tutte le funzionalità per abilitare la comunicazione tra SCP e inter-SCP.

1.1 Sotto-Attività e Gantt

Le principali **sotto-attività** che hanno portato a questi risultati sono le seguenti (segue GANTT con le rispettive tempistiche relative):

1. Integrazione SCP-PELL: avendo avuto successo lo studio di fattibilità (si veda report LA 1.19), si è proceduto con l'implementazione relativa;
2. SCPS 2.0: evoluzione delle specifiche SCPS;
3. Replicabilità e Setup: preparazione delle macchine virtuali ed evoluzione della configurazione dei componenti del prototipo SCP/iSCP;
4. Guida Utente (draft) per l'Adesione al framework SCP;
5. Progettazione e Sviluppo SCP-DASH per la visualizzazione dei dati e dei KPI;
6. Definizione Caso Studio Pilota: descrivente la comunicazione interoperabile su scala nazionale;
7. Prototipo iSCP 0.1: prototipo di piattaforma inter-city finale, agente su scala nazionale;
8. Report LA1.20: stesura del presente report.

Si noti che la prima parte del lavoro svolto in questa LA 1.20 (annualità 2020) è direttamente connessa a quello pregresso della LA 1.19 (annualità 2019), mentre la seconda parte è relativa all'avvio dello sviluppo nella direzione di arrivare alla fase sperimentale con la LA 1.21 (annualità 2021).

Linea Attività	Sotto Attività	2020												2021			
		gennaio	febbraio	marzo	aprile	maggio	giugno	luglio	agosto	settembre	ottobre	novembre	dicembre	gennaio	febbraio	marzo	aprile
LA 1.20	Integrazione SCP-PELL																
	SCPS 2.0																
	Replicabilità e Setup																
	Guida Utente (draft) per l'Adesione																
	Progettazione e Sviluppo SCP-DASH																
	Definizione Caso Studio Pilota su scala nazionale																
	Prototipo iSCP 0.1																
	Report LA 1.20																

Figura 1 - Gantt relativo a Linea di Attività 1.20

1.2 Organizzazione del rapporto tecnico

Questo rapporto tecnico relativo alla LA 1.20 è organizzato come segue:

Il **Capitolo 1** è questa **Introduzione** al lavoro svolto nell'annualità 2020.

Nel **Capitolo 2**, è riportato il **Caso Studio Pilota**, che è una descrizione che ben rappresenta l'obiettivo finale del triennio: in esso sono evidenziati tutti i concetti chiave e componenti necessari per abilitare la comunicazione interoperabile da una piattaforma smart city urbana funzionante (la SCP-Casaccia) verso la piattaforma inter-city agente su scala nazionale (la inter-SCP), per ottenere una visualizzazione dati finale.

Il caso studio pilota è stato definito grazie al lavoro svolto nella LA 1.19 (annualità 2019), di cui riutilizza la maggior parte dei risultati nella sua descrizione:

- è stata seguita l'architettura del framework su scala nazionale definita ed è stato sviluppato il singolo flusso dati da SCP a inter-SmartCityPlatform (inter-SCP);
- è stato descritto nello specifico la comunicazione interoperabile basata su bridge;
- considera diversi scenari (su scala nazionale, su scala urbana e di integrazione con PELL con protocollo MQTT) e quindi lo stesso risultato portrà essere riutilizzato in ognuno di questi casi applicativi;
- ha sfruttato le migliorie al componente SCP-GUI (in particolare l'introduzione del nuovo utente SERVICE, l'associazione "1 account con N solution/service" e il nuovo protocollo di comunicazione "attivo" per definire produzioni usando un modulo bridge);
- ha utilizzato lo "stato dell'arte dei servizi per la visualizzazione di dati" per individuare un set di view e di grafici (chart) con cui sviluppare la dashboard della piattaforma inter-city.

Il capitolo termina con una prima breve presentazione del **prototipo inter-SCP nella sua versione 0.1**.

Nel **Capitolo 3**, viene descritto l'intervento per evolvere ed arricchire le **specifiche SCPS 2.0**, sia dal punto di vista architetturale, che del protocollo di trasporto via web service, sia dal punto di vista della rappresentazione dei dati con la definizione di nuovi UrbanDataset.

Nel **Capitolo 4**, viene descritta la progettazione e lo sviluppo del **nuovo componente SCP-DASH**, interfaccia utente per la visualizzazione di dati attraverso tabelle e grafici, che verrà utilizzato per definire una prima dashboard per la "governance dei dati urbani" su scala nazionale.

Nel **Capitolo 5**, viene descritto il lavoro di implementazione software per effttuare l'**integrazione tra la piattaforma SCP e la piattaforma PELL**, implementando lo studio di fattibilità svolto in precedenza.

Nel **Capitolo 6**, viene riportato il percorso **di adesione** per una SCP che vuole connettersi alla inter-SCP e quindi divenire parte del Framework distribuito su scala nazionale.

Nel **Capitolo 7**, verranno riportate le **Conclusioni** relative al lavoro di questa annualità, in riferimento al lavoro della precedente (LA 1.19, 2019) e alla successiva (LA 1.21, 2021) nella quale si proseguirà il lavoro al fine di implementare il caso studio pilota nella sua totalità e portare così il prototipo iSCP alla versione 1.0 finale.

In **Appendice A**, viene riportato il lavoro eseguito per favorire la replicabilità del prototipo SCP/iSCP, agevolando la fase di installazione e configurazione. E' stato scelto di collocare questa descrizione in appendice per favorire la leggibilità del presente report: il lavoro fatto per migliorare la replicabilità consiste, infatti, in una serie di interventi molto tecnici che richiedono conoscenza di diversi aspetti tecnologici, in primis del sistema operativo Linux.

In **Appendice B**, vengono riportate alcune tabelle di codici, frutto del lavoro svolto sulla specifica *SCPS Information 2.0*, descritto nel par.3.2.

In **Appendice C**, viene riportato il listato di un file di configurazione report per il componente SCP-DASH (a completamento della descrizione data nel cap.4). Tale file testuale permette di specificare ogni tabella e grafico contenuti in un report visualizzato nella dasboard, con la possibilità di indicare l'UrbanDataset in input e il mapping tra input e output.

1.3 Gruppi di lavoro

Nella LA 1.20, hanno partecipato e operato in sinergia i seguenti gruppi di lavoro:

- ENEA TERIN-SEN-SCC
  - supervisione e coordinamento generale di tutte le attività;
  - definizione specifiche SCPS 2.0;
  - definizione architettura framework e progettazione software di ogni componente;
  - definizione del caso studio pilota;
  - implementazione e configurazione prototipo iSCP 0.1;
  - implementazione integrazione SCP-PELL;
- ENEA TERIN-SEN-CROSS
  - definizione specifiche SCPS 2.0;
  - preparazione architettura hardware e infrastruttura ICT per il deploy di macchine virtuali;
- ENEA TERIN-ICT-HPC
  - consolidamento e miglioramento replicabilità, setup, configurazione del prototipo SCP/iSCP;
- Heartwood Labs (società esterna)
  - Sviluppo software relativo alle interfacce grafiche web per l'utente: consolidamento ed evoluzione SCP-GUI, sviluppo nuovo componente SCP-DASH.

Collegamento con altre linee di attività:

- Università di Bologna (LA 1.22 [19])
  - sviluppo strumenti software relativi all'Ontologia dove vengono definiti gli UrbanDataset;
- Politecnico di Milano (LA 1.24 [20])
  - definizione di una metodologia per l'individuazione ed elaborazione di KPI per la valutazione dei consumi energetici e di altre performance smart;
- Università di Insubria (LA 1.26)
  - definizione dell'Urban Check Up Model per effettuare indagini preliminari verso i referenti di municipalità e aziende in vista di una transizione tecnologica in ambito Smart City;
- PELL (LA 1.28 [17])
  - implementazione integrazione SCP-PELL.

2 Caso Studio Pilota

Il **caso studio pilota** è una descrizione specifica dell'utilizzo reale delle metodologie e delle tecnologie in oggetto che, per le sue stesse caratteristiche, viene considerato come caso fondamentale: è stato definito in modo tale da coprire una casistica più ampia possibile e non solo il caso concreto che descrive direttamente. Implementando il caso studio pilota potremo, successivamente, configurare un numero consistente di casi ad esso riconducibili, che necessitano degli stessi componenti e utilizzeranno il medesimo approccio.

Come sappiamo, lo scopo finale della linea di attività 1.20 è quella di ottenere una piattaforma inter-SmartCityPlatform (inter-SCP o iSCP) per la governance su scala nazionale e quindi in grado di raccogliere dati urbani dalle diverse piattaforme cittadine, utilizzando specifiche pubbliche che definiscono, centralmente, le tipologie di dataset (denominati UrbanDataset o UD) che saranno oggetto di scambio (tali definizioni di UrbanDataset vengono registrate nell'Ontologia centrale [4]).

La soluzione ideata consiste nell'utilizzo delle specifiche SCPS e del prototipo di piattaforma Smart City Platform (SCP) che era stato ideato inizialmente per ricevere, passivamente, dati dalle sorgenti della città, ed estenderlo per recuperare, attivamente, i dati presso altre SCP. In alternativa, ciò che si sarebbe potuto fare sarebbe stata la realizzazione di una nuova piattaforma inter-SCP creata ad hoc, per recuperare i dati dalle altre SCP e quindi ottenere infine due diversi prototipi e un canale di trasmissione ad hoc, poco riutilizzabile se non per il caso in oggetto.

Lo sforzo aggiuntivo che si è fatto è stato invece quello di generalizzare il caso SCP per permettere di:

- abilitare la inter-SCP a raccogliere attivamente dati dalle SCP;
- abilitare la SCP a raccogliere attivamente i dati dalle Solution verticali urbane;

continuando ad avere un unico prototipo configurabile per entrambi i casi e, quindi, molto più potente.

Per fare questo è stato necessario ideare un approccio di comunicazione che utilizzi uno SCP-Bridge (così come verrà spiegato nel par.2.3). Analizzando diversi scenari, si è giunti alla conclusione che lo SCP-Bridge potesse essere generalizzato ad una casistica più ampia di moduli software generici eseguibili da uno scheduler apposito; tali moduli software possono eseguire dunque diverse azioni:

- lettura/scrittura locale/remota di UrbanDataset (p.es. SCP-Bridge),
- combinazione di diversi UD in input in un singolo UD in output (p.es. DataFusion),
- esportazione di dati dal proprio sistema locale in formato UD,
- altre azioni, circoscrivibili in moduli software indipendenti, che possono anche non essere strettamente relativi alle specifiche SCPS.

Questi moduli software interni ad una SCP sono, in questo contesto, denominati **SERVICE**, e possono essere utilizzati (e riutilizzati) per implementare molti scenari, anche combinandoli tra loro.

L'esecutore o scheduler dei SERVICE, prende il nome di **SCP-Scheduler**, componente della SCP che è in grado di registrare i diversi moduli SERVICE, avviarli con una periodicità predefinita, fermarli, monitorarli, gestirli.

Tramite lo SCP-Scheduler e diversi moduli SERVICE sarà dunque possibile abilitare la comunicazione tra SCP e quindi, nel caso specifico, sarà possibile abilitare la comunicazione tra diverse SCP cittadine e la inter-SCP agente su scala nazionale; ma non solo.

Una volta che i dati saranno recuperati nella inter-SCP si potranno visualizzare attraverso un'apposita **SCP-Dashboard** (progettata partendo dallo studio "Stato dell'Arte Servizi di Visualizzazione Dati"svolto nella LA 1.19 [14], capitolo 7 del report). Anche in questo caso si è svolto uno sforzo aggiuntivo per non creare un cruscotto monouso ma per ottenere un componente riutilizzabile in una casistica più ampia. Con la SCP-

Dashboard sarà infatti possibile ottenere una visualizzazione dei dati, tramite tabelle e diagrammi (chart) di entrambi i casi:

- visualizzare i dati raccolti dalla inter-SCP relativi al caso nazionale;
- visualizzare i dati raccolti da una SCP urbana relativi al caso cittadino/distrettuale.

Anche per preparare i dati in input per la SCP-Dashboard, vedremo, risulterà utile utilizzare lo SCP-Scheduler.

Allo scopo di guidare lo sviluppo verso un'implementazione reale che potesse fungere da dimostratore finale dell'approccio, è stato dunque definito il **caso studio pilota** "*Da SCP-Casaccia a inter-SCP*", che prevede la comunicazione tra SCP-Casaccia (agente su scala urbana) e inter-SmartCityPlatform (iSCP) (agente su scala nazionale), tramite l'utilizzo di SCP-Scheduler, SCP-Dashboard, nonché di una serie di SERVICE creati per il caso in oggetto. Si raggiungerà questo obiettivo definendo una metodologia che sarà valida per connettere qualunque altra piattaforma Smart City agente su scala urbana alla inter-SCP, tramite un approccio che si basa sull'adesione alle specifiche pubbliche SCPS.

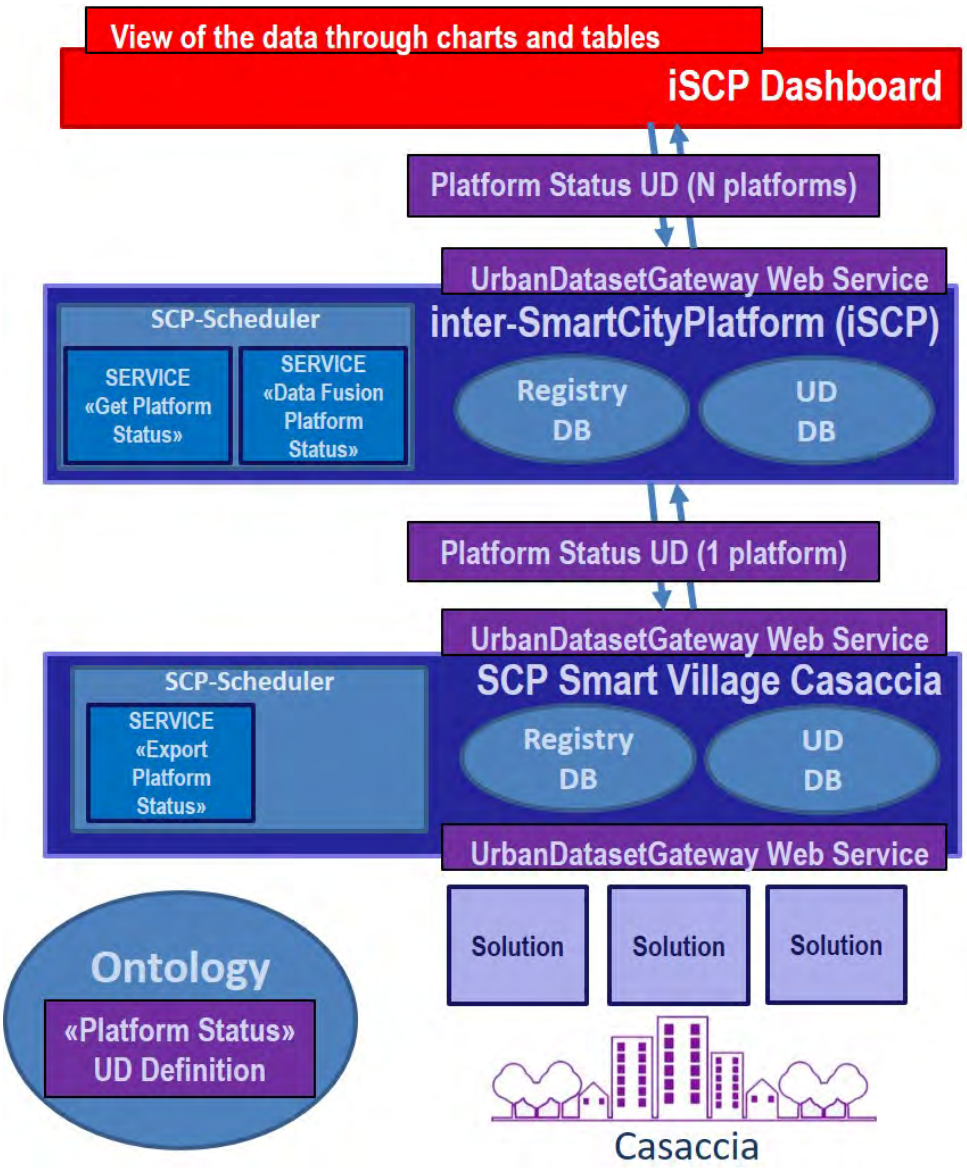

Figura 2 - Caso Studio Pilota "Da SCP-Casaccia a inter-SCP"

Il caso studio pilota "Da SCP-Casaccia a inter-SCP" è organizzato nelle seguenti diverse fasi (schematizzate nella precedente figura di insieme, dal basso verso l'alto):

1 definizione UrbanDataset (nell'Ontologia) per rappresentare i dati da trasmettere;

2 esportazione e pubblicazione dell'UD in una SCP reale e funzionante: verrà utilizzato allo scopo il prototipo SCP-Casaccia attualmente utilizzato per lo Smart Village Casaccia;

3 recupero dell'UD da parte della inter-SCP (iSCP) tramite SCP-Scheduler;

4 preparazione dei dati di input per la SCP-Dashboard, elaborando gli UD raccolti dalla iSCP: anche in questo caso si utilizzerà lo SCP-Scheduler per effettuare azione di DataFusion;

5 il recupero degli UD nella SCP-Dashboard per la visualizzazione dati finale.

Per la piena comprensione di questo testo si raccomanda la lettura del Report RdS/PTR(2019)/007, relativo all'annualità LA 1.19, propedeutica alla LA 1.20 corrente, in particolare dei capitoli:
2. "Architettura Framework", con riguardo al par. 2.2 "Architettura Concettuale inter-city";
3. "Comunicazione Interoperabile", con riguardo al par. 3.2 "SCP Bridge";
4. "Studio di Fattibilità Integrazione SCP – PELL", par. 4.3.4 "Task 1.4 Generalizzazione Bridge".

2.1 Definizione UD

Per il caso studio pilota è stato necessario definire un solo UrbanDataset (UD).

Tale UD è stato denominato "*Platform Status*", con questa descrizione: "*Informazioni dello stato di funzionamento della piattaforma all'istante di generazione dell'UrbanDataset*".

In altre parole, questo UD riporta una serie di informazioni di stato della piattaforma da cui è esportato; ciò che si intende fare, è inviare periodicamente informazioni di stato da una SCP alla inter-SCP.

Nel caso studio in oggetto, quindi, tale UD rappresenterà lo stato della piattaforma SCP-Casaccia che verrà inviato alla piattaforma inter-SCP che, idealmente, raccoglierà UD analoghi da tutte le altre piattaforme urbane che saranno a lei connessa. In caso di successo del caso studio pilota, infatti, si potrà replicare il procedimento e connettere altre piattaforme urbane e, successivamente, si potranno gestire altri UD relativi a KPI definiti su scala nazionale, che potranno essere implementati e inviati da SCP cittadine alla inter-SCP con lo stesso approccio.

La descrizione dettagliata dell'UD "*Platform Status*" e della relativa attività di definizione viene data nel par.3.2.2, assieme alla scheda di progettazione.

L'UD "*Platform Status*" trova oggi la sua definizione nell'Ontologia [13].

Una volta definito nell'Ontologia, l'UD può essere dichiarato come supportato sia nella SCP-Casaccia che nella inter-SCP, dove avrà DATASET-ID: "*PlatformStatus-1.0*" (secondo convenzione data nelle specifiche SCPS Collaboration [6]).

Si ricorda che la definizione centralizzata degli UrbanDataset viene sempre eseguita da ENEA (in particolare dal laboratorio CROSS) che analizza, certifica e registra queste definizioni nell'Ontologia centrale.

A questo punto l'UD "Platform Status" è registrato nell'Ontologia centrale ed è possibile scaricare il template in formato JSON, che verrà utilizzato nella successiva fase di esportazione.

2.2 Esportazione UD

Nel precedente paragrafo abbiamo descritto la prima fase del caso studio pilota, ovvero la definizione dell'UrbanDataset "Platform Status". La seconda fase è relativa all'esportazione dalla SCP-Casaccia delle informazioni di stato, rappresentate in formato JSON tramite questo UD.

Immaginiamo l'azione di esportazione come un singolo modulo software che:

- verifica lo stato generale del sistema;
- legge dal Registry della SCP le informazioni necessarie;
- esporta le informazioni in formato JSON con il template dell'UrbanDataset "Platform Status";
- pubblica l'UD esportato rendendolo disponibile alla inter-SCP.

Riportiamo un sottoinsieme di queste informazioni, come esempio concreto relativo alla SCP-Casaccia:

DataDescription: "Stato corrente della SCP Smart Village Casaccia (SCP Casaccia)"

PlatformID: "SCP-1"

PlatformName: "SCP Casaccia"

LogCode: "info"

LogDescription: "Funzionamento nella norma."

TotalManagedVerticalsCount: "12" (numero delle solution verticali gestite)

La descrizione completa dell'UD "Platform Status" viene fornita nel par.3.2.2.
Tutte le informazioni statiche e dinamiche possono essere recuperate o calcolate dalla banca dati Registry interna ad ogni SCP. Una volta valorizzato l'UD, esso deve essere validato con il JSON Schema.
Questa azione di esportazione dello stato corrente della SCP sarà implementata nella LA 1.21 (annualità 2021) in un apposito modulo software nominato "*Export Platform Status*".
Tale modulo, sviluppato in java, viene registrato sulla SCP come SERVICE e deve venire eseguito periodicamente. Il componente per effettuare la registrazione, esecuzione e in generale la gestione del ciclo di vita di ogni SERVICE è lo SCP-Scheduler (la cui progettazione e sviluppo verranno trattati nell'ultima annualità del triennio, 2021, nella LA 1.21).

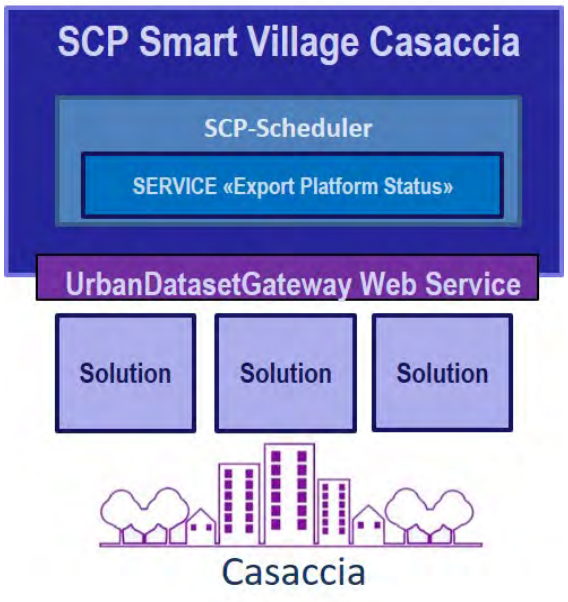

Figura 3 - SCP-Scheduler per esportare lo "Platform Status" UD

L'UrbanDataset "Platform Status" contiene una serie di informazioni relative allo stato della piattaforma urbana di cui si sta effettuando il monitoraggio tra cui lo stato generale di funzionamento, il numero di Solution gestite e di UrbanDataset prodotti o acceduti.
Una volta generato l'UD "Platform Status", questo viene pubblicato sulla SCP-Casaccia, inserendolo nella banca dati degli UrbanDataset tramite l'UrbanDatasetGateway e, in questo modo, esso viene reso immediatamente disponibile alla inter-SCP, a cui è stato configurato esplicitamente l'accesso.

La configurazione relativa alla frequenza di generazione dell'UD viene inserita tramite il componente SCP-GUI (configurando il SERVICE interno, in modo identico a una SOLUTION esterna) e viene utilizzata dallo SCP-Scheduler che in questo modo sa ogni quanto eseguire tale esportazione (p.es. una volta al giorno o una volta ogni ora, decisione che verrà presa nella fase finale di sperimentazione).

A questo punto l'UD è pronto sulla SCP-Casaccia per essere recuperato dalla inter-SCP.

2.3 Trasmissione UD

Prima di descrivere la parte del caso studio pilota che si occupa della trasmissione dell'UD dalla SCP-Casaccia alla inter-SCP, riprendiamo alcune delle considerazioni fatte nella precedente annualità (LA 1.19):

- i laboratori ENEA hanno ritenuto vantaggioso valorizzare e riutilizzare il risultato della piattaforma agente su scala urbana, denominata SmartCityPlatform (SCP), evolvendola al caso nazionale;
- le SCP sono installazioni indipendenti, per consentirne l'installazione sia sui server di ENEA ma, volendo, anche sui server delle municipalità o di chi ne facesse esplicita richiesta;
- la inter-SCP può essere implementata con una SCP che deve essere "evoluta" in quanto, al momento, due istanze del prototipo SCP di ENEA non sono in grado di comunicare tra loro;
- la comunicazione interoperabile tra SCP (nonché con altre piattaforme nazionali come PELL) può essere risolta con uno " SCP Bridge", ovvero un modulo software che agisce da sistema "attivo" tra due sistemi "passivi" (si veda report LA 1.19, par. 3.2, da cui riprendiamo la seguente figura).

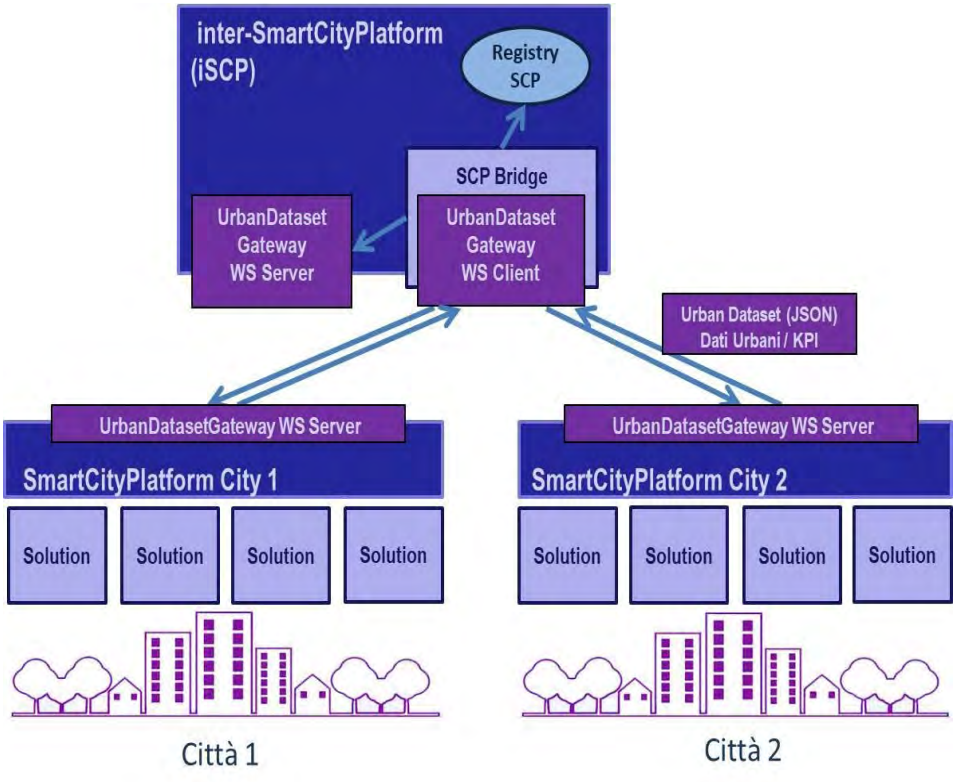

Figura 4 - SCP-Bridge per abilitare comunicazione tra iSCP e SCP (ripresa da LA 1.19)

A queste prime riflessioni ne sono seguite altre, riguardo:

- le specifiche SCPS che dovevano essere evolute per permettere di coprire il "caso nazionale" (si veda cap.3), anche con KPI appropriati (si veda LA 1.24 svolta in parallelo dal Politecnico di Milano);
- gestione dell'esecuzione periodica dello "SCP Bridge";
- generalizzazione del concetto di "SCP Bridge" con un generico modulo "SERVICE" che, oltre a compiere le azioni previste dal bridge, potesse compiere altre azioni.

Queste ultime riflessioni relative allo "SCP Bridge" sono state particolarmente importanti per arrivare alla **definizione più generale di un componente "Scheduler" interno alla SCP ("SCP-Scheduler")**: gestore di generici moduli software, denominati SERVICE, che può sfruttare il sistema di permessi in produzione e accesso della SCP (e le informazioni a corollario, per divenire parametrico), nonché il sistema di comunicazione che utilizza il web service *UrbanDatasetGateway* (basato su specifiche *SCPS Communication 2.0* [7]) portandolo il sistema di comunicazione "passivo", ovvero in attesa di richieste, ad essere anche "attivo", ovvero capace di recuperare UrbanDataset da altri sistemi.

Ricordiamo che l'obiettivo del caso studio pilota è l'implementazione della comunicazione interoperabile, basata su specifiche SCPS, tra la SCP-Casaccia e la inter-SCP, che avviene tramite l'invio di un UrbanDataset da SCP-Casaccia a iSCP. Per implementare tale comunicazione sarà necessario sviluppare alcuni moduli software SERVICE (interni alla SCP), che verranno configurati per produrre o recuperare UrbanDataset, e per questo hanno una funzione analoga a quella delle SOLUTION (che invece sono esterne alla SCP).
Si veda, per una spiegazione generale dell'architettura SCP, la specifica "SCPS Functional 2.0" [3].

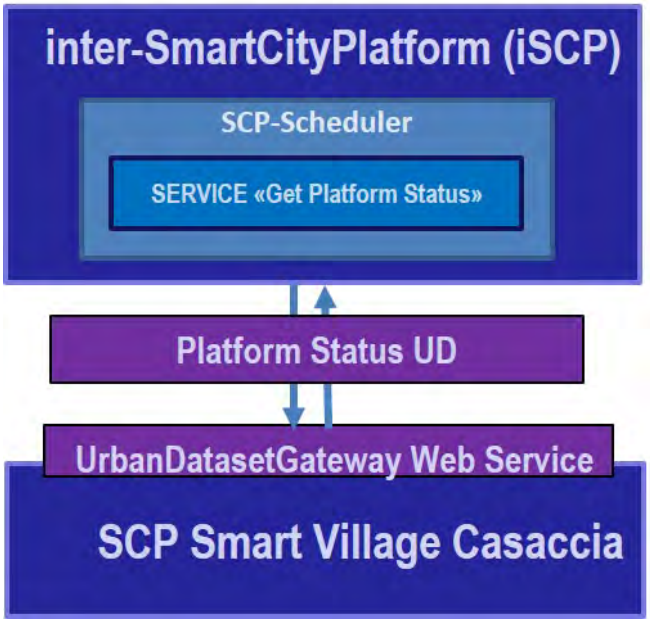

Figura 5 - SCP-Scheduler per abilitare comunicazione tra iSCP e SCP

Il modulo "SCP Bridge" inizialmente pensato come modulo apposito per permettere la comunicazione tra SCP, può essere quindi implementato con lo SCP-Scheduler che ha al suo interno un SERVICE apposito, che chiameremo "Get Platform Status".
Il SERVICE "Get Platform Status":
- è configurato per trattare l'UD "Platform Status" (che, come abbiamo visto nei par. 2.1 e 2.2, è l'UrbanDataset definito per il caso studio pilota in oggetto);
- agisce da client web service, richiamando il metodo *UrbanDatasetGateway.lastRequest*, che permette il recupero dell'ultimo UD generato del tipo specificato (si veda par.3.3.1);

- salva l'UD recuperato nella banca dati degli UrbanDataset, utilizzando una chiamata all'*UrbanDatasetGateway.push* locale, che salva l'UD e lo cataloga opportunamente.
Si noti che la inter-SCP recupererà l'UrbanDataset dalla SCP-Casaccia ma, allo stesso modo, potrà recuperare tale UD da tutte le piattaforme urbane desiderate.
Lo SCP-Scheduler, quindi, può eseguire più SERVICE registrati con una frequenza pre-configurata, anche più di una volta, per ogni connessione configurata su scala nazionale tra iSCP e SCP.
Su scala urbana, invece, lo SCP-Scheduler potrà essere utilizzato per configurare connessione "attive" tra SCP e Solution dove, diversamente da ciò che accade ora, le SCP potranno recuperare gli UD dalle Solution anziché limitarsi a riceverli.

Si noti, inoltre, che questo stesso meccanismo di recupero dell'UD tramite un service di tipo "bridge" è valido anche per effettuare l'integrazione SCP-PELL (di cui si parlerà nel cap.5).

Questa evoluzione architetturale, da SCP-Bridge a SCP-Scheduler, permette non solo di implementare la comunicazione interoperabile tra diverse SCP (e, quindi, tra SCP e iSCP) ma anche una serie di altre funzionalità necessarie che non erano state inizialmente pensate e che potranno essere implementate tramite SERVICE appositi, per esempio:

- l'esportazione periodica di UrbanDataset in una SCP (si veda par.2.1);
- azioni di Data Fusion o Data Integration (si veda il prossimo par.2.4);

che da un lato permetterano l'implementazione del caso studio pilota, dall'altro andranno a potenziare il prototipo SCP, sia su scala nazionale che su scala urbana, aumentandone le potenzialità e applicazioni di utilizzo permettendo una organizzazione modulare dei task e un'azione di monitoraggio puntuale in tutte le fasi che compongono l'intero flusso dati. Lo progettazione e sviluppo dello SCP-Scheduler verranno maggiormente dettagliati nell'ultima annualità del triennio, il 2021, nella LA 1.21.

A questo punto, l'UD "Platform Status" è stato recuperato dalla piattaforma inter-SCP, sia quello relativo alla singola SCP-Casaccia che quelli delle altre eventuali piattaforme urbane connesse alla inter-SCP.

2.4 Data Fusion di UD

L'UrbanDataset "Platform Status" generato dalla SCP-Casaccia è stato recuperato, nella precedente fase, dalla inter-SCP. A questo punto, prima di arrivare all'ultima fase di visualizzazione dati nella dashboard della inter-SCP, è necessario **preparare i dati**: ricordiamo infatti che stiamo definendo una metodologia che possa essere replicata per ogni SCP connessa alla inter-SCP e, quindi, dobbiamo immaginare di aver recuperato una serie di UrbanDataset "PlatformStatus", uno per ogni SCP connessa alla inter-SCP.
Ciò che è necessario fare è **compiere un'azione di DataFusion**.
Per **DataFusion** intendiamo "il processo che, applicato a dati tra loro armonizzati, consente la creazione di nuova conoscenza grazie alla loro combinazione" (definizione presa dal "*Glossario SCPS 2.0*" [12]).

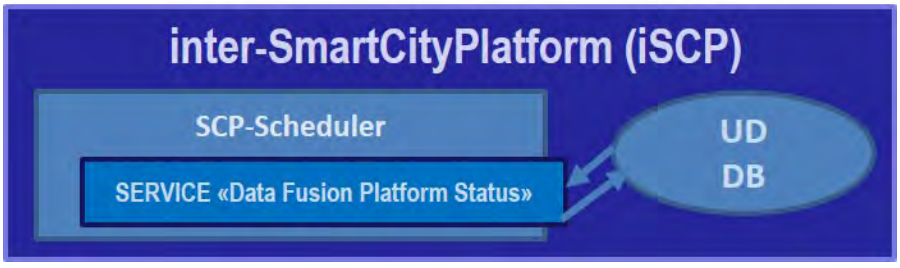

Figura 6 - DataFusion nella iSCP

Per il nostro caso studio pilota, ciò che si deve fare è effettuare una DataFusion dei diversi UD "Platform Status", recuperati idealmente da diverse piattaforme cittadine, in un unico UrbanDataset "Platform Status" che diviene input per la iSCP-Dashboard (dashboard di visualizzazione dati predisposta per la iSCP).

Tale azione di DataFusion sarà a carico di un SERVICE che chiameremo "Data Fusion Platform Status", configurato ed eseguito periodicamente per mezzo dello SCP-Scheduler (che per la terza volta, dopo l'esportazione e la trasmissione dell'UD, acquista un ruolo importante per eseguire azioni periodiche).
Il SERVICE "Data Fusion Platform Status" prende il singolo set di informazioni contenuto nel singolo UD "Platform Status" recuperato da una singola SCP e lo copia in un apposito "Platform Status" che chiameremo informalmente "multiplo", che contiene le descrizioni di tutte le SCP, aggiornando opportunamente le informazioni di contesto.
Si noti che il sistema di catalogazione della SCP già permette di distinguere i diversi "Platform Status" tramite il sistema di identificazione basato su *resource_id*, che permette di identificare in maniera univoca uno specifico UD, prodotto da una specifica Solution/Service in una particolare SCP (secondo le specifiche "*SCPS Collaboration 2.0*" [6]). Si veda a tal proposito la configurazione descritta nel par.2.6.
Una volta che l'azione di DataFusion è conclusa, l'UD "Platform Status" risultante viene salvato nella banca dati degli UrbanDataset tramite chiamata all'*UrbanDatasetGateway.push* (in accordo con le specifiche "*SCPS Communication 2.0*" [7]).

A questo punto, avendo configurato l'accesso per la iSCP-Dashboard, l'UD "Platform Status" descrivente lo stato di N SCP connesse alla inter-SCP è automaticamente pronto per il relativo recupero.

2.5 Visualizzazione Finale UD

L'UrbanDataset "Platform Status" può essere recuperato dalla iSCP-Dashboard, ed essere utilizzato per la creazione di un report web (composto a sua volta da un insieme di tabelle e grafici).

Prima di descrivere tale fase, vogliamo fornire una descrizione generale dei due componenti che permettono alla SCP l'interazione utente tramite interfacce grafiche: SCP-GUI e SCP-DASH.
Il prototipo SCP sviluppato nel precedente triennio [8] era dotato di un'interfaccia denominata SCP-GUI.
La **SCP-GUI** è l'interfaccia utente (*Graphical User Interface*) che permette la gestione in una SCP delle connessioni con le diverse Solution verticali (gestione dell'Interoperability Layer secondo l'Architettura di Riferimento descritta nelle nuove *SCPS Functional 2.0* [3]). Questa interfaccia web permette inoltre di monitorare in tempo reale gli UrbanDataset che la SCP riceve e permette di poterli leggere o effettuarne il download. Un UrbanDataset, però, preso singolarmente fornisce un'informazione specifica che è utile per effettuare delle verifiche a basso livello ed esaminare i dati specifici raccolti ma, per la maggior parte degli utilizzatori finali, ha un'utilità minima. Si pensi **per esempio** al consumo energetico quotidiano di un edificio "A": questa informazione, ricevuta tramite un UrbanDataset, può essere poco efficace se presa singolarmente ma, se definiamo un grafico con una curva temporale che ci mostra i consumi quotidiani del palazzo "A" nell'ultimo mese o nell'ultimo anno, magari comparandolo con una seconda curva relativa ad un palazzo "B", ecco che l'insieme dei dati, aggregati e mostrati in maniera opportuna, hanno una grande efficacia per chi deve effettuare la governance di un distretto o di una città.

Nella precedente annualità 2019, nella Linea di Attività 1.19 [14], è stato eseguito uno studio dello stato dell'arte delle interfacce di visualizzazione dati; ciò è risultato molto utile per la **progettazione e sviluppo del nuovo componente SCP-DASH** (che verrà descritto nel cap.4).

La **SCP-DASH** è un'interfaccia che permette la visualizzazione dei dati tramite report composti da insiemi di tabelle, chart e grafici di diverso tipo (è una dashboard per SCP, e come tale è collocata nell'Application Layer secondo l'Architettura di Riferimento descritta nella specifica *SCPS Functional 2.0* [3]).
Nel Caso Studio Pilota è stato deciso di utilizzare il componente SCP-DASH come punto di arrivo finale del caso studio pilota: esso permette di definire una dashboard per la inter-SCP (iSCP-Dashboard) per la visualizzazione dei dati raccolti dalle piattaforme urbane tramite una prima versione di cruscotto di monitoraggio. Vediamo a livello architetturale come si colloca la SCP-DASH nel flusso dati delineato.

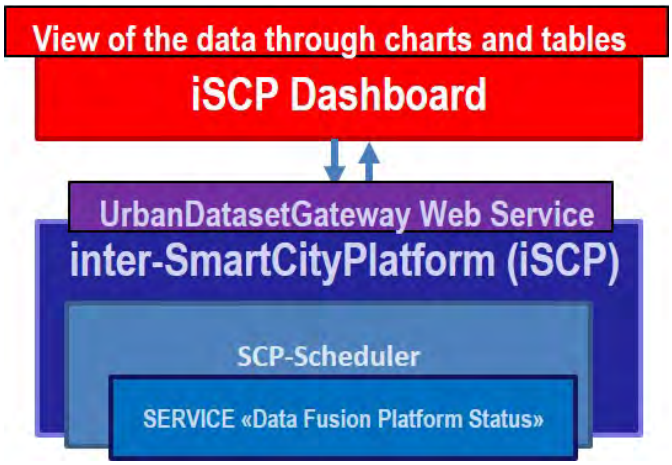

Figura 7 - Architettura concettuale iSCP-Dashboard

Per trasmettere i dati dalla iSCP alla sua dashboard (iSCP-Dashboard), si utilizzerà lo stesso canale interoperabile basato su specifiche SCPS (e quindi con una rappresentazione dei dati per mezzo di

- UrbanDataset (secondo specifica SCPS Information 2.0 [5]) e
- trasporto dei dati tramite web service *UrbanDatasetGateway* (secondo specifica *SCPS Communication 2.0* [7]).

Il componente SCP-Dash, infatti, è stato progettato per ricevere in input i dati rappresentati nel formato UrbanDataset (si veda il cap.4).

Una volta che gli UrbanDataset di input per la iSCP-Dashboard sono stati recuperati, essi vengono istantaneamente utilizzati per la valorizzazione dei report configurati. Nella seguente figura viene presentata la prima ipotesi schematica di come la dashboard potrebbe essere configurata per mostrare un primo report, aggiornato periodicamente, relativo alle piattaforme urbane connesse, al loro stato di funzionamento e a una serie di altre informazioni, anche su mappa, utili al monitoraggio su scala nazionale.

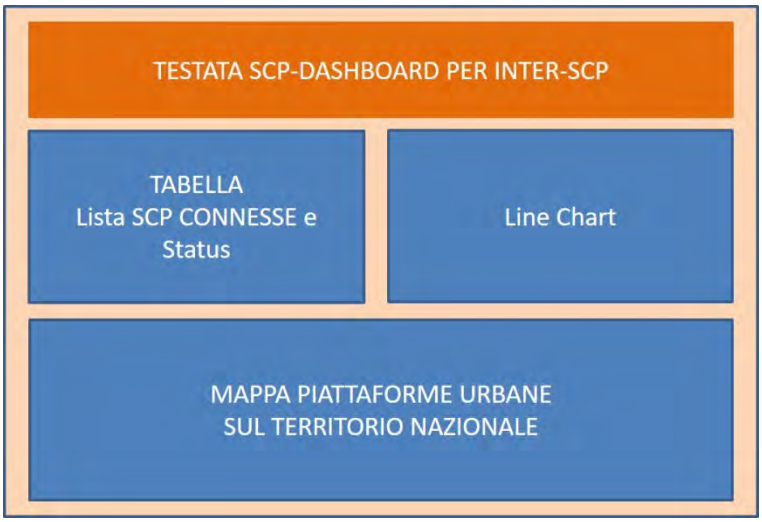

Figura 8 - Prima ipotesi report per la SCP-DASH della iSCP

Si rimanda al cap.4 per una overview dei grafici supportati dal componente SCP-DASH sviluppato.
Il risultato finale, ovvero l'implementazione della iSCP-Dashboard, completa di grafici aggiornati con dati recuperati dalla SCP-Casaccia, sarà mostrato nella LA 1.21 (annualità 2021) che descriverà l'implementazione e la sperimentazione finale del caso studio pilota.

2.6 Configurazione Collaborazioni

Abbiamo presentato, nei precedenti paragrafi, le diverse fasi principali del caso studio pilota:

1. definizione UD "Platform Status" nell'Ontologia (par.2.1);
2. esportazione UD "Platform Status" nella SCP-Casaccia (par.2.2);
3. trasmissione UD "Platform Status" da SCP-Casaccia a inter-SCP tramite SCP-Scheduler (par.2.3);
4. data fusion, nella inter-SCP, degli N UD "Platform Status" delle N SCP connesse in 1 UD (par.2.4);
5. visualizzazione dei dati tramite iSCP-Dashboard (par.2.5);

e abbiamo visto che ognuna di queste tre fasi

- richiede l'utilizzo dello SCP-Scheduler che esegue appositi SERVICE per la generazione o trasformazioni dei dati;
- utilizza il web service *UrbanDatasetGateway* (*SCPS Communication 2.0*) per la comunicazione.

Si noti come il caso studio pilota non solo metta in evidenza il funzionamento della comunicazione tra una piattaforma urbana e la inter-SCP ma utilizzi i nuovi componenti sviluppati andando a esplicitare tutte le nuove funzionalità e, in questo modo, permetta di descrivere con un singolo flusso dati, seppur complesso, il risultato finale a cui tendere per dimostrare l'efficacia di tutte le nuove funzionalità sviluppate.

Segue descrizione delle **configurazioni** necessarie, **già effettuate nel prototipo inter-SCP 0.1**, per le diverse fasi del caso studio per preparare i diversi componenti a leggere/scrivere gli UrbanDataset locali/remoti, per mezzo delle due piattaforme SCP e iSCP coinvolte nel caso studio.
Le configurazioni delle collaborazioni seguono la specifica SCPS Collaboration [6].

2.6.1 Configurazione SCP-Casaccia

Lato SCP-Casaccia, abbiamo visto (par.2.2) che il modulo SERVICE "Export Platform Status" si occuperà di esportare l'UD "Platform Status" e poi lo salverà localmente rendendolo disponibile alla inter-SCP. Sono necessarie, dunque, le seguenti configurazioni:

1. produzione dell'UD, che avverrà da parte del SERVICE;
2. accesso all'UD, che avverrà da parte della inter-SCP.

Le configurazioni del SERVICE nello SCP-Scheduler saranno descritte nella LA 1.21 (annualità 2021).

2.6.1.1 Produzione UD da "Export Platform Status"

Per configurare il SERVICE "*Export Platform Status*" come produttore dell'UrbanDataset *"Platform Status"*, prima di tutto occorre registrare un account utente e dare il nome al SERVICE in oggetto, per esempio:

1. account email: "scp.*services@enea.it*",
2. service name: "*Export Platform Status*",
3. service id (generato automaticamente): "*ExportPlatformStatus-102*",

ricordando che a un singolo account potranno essere associati più SERVICE.

Una volta registrato il SERVICE "*ExportPlatformStatus-102*" nella SCP-Casaccia, si potrà definire la PRODUZIONE dell'UD, ovvero creare l'associazione tra tale SERVICE e l'UD "*PlatformStatus-1.0*" .

Questa nuova PRODUZIONE (che ricordiamo essere locale alla SCP-Casaccia) prenderà come *resource_id*:

"*SCP-1_ExportPlatformStatus-102_PlatformStatus-1.0_20210101120000*"
(generato automaticamente seguendo la convenzione definita nella specifica *SCPS Collaboration 2.0* [6]).

Ciò implica che il modulo software SERVICE che esporta l'UD dovrà inviarlo all'*UrbanDatasetGateway* web service locale, richiamando il metodo di *push* (secondo la specifica *SCPS Communication 2.0* [7]), consentendone così il monitoraggio.

Configuriamo, inoltre, riguardo il suddetto SERVICE:

- di agire una volta al giorno in modalità "Append", ovvero salvando tutti gli UD, senza sovrascrivere i precedenti (per consentire eventuali analisi interne sul proprio stato nel tempo);

- di essere NON OPENDATA, in quanto può contenere informazioni sullo stato della SCP considerate sensibili.

A questo punto l'UD viene esportato dal SERVICE "*Export Platform Status*" e viene anche salvato nella banca dati locale degli UrbanDataset della SCP-Casaccia.

2.6.1.2 Accesso UD per inter-SCP

L'ipotesi fatta nel punto precedente è che l'UD esportato NON sia OPENDATA, in caso contrario l'accesso sarebbe libero per chiunque, ma si preferisce configurare un accesso esclusivo per la inter-SCP (par. 2.3).

E' necessario dunque registrare la inter-SCP sulla SCP-Casaccia con un account utente e un "Solution Name", in quanto la inter-SCP sarà vista dalla SCP-Casaccia come una normale Solution verticale a cui fornire accesso.

Definiremo, dunque:

4. account email: "*inter.scp@enea.it*",
5. solution name: "*inter Smart City Platform*",
6. solution id: "*interSmartCityPlatform-999*",
7. e il relativo ACCESSO all'UD, ovvero creare l'associazione

   tra la SOLUTION "*interSmartCityPlatform-999*" e il resource_id

   "SCP-1_ExportPlatformStatus-102_PlatformStatus-1.0_20210101120000".

Si noti che le configurazioni di produzione e accesso definite per la SCP-Casaccia in questo par.2.6.1 saranno identiche in tutte le SCP fornite da ENEA e quindi saranno configurate nella SCP-Template (descritta in Appendice A) utilizzata come prototipo di origine di tutte le SCP, in modo tale che **tutte le SCP ereditino tale configurazione** senza la necessità di doverla ridefinire ogni volta.

2.6.2 Configurazione inter-SCP

Lato inter-SCP, abbiamo visto (par.2.3) che il modulo SERVICE "Get Platform Status" si occuperà di recuperare l'UD "Plafrom Status", lo salverà localmente alla inter-SCP per renderlo disponibile in input al modulo SERVICE "Data Fusion Platform Status" che, a sua volta, genererà un UD in output che salverà localmente nella inter-SCP (par.2.4) e che verrà letto e utilizzato dalla iSCP-Dashboard (par.2.5).

Sono necessarie, dunque, le seguenti configurazioni:

1. produzione dell'UD, che avverrà da parte del SERVICE che ha recuperato l'UD dalla SCP-Casaccia e lo deve salvare localmente nella inter-SCP;
2. accesso all'UD di input, che avverrà da parte del SERVICE relativo alla DataFusion;
3. produzione dell'UD in output, che avverrà da parte del SERVICE relativo alla DataFusion;
4. accesso all'UD output della DataFusion, che avverrà da parte della iSCP-Dashboard.

Le configurazioni dei SERVICE nello SCP-Scheduler saranno descritte nella LA 1.21 (annualità 2021).

2.6.2.1 Produzione UD da SCP-Casaccia

Abbiamo visto che, lato inter-SCP, il modulo SERVICE "*Get Platform Status*" recupererà l'UD *"Platform Status"* dalla SCP-Casaccia. Una volta recuperato l'UD, il SERVICE "*Get Platform Status*" scriverà l'UD nel sistema locale della inter-SCP, fungendo da "bridge", come se fosse la SCP-Casaccia stessa a compiere l'azione.

In questo modo, la piattaforma inter-SCP monitorerà le produzioni degli UD come se fossero le piattaforme urbane ad aver inviato gli UD.

Per configurare la SCP-Casaccia come produttore dell'UrbanDataset *"Platform Status"*, occorre registrarla come Solution presso la inter-SCP, con:

1. account email,
2. solution name: "*SCP Casaccia*",
3. service id (generato automaticamente): "*SCPCasaccia-1*".

Una volta registrata la SOLUTION SCP-Casaccia, si potrà definire la PRODUZIONE dell'UD, ovvero creare l'associazione con l'UD "*PlatformStatus-1.0*" .

Questa nuova PRODUZIONE (che ricordiamo essere locale alla inter-SCP) sarà identificata con *resource_id*:

"*SCP-999_SCPCasaccia-1_PlatformStatus-1.0_20210101120000*"
(generato automaticamente seguendo la convenzione definita nella specifica *SCPS Collaboration 2.0* [6]).

La configurazione suddetta, che si può effettuare con la SCP-GUI, è evidente nella seguente figura.

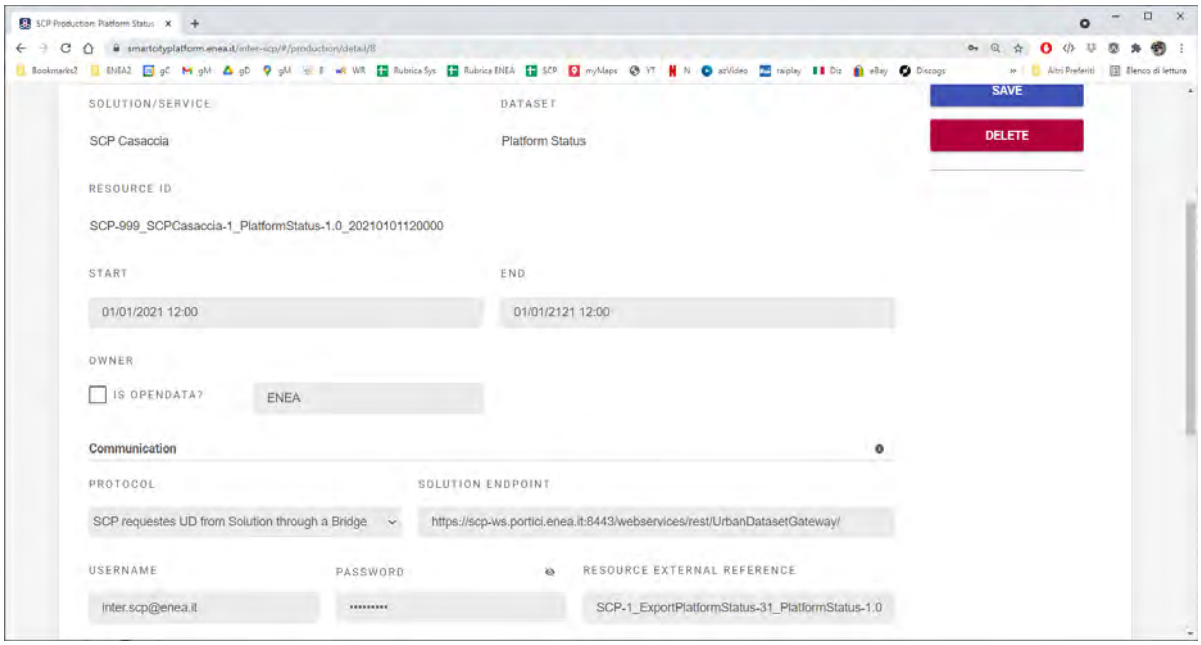

Figura 9 - Configurazione comunicazione con Bridge

Si noti che il protocollo di trasmissione sfrutta la nuova opzione "SCP requests UD from Solution through a Bridge" che è ciò che farà esattamente il SERVICE "*Get Platform Status*" recuperando l'UD dalla solution SCP-Casaccia, tramite l'endpoint specificato, le credenziali di accesso predisposte (par.2.6.1.2) e il resource_id che identifica la risorsa esportata dal modulo "Export Platform Status" nella SCP-Casaccia (par.2.2).

Il SERVICE "*Get Platform Status*" che recupera l'UD dovrà poi inviarlo all'*UrbanDatasetGateway* locale, richiamando il metodo di *push* (secondo la specifica *SCPS Communication 2.0* [7]).

Configuriamo, inoltre, riguardo la suddetta produzione:

- di agire una volta al giorno in modalità "Append", ovvero salvando tutti gli UD, senza sovrascrivere i precedenti (per consentire eventuali analisi sullo stato nel tempo);

- di essere NON OPENDATA, in quanto può contenere informazioni sullo stato della SCP considerate sensibili.

A questo punto l'UD viene recuperato dal SERVICE "*Get Platform Status*" e viene anche salvato nella banca dati locale degli UrbanDataset della inter-SCP.

Si noti che la configurazione suddetta dovrà essere ripetuta per ogni nuova piattaforma urbana monitorata e quindi connessa alla inter-SCP per permettere alla inter-SCP di ottenere il relativo UD "Platform Status".

2.6.2.2 Accesso UD per input Data Fusion

L'ipotesi fatta nel punto precedente è che l'UD "*Platform Status*" recuperato dalla SCP-Casaccia nella inter-SCP NON sia OPENDATA, in caso contrario l'accesso sarebbe libero per chiunque ed è sconsigliato; questo UD non deve essere né letto e né utilizzato dagli utenti, ma servirà come input del modulo di DataFusion (par. 2.4) a cui, quindi, è necessario configurare l'accesso.

E' necessario dunque registrare il modulo di DataFusion, come SERVICE, sulla inter-SCP con un account utente specifico e un adatto "Solution Name" (poiché le SCP trattano i SERVICE come fossero Solution).

Definiremo, dunque:

1. account email: "*scp.services@enea.it*",
2. solution name: "*Data Fusion Platform Status*",
3. solution id: "*DataFusionPlatformStatus-103*",
4. e il relativo ACCESSO all'UD, ovvero creare l'associazione

   tra il SERVICE "*DataFusionPlatformStatus-103*" e il resource_id

   "*SCP-999_SCPCasaccia-1_PlatformStatus-1.0_20210101120000*".

Si noti che soltanto il punto 4 della configurazione suddetta dovrà essere ripetuta per ogni nuova piattaforma urbana connessa alla inter-SCP per permettere al modulo di DataFusion di ottenere tutti gli UD "Platform Status" da ogni piattaforma urbana monitorata.

2.6.2.3 Produzione UD per output Data Fusion

Per configurare il SERVICE "*Data Fusion Platform Status"* come produttore dell'UrbanDataset *"Platform Status"* (non il singolo UD prodotto da ogni SCP ma quello "multiplo" che contiene gli stati di tutte le piattaforme, come spiegato nel par.2.4), utilizzeremo lo stesso SERVICE definito nel punto precedente che è esattamente la stessa entità che ha acceduto all'input e ora vuole scrivere l'output tramite UD:

1. account email: "*scp.services@enea.it*",
2. solution name: "*Data Fusion Platform Status*",
3. solution id: "*DataFusionPlatformStatus-103*",

ricordando che a un singolo account potranno essere associati più SERVICE.

Si potrà così immediatamente definire la PRODUZIONE dell'UD, ovvero creare l'associazione tra tale SERVICE e l'UD "*PlatformStatus-1.0*" .

Questa nuova PRODUZIONE (che ricordiamo essere locale alla inter-SCP) prenderà come *resource_id*:

"*SCP-999_DataFusionPlatformStatus-103_PlatformStatus-1.0_20210531000000*"
(generato automaticamente seguendo la convenzione definita nella specifica *SCPS Collaboration 2.0* [6]).

Ciò implica che il modulo software SERVICE che genera l'UD come output della DataFusion dovrà inviarlo all'*UrbanDatasetGateway* web service locale, richiamando il metodo di *push* (secondo la specifica *SCPS Communication 2.0* [7]), consentendone così il monitoraggio.

Configuriamo, inoltre, riguardo il suddetto SERVICE:

- di agire una volta al giorno in modalità "Append", ovvero salvando tutti gli UD, senza sovrascrivere i precedenti (per consentire eventuali analisi interne sul proprio stato nel tempo);

- di essere NON OPENDATA, in quanto può contenere informazioni sullo stato della SCP considerate sensibili.

A questo punto l'UD viene generato dal SERVICE "*Data Fusion Platform Status*" e viene anche salvato nella banca dati locale degli UrbanDataset della inter-SCP.

Si noti che la suddetta configurazione non dovrà essere ripetuta ma sarà valida per ogni nuova piattaforma urbana connessa alla inter-SCP.

2.6.2.4 Accesso UD per iSCP-Dashboard

L'ipotesi fatta nel punto precedente è che l'UD "*Platform Status*" output della DataFusion nella inter-SCP NON sia OPENDATA, in caso contrario l'accesso sarebbe libero per chiunque ed è sconsigliato; questo UD servirà come input per la iSCP-Dashboard (par.2.5) a cui, quindi, è necessario configurare l'accesso.

E' necessario dunque registrare la iSCP-Dashboard, come SERVICE, sulla inter-SCP con un account utente specifico e un adatto "Solution Name" (poiché le SCP trattano i SERVICE come fossero Solution).

Definiremo, dunque:

1. account email: "*scp.services@enea.it*",
2. solution name: "*iSCP Dashboard*",
3. solution id: "*iSCPDashboard-104*",
4. e il relativo ACCESSO all'UD, ovvero creare l'associazione

   tra il SERVICE "*iSCPDashboard-104*" e il resource_id

   "*SCP-999_DataFusionPlatformStatus-103_PlatformStatus-1.0_20210531000000*".

Si noti che la suddetta configurazione non dovrà essere ripetuta ma sarà valida per ogni nuova piattaforma urbana connessa alla inter-SCP poiché la iSCP-Dashboard accederà al file di output della DataFusion che conterrà ogni SCP connessa e monitorata dalla inter-SCP.

2.7 Caso Studio Urbano

Sebbene non pianificato tra gli obiettivi della LA 1.20 è stato definito un secondo caso studio, denominato "*Da SCP-Casaccia a SCP-Dashboard",* che permette di definire il flusso dati nel caso di comunicazione su scala cittadina, utilizzando i nuovi componenti descritti in precedenza: "SCP-Scheduler" e "SCP-Dashboard".
La definizione di questo secondo caso studio ha permesso di effettuare una progettazione dei componenti che ne permettesse il riuso su una casistica più ampia e su diverse scale (nazionale e cittadina).
Questo caso studio è risultato indispensabile per la progettazione del componente SCP-DASH (cap.4).

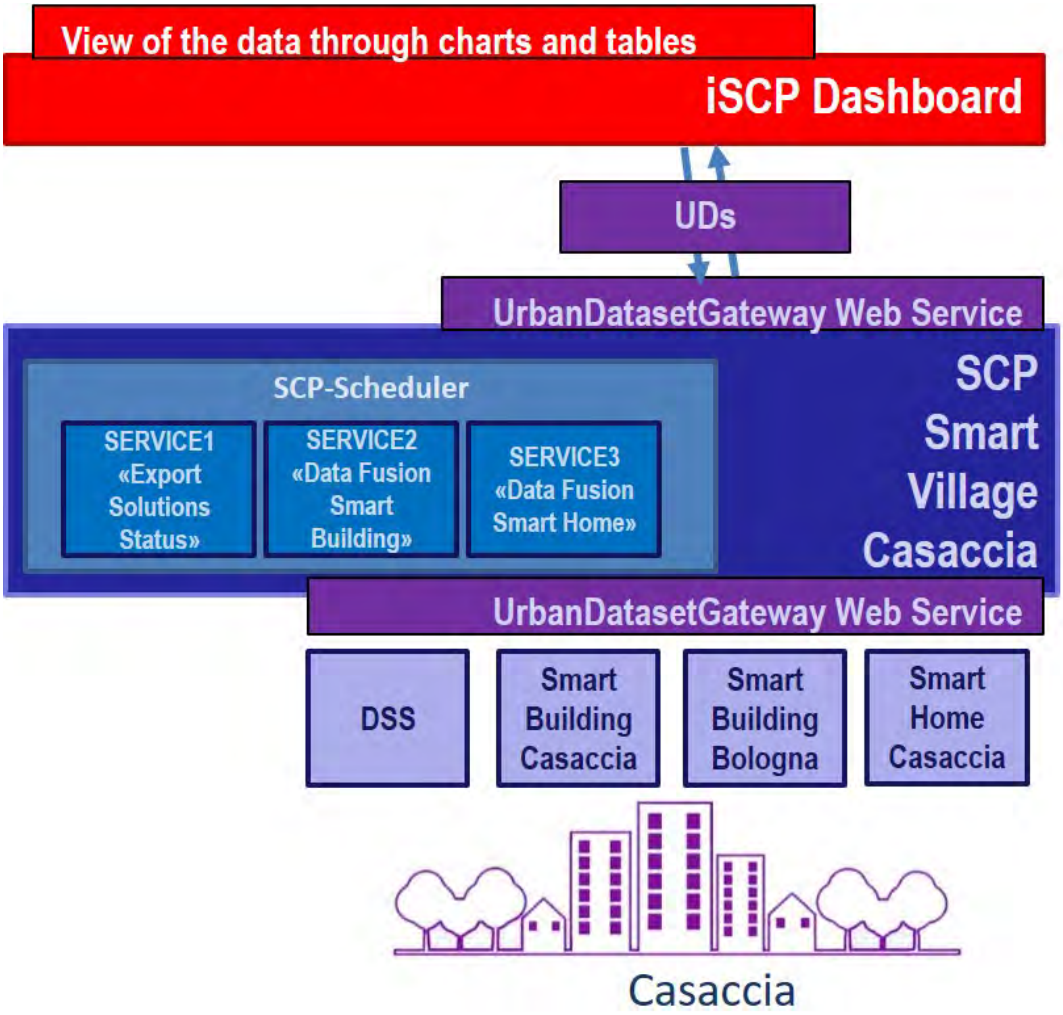

Figura 10 - Caso Studio "Urbano"

In questo secondo caso studio, l'accento è posto sulla preparazione dei dati e successiva visualizzazione degli stessi tramite dashboard definita per il prototipo "Smart City Platform Smart Village Casaccia" (SCP-Casaccia).

L'obiettivo del caso studio "urbano", infatti, è quello di utilizzare gli UrbanDataset ricevuti dalla SCP-Casaccia per una visualizzazione dei dati nella SCP-Dashboard tramite grafici e tabelle; questa preparazione dei dati consisterà in azioni di esportazione e di data fusion delegate ad appositi SERVICE (mostrati in figura).

Il caso studio in oggetto, quindi, è diverso dal caso studio pilota in quanto si focalizza sui SERVICE che preparano i dati per la SCP-Dashboard anziché descrivere tutto il flusso verticale dei dati tra SCP e iSCP.

Il caso studio "Da SCP-Casaccia a SCP-Dashboard" prevede:

1. Individuazione delle Solution e degli UD di riferimento prodotti verso la SCP-Casaccia;
2. Sviluppo dei moduli SERVICE (tra cui la Data Fusion di alcuni UrbanDataset per preparare diversi input per la SCP-Dashboard);
3. Configurazione dei SERVICE in accesso (per recuperare input) e in produzione (per salvare gli output);
4. Configurazione dell'input per la SCP-Dashboard;
5. Configurazione della visualizzazione dati in output della SCP-Dashboard associata alla SCP-Casaccia.

Segue una breve descrizione di ognuno dei cinque punti suddetti.

2.7.1 Individuazione Solution e UD

Prima di tutto, in questo caso studio, dobbiamo individuare quelle che saranno le sorgenti dati (Solution) e i relativi UrbanDataset inviati alla SCP-Casaccia che dovranno essere trattati.

I seguenti sono gli UD che si intendono preparare per la visualizzazione nella SCP-Dashboard:

- "Weather Condition" (Dati Meteo)
  prodotto dalla solution "Decision Support System";
- "Building Consumption" (Consumi in ambito Edifici)
  prodotto dalle solution "Smart Building Bologna" e "Smart Building Casaccia";
- "Smart Home Production" e "Smart Home Consumption" (Produzione e Consumi in ambito Smart Home) prodotti dalla Solution "Smart Home Casaccia";
- "Platform Status" (Stato della Piattaforma dal punto di vista delle Solution)
  prodotto da un SERVICE apposito di esportazione, "Export Solutions Status".

Quest'ultimo UD, come si descriverà nel prossimo paragrafo, è quello utilizzato nel caso studio pilota per esportare lo stato della piattaforma da inviare alla inter-SCP ma può essere utilizzato in questo caso per esportare lo stato delle Solution gestite dalla SCP corrente.

Gli UD dei primi 3 punti sono attualmente già generati e inviati alla SCP-Casaccia.

L'UD "Platform Status" del punto 4 dovrà essere generato periodicamente dal SERVICE "Export Solutions Status" che andrà sviluppato. Diversamente dal SERVICE "Export Platform Status", visto nel caso studio pilota, in questo caso sarà necessario esportare tante linee quante sono le Solutions gestite nella SCP-Casaccia, ogni linea descriverà una Solution. Grazie a questo output potrà essere abilitata una view tabellare nella SCP-Dashboard (descritta nei paragrafi seguenti).

2.7.2 Sviluppo Moduli SERVICE

Allo scopo di preparare i dati ricevuti dalla SCP-Casaccia in dati utilizzabili dalla SCP-Dashboard è necessaria un'azione di trasformazione che può essere eseguita da moduli SERVICE.

Ogni modulo SERVICE è un modulo software che esegue un insieme di azioni circoscritto e potrà essere gestito dallo SCP-Scheduler (avvio, esecuzione periodica, stop).

I moduli software individuati necessari sono i seguenti:

- SERVICE1 "*Export Solutions Status*": esportazione UD «Platform Status»;
- SERVICE2 "*Data Fusion Smart Building*": fusione di due UrbanDataset "Smart Building Consumption", inviati da Solution diverse, ed esportazione in un UD «Whatever» (dedicato);

- SERVICE3 "*Data Fusion Smart Home"*: fusione di due UrbanDataset "Home Aggregated Electric Production" e "Home Aggregated Electric Consumption", ed esportazione in un UD «Whatever» (dedicato).

2.7.3 Configurazione Moduli SERVICE in SCP-Casaccia

La configurazione dei SERVICE nella SCP-Casaccia implica la definizione, per ognuno di essi, del permesso di accesso a UD (input) e/o di produzione (output).

In particolare:

- SERVICE1 "*Export Solutions Status"*:
    - Definizione 1 Produzione UD «Platform Status»;
- SERVICE2 "*Data Fusion Smart Building"*:
    - Definizione 2 Accessi a due UrbanDataset "Smart Building Consumption" creati dalle Solution "Smart Building Bologna" e "Smart Building Casaccia";
    - Definizione 1 Produzione UD «Whatever» (dedicato);
- SERVICE3 "*Data Fusion Smart Home"*:
    - Definizione 2 Accessi ai due UrbanDataset "Home Aggregated Electric Production" e "Home Aggregated Electric Consumption" creati dalla Solution "Smart Building Bologna" e "Smart Building Casaccia";
    - Definizione 1 Produzione UD «Whatever» (dedicato).

A questo punto i moduli SERVICE potranno essere registrati nello SCP-Scheduler ed essere avviati.

2.7.4 Configurazione dell'input per la SCP-Dashboard

La configurazione dei SERVICE nella SCP-Dashboard implica la definizione, per ognuno degli UD utilizzati, la definizione del permesso di accesso a UD (input).

In particolare sarà necessario definire:

- 1 Accesso a UD «Platform Status» prodotto da SERVICE1 "*Export Solutions Status"*:
- 1 Accesso a UD «Whatever» (dedicato) prodotto da SERVICE2 "*Data Fusion Smart Building"*;
- 1 Accesso a UD «Whatever» (dedicato) prodotto da SERVICE3 "*Data Fusion Smart Home"*;
- 1 Accesso a UD "Weather Condition" (Dati Meteo) prodotto dalla solution "Decision Support System" verso la SCP-Casaccia.

2.7.5 Configurazione output SCP-Dashboard (Visualizzazione Dati)

La configurazione dei dati di OUTPUT (Diagrammi) della SCP-Dashboard associata alla SCP-Casaccia implica:

5.1 la configurazione del Report (insieme di diagrammi) che si vuole ottenere nella SCP-Dashboard;

5.2 il recupero degli UD preparati come input della SCP-Dashboard;

5.3 view del Report finale nella SCP-Dashboard della iSCP.

Una prima ipotesi schematica di quella che potrà essere la SCP-Dashboard associata alla SCP-Casaccia è mostrata nella seguente figura (si veda inoltre il cap.4 relativo allo sviluppo del componente SCP-DASH per una descrizione più dettagliata dei diversi componenti che compongono il report della dashboard).

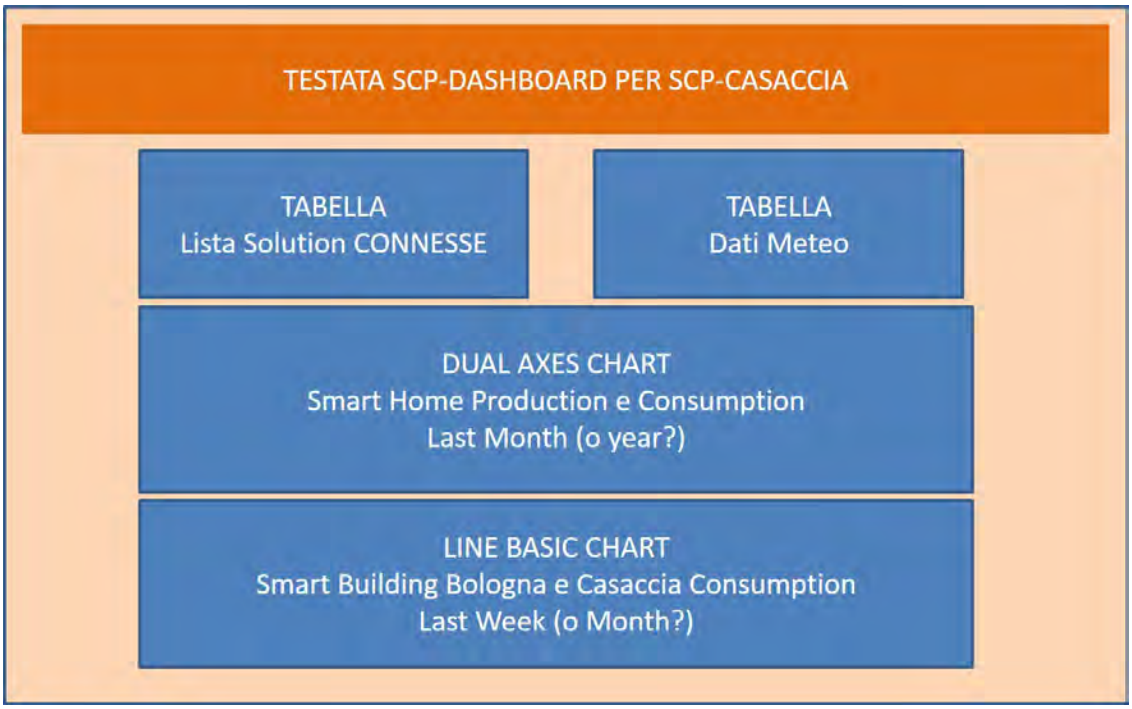

Figura 11 - View schematica relativa alla SCP-Dashboard per SCP-Casaccia

Si noti che il caso studio “Da SCP-Casaccia a SCP-Dashboard” è un caso studio supplementare a quanto pianificato e quindi, sebbene ricopra una importanza fondamentale per la progettazione e sviluppo dei nuovi componenti software, potrebbe non essere implementato interamente entro la fine del triennio.

2.8 Prototipo inter-SCP versione 0.1

Il primo prototipo di inter-SmartCityPlatform (inter-SCP o iSCP) versione 0.1 è stato installato, configurato e pubblicato su server ENEA (si veda URL [11] nei riferimenti bibliografici).

Le azioni che al momento sono state eseguite sul prototipo SCP sono le seguenti:

- azioni a livello di sistema per migliorare la replicabilità del prototipo SCP (Appendice A);
- aggiornamento delle specifiche SCPS alla versione 2.0 (cap.3);
- progettazione e sviluppo nuovo componente SCP-DASH (cap.4);
- supporto alla comunicazione interoperabile tramite SERVICE di "bridge" (par.2.3 e cap.6);
- configurazione delle collaborazioni per le produzioni e accesso di UD (par.2.6);
- configurazione della SCP-Casaccia e invio primo UD "Whatever" di test tramite web service;

queste modifiche, nella direzione di implementare il caso studio pilota descritto, hanno permesso di preparare la prima versione 0.1 del prototipo inter-SCP che è stato installato e configurato su infrastruttura ICT di ENEA e pubblicato tramite proxy (si veda [11]).

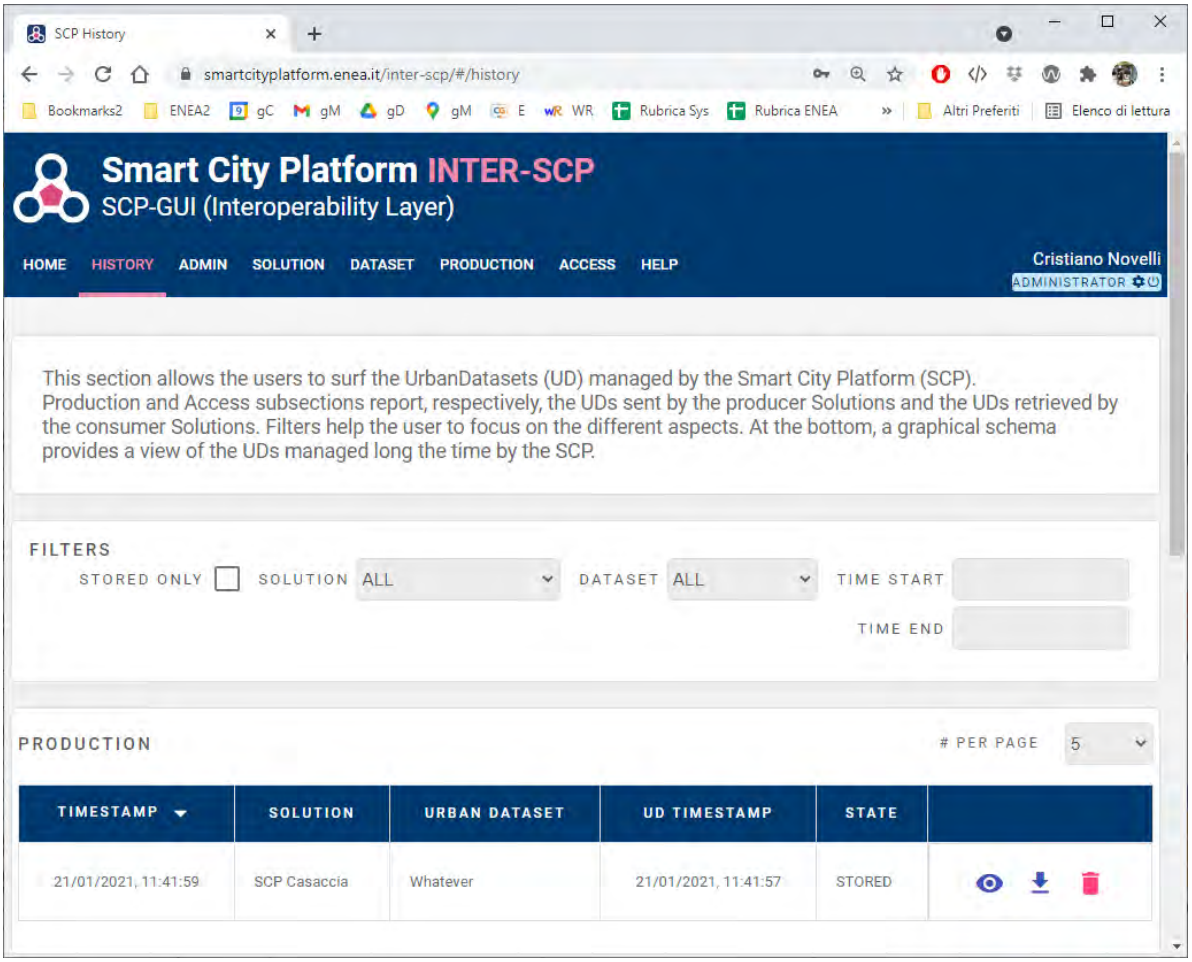

Figura 12 - inter-SCP 0.1

Come di può vedere nella precedente figura la inter-SCP è già in grado di ricevere UD dalla SCP-Casaccia ma non è ancora implementato il meccanismo che automatizza questa comunicazione (e che potrà essere fatto grazie allo SCP-Scheduler, così come descritto nel par.2.3).

E' stato inoltre definito un account temporaneo di tipo READER che permette la navigazione delle funzionalità della SCP-GUI come un utente ADMNISTRATOR ma senza permessi in scrittura:
username: guest.rds@enea.it
password: gue5t.rd5

Lo stesso account potrà essere utilizzato anche per accedere al nuovo componente SCP-DASH [15] (di cui viene fornita una descrizione nel capitolo 4 ma che al momento non ha ancora report configurati).

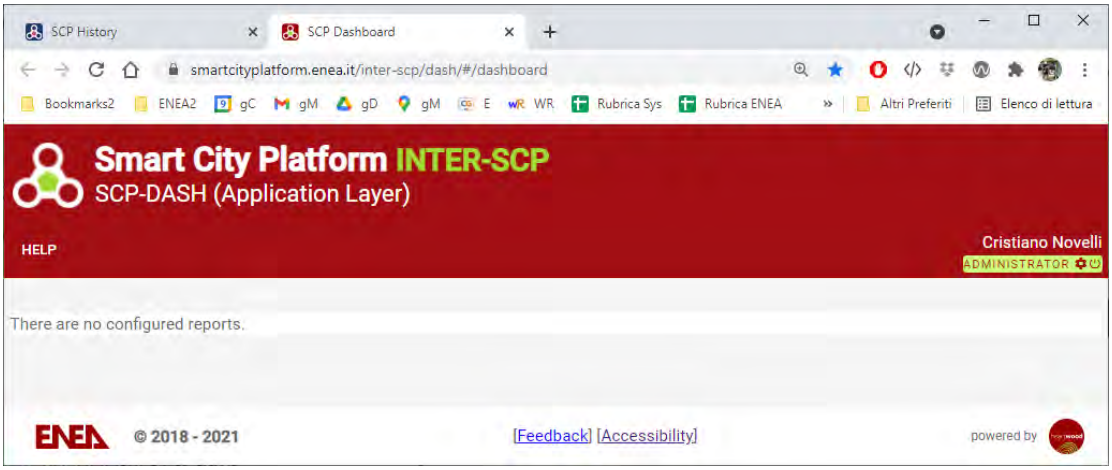

Figura 13 - iSCP Dashboard

Si noti che il componente SCP-DASH eredita tutte le funzionalità della gestione utenti sviluppata per il componente SCP-GUI e memorizzata nel Registry: ogni utente di tipo SOLUTION, quindi, vedrà solo i grafici che prendono in input UrbanDataset a cui egli ha accesso; ogni utente di tipo ADMINISTRATOR/READER potrà invece vedere tutti i report e tutti i grafici e tabelle che li compongono.

3 Evoluzione SCPS

Come abbiamo visto nella descrizione del caso studio pilota, si sono individuate nuove esigenze nel prototipo SCP/iSCP che troveranno risposta nelle funzionalità dei nuovi componenti.
Questo nuovo sviluppo di funzionalità è portato avanti tenendo ben presente che il prototipo SCP è basato sulle specifiche "SCPS for interoperability layer" e che quindi esse devono essere coerenti e descrivere eventuali nuove caratteristiche relative alla comunicazione interoperabile.
In particolare, in questo capitolo, descriveremo l'aggiornamento delle specifiche SCPS per supportare pienamente il caso studio su scala nazionale:

1. evoluzione architetturale;
2. evoluzione formato *UrbanDataset*, per la gestione di dati e KPI su scala nazionale;
3. evoluzione web service *UrbanDatasetGateway*, per supportare la comunicazione su scala nazionale.

3.1 Evoluzione Architetturale

La specifica "*SCPS Functional*" è la specifica introduttiva delle SCPS che, oltre a definire i concetti chiave e le principali funzionalità, presenta anche l'architettura di riferimento.

In questa nuova versione 2.0:

- è avvenuta una review completa per consolidare il documento,
- è stata inglobata la specifica SCPS Core 1.0 e il Glossario,
- è stata rivista ed aggiornata l'architettura di riferimento, esplicitando la suddivisione in due livelli: Interoperability Layer e Application Layer.

Non riportiamo in questo paragrafo l'intera specifica *SCPS Functional 2.0* e il *Glossario 2.0*, lasciando alla fine del report i relativi riferimenti bibliografici con gli URL con cui è possibile accedere alle versioni complete e pubbliche (si veda [5] e [12]); riporteremo, invece, una parte fondamentale dell'aggiornamento svolto a questa specifica e che è relativo all'architettura di riferimento.

Segue frammento ripreso dalla specifica *SCPS Functional 2.0*.

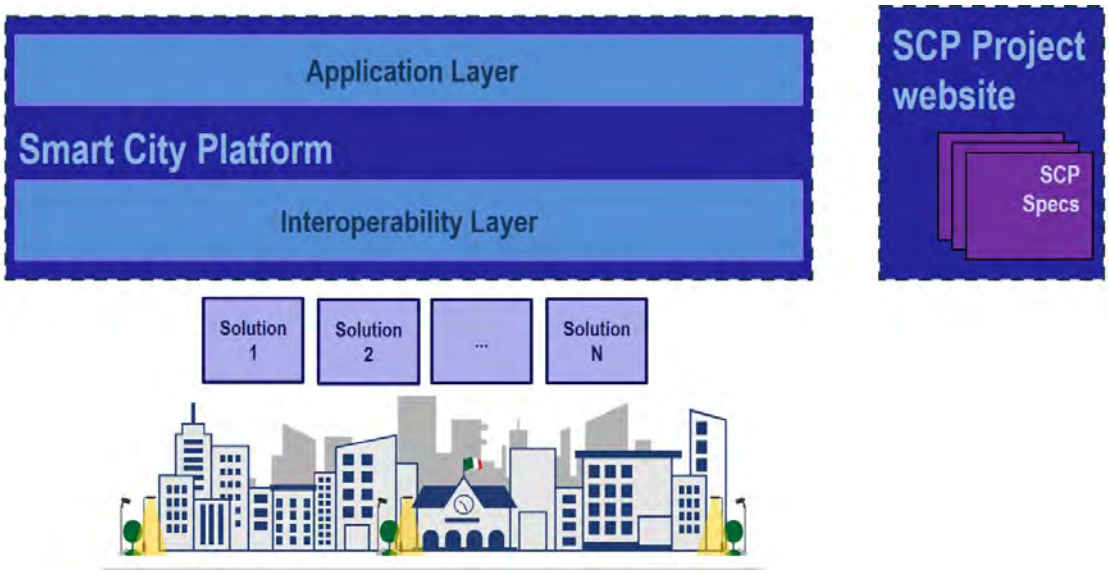

Figura 14 - Introduzione all'Architettura di Riferimento SCP

Le specifiche SCPS intendono risolvere una serie di problematiche per ottenere la comunicazione interoperabile tra i sistemi, per questo motivo vengono chiamate anche ***SCPS for Interoperability Layer****.*

In altre parole le specifiche si concentrano sugli aspetti relativi alla connessione interoperabile delle Solution Verticali con la piattaforma SCP (Interoperability Layer) e non degli aspetti relativi all'uso che si farà dei dati recuperati e armonizzati (Application Layer).

Le specifiche SCPS sono pubblicate sul sito web del progetto SCP: smartcityplatform.enea.it
che è il punto di riferimento per conoscere l'iniziativa, imparare i concetti chiave e l'approccio architetturale, aderire al framework, utilizzare le risorse e gli strumenti messi a disposizione, fruire delle specifiche SCPS (di cui questo documento è parte introduttiva) ed esplorare i progetti relativi che hanno aderito alle specifiche o che utilizzano il prototipo ENEA.

Le specifiche SCPS for Interoperability Layer hanno un approccio modulare in cinque livelli i quali ricoprono rispettivamente una parte fondamentale dell'Architettura di Riferimento rappresentata nel seguente schema.

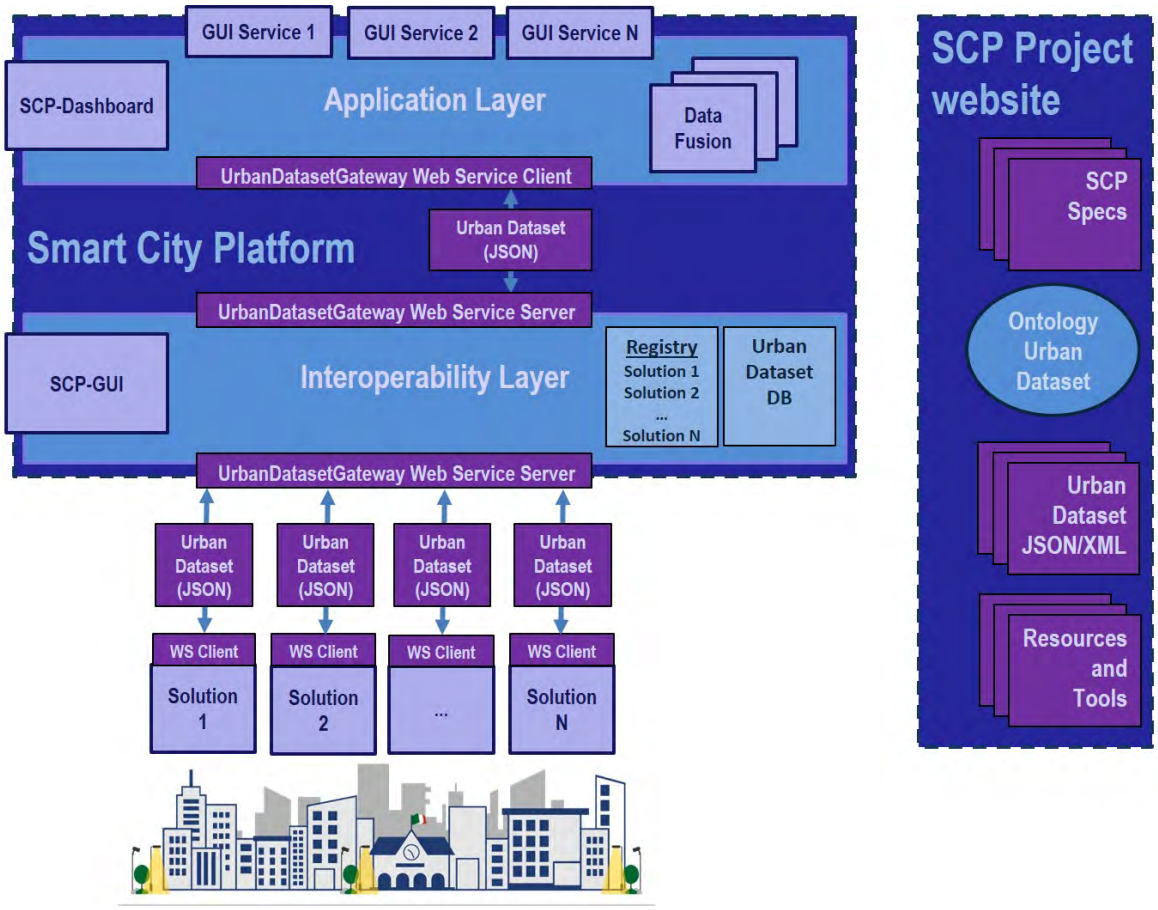

Figura 15 - Architettura di Riferimento SCP

Osservando l'Architettura di Riferimento, possiamo descrivere i diversi elementi che la compongono e che sono afferenti alle cinque specifiche SCPS:

- *SCPS Functional (specifica corrente): fornisce la descrizione dell'Architettura di Riferimento, andando a definire in primis i concetti di Solution verticale e SCP orizzontale; è molto utile cominciare lo studio delle specifiche a partire da questo modulo per comprendere la visione generale, condividere l'approccio, muoversi tra le specifiche SCPS con consapevolezza di poter risolvere, gradualmente, ogni aspetto della comunicazione interoperabile.*

- *SCPS Semantic : questa specifica affronta il problema della definizione semantica dei dati. Tale definizione viene eseguita in maniera centralizzata e viene salvata nell'Ontologia (parte alta dell'Architettura). I dati vengono strutturati come dataset, denominati UrbanDataset (UD), garantendo così un'unica definizione centralizzata, sia degli UrbanDataset che delle proprietà che li compongono, con stessi nomi, formati e unità di misura. Questa fase di definizione degli UrbanDataset è gestita da ENEA che recepisce gli input della comunità e decide se estendere o inserire i nuovi UrbanDataset necessari.*

- *SCPS Information : questa specifica si occupa della definizione sintattica dei dati; in altre parole questa specifica fornisce il modello dei dati astratto con cui rappresentare gli UrbanDataset in un formato condiviso (JSON o XML); tale formato dati verrà utilizzato nello scambio dati tra le Solution verticali e la SCP orizzontale. Questa fase di utilizzo del formato dati per rappresentare gli UrbanDataset sarà affrontata dalle Solution Verticali che dovranno esportare i dati dal proprio sistema gestionale e rappresentarli nel formato UrbanDataset prima di inviarli alla SCP; allo stesso modo, le Solution Verticali che richiederanno gli UrbanDataset dalla SCP potranno validarli, interpretarli correttamente e importarli nei propri sistemi.*

- *SCPS Collaboration: questa specifica definisce il concetto di Collaboration tra Solution e SCP, ovvero la specificazione dell'insieme di informazioni necessarie per configurare la comunicazione interoperabile tra i sistemi in gioco; queste informazioni saranno memorizzate nella bancadati della SCP, denominata Registry, che tiene traccia delle Solution registrate e dei relativi permessi, in produzione e accesso, per abilitare lo scambio di UrbanDataset. Questa fase di configurazione presso la SCP deve essere abilitata in ultima istanza dall'amministratore della SCP.*

- *SCPS Communication: questa specifica definisce il protocollo di comunicazione interoperabile, tra le Solution e la SCP, tramite la scelta dei pattern architetturali e la definizione dell'interfaccia del web service UrbanDatasetGateway, abilitante l'invio e il recupero di UrbanDataset, che vengono infine salvati nel relativo UrbanDataset database. Questa fase prevede, nell'implementazione più classica, che la SCP esponga il web service sviluppato con tecnologia RESTful e che le Solution, tramite un apposito client aderente alla specifica, richiamino i metodi esposti dal web service per l'invio o il recupero di UrbanDataset. Nello stesso modo, questa comunicazione interoperabile può essere anche utilizzata internamente tra Interoperability Layer e Application Layer della SCP: in questo modo le applicazioni di DataFusion e di visualizzazione grafica dei dati per l'utente (GUI) possono accedere ai dati armonizzati tramite lo stesso canale.*

Ogni specifica SCPS potrebbe essere adottata indipendentemente dalle altre, consentendo un'adozione parziale e graduale, ma per la piena interoperabilità tra i sistemi bisogna aderire a tutte e cinque le specifiche.

Una descrizione dettagliata dei concetti chiave e dei componenti viene fornita nel Glossario [12].

E' importante notare che la revisione architetturale suddivide in maniera netta ciò che afferisce alla comunicazione interoperabile con la SCP (*Interoperability Layer*) da ciò che è invece relativo alla successiva rielaborazione dell'informazione (*Application Layer*, p.es. i moduli di DataFusion e la visualizzazione tramite Dashboard, presentati nella descrizione del caso studio pilota nel cap.2). Ciò ha gettato le fondamenta per una evoluzione ed adattamento delle specifiche SCPS al caso nazionale in modo che non si snaturasse la loro principale funzione, relativa alla comunicazione interoperabile, permettendo così di collocare i nuovi componenti definiti (come SCP-Scheduler e iSCP-Dashboard) nell'architettura esistente, in modo ordinato.

3.2 Evoluzione UrbanDataset

Uno degli obiettivi della LA20 è quello di "valutare (ed eventualmente estendere) le specifiche per l'interoperabilità per abilitare la comunicazione tra piattaforme, su scala nazionale". Un'estensione prioritaria rispetto all'obiettivo di realizzare un Framework per la Governance dei Dati Energetici è la definizione della modalità con cui esprimere tali dati.

In quest'ottica, è stata avviata l'estensione delle specifiche SCPS Semantic e Information realizzate nello scorso triennio, per permettere la definizione e lo scambio di KPI (Key Performance Indicator) che esprimano in modo sintetico, significativo e uniforme i livelli di efficienza e sostenibilità della gestione cittadina. Questa estensione ha richiesto sia la definizione dell'approccio metodologico (nel senso che è stato definito il metodo con cui gestire questa tipologia di dati) sia formale (nel senso che è stato definito il formato condiviso con cui esprimere e scambiare tali dati).

Il punto di partenza per questa estensione è stata l'analisi di alcuni casi studio che hanno consentito di capire sia i requisiti, nella fase di analisi, che la fattibilità, nella fase di implementazione dei KPI nell'ontologia.
Il caso studio preso principalmente in esame è stato quello relativo al contesto applicativo verticale "*Smart Lighting*". Questo caso si basa su uno scenario applicativo maturo ed è stato possibile interagire strettamente con i responsabili della Solution per capire bene i requisiti e le necessità. In particolare, grazie a questa analisi è emersa anche la necessità di definire delle specifiche che abilitino una Solution o piattaforma Smart City ad inviare le informazioni sul suo stato di funzionamento necessarie per valutare l'affidabilità dei KPI inviati.

Quindi, tramite l'analisi di questi casi studio, da una parte, sono stati definiti i requisiti per l'inserimento dei KPI all'interno dell'Ontologia SCPS (e la modalità in cui gli UrbanDataset li mettono a disposizione), dall'altro sono state progettate e rese disponibili nell'applicazione web *SCPS WebLibrary* [16] le seguenti specifiche:

- UrbanDataset *Platform Status* (utilizzato nel caso studio pilota, descritto nel cap.2)
- UrbanDataset per l'invio di indicatori relativi al contesto *SmartLighting*:
  - *POD KPI PELL*
  - *City Public Lighting Indicators*

3.2.1 Modalità di gestione di KPI e indicatori urbani nell'Ontologia SCPS

Il caso studio preso in esame ha portato ad una iniziale elaborazione di due possibili approcci per la gestione dei KPI nell'Ontologia e il loro invio tramite UrbanDataset:

- il primo approccio avrebbe previsto l'inserimento dei KPI come proprietà all'interno dell'ontologia; questo avrebbe vincolato il numero di diversi KPI da poter inviare con uno stesso tipo di UD e, conseguentemente, per supportare tutti gli scenari possibili (invio di gruppi diversi di KPI), sarebbe stato necessario creare una specifica UrbanDataset per ogni sottoinsieme di KPI di cui si voleva abilitare l'invio;
- il secondo approccio, invece, avrebbe gestito i KPI tramite liste di codici da associare a generiche proprietà dell'ontologia, in modo da consentire la definizione di UrbanDataset atti ad inviare qualsiasi KPI purché presente nelle liste di codici disponibili nell'ontologia, evitando di dover predefinire tutte le possibili combinazioni di KPI a livello di ontologia.

La seguente tabella riporta un esempio e una comparazione dei due approcci.

Tabella 1 – Confronto fra i due approcci proposti

	Primo approccio	Secondo approccio
Descrizione	**Prevede l'implementazione dei KPI come proprietà dell'ontologia**: - il nome della proprietà dell'UD esplicita il KPI - ogni KPI è espresso tramite una proprietà ogni riga dell'UD contiene tutti i KPI prestabiliti per quell'UD - il numero di diversi KPI che possono essere inviati tramite messaggi che istanziano l'UD è fisso e predefinito a livello di ontologia.	**Prevede l'implementazione dei KPI tramite liste di codici da associare a proprietà dell'ontologia**: - il nome della proprietà dell'UD NON esplicita il KPI - per esprimere ogni KPI sono necessarie 2 proprietà: una per fornirne il nome (che potrà assumere solo i valori presenti nella lista di codici), una per il valore - ogni riga dell'UD contiene un solo KPI, selezionato dalla lista di codici associata a quella proprietà - il numero di diversi KPI che possono essere inviati tramite messaggi che istanziano l'UD è libero.
Esempio	"line": [{ "id": 1, "property": [{ "name": "**CityName**", "val": "Bologna" }, { "name": "**KPI_A**", "val": "25" }, { "name": "**KPI_B**", "val": "15" }, { "name": "**KPI_C**", "val": "7" }] }, { "id": 2, "property": [{ "name": "**CityName**", "val": "Roma" }, { "name": "**KPI_A**", "val": "22" }, { "name": "**KPI_B**", "val": "17" },	"line": [{ "id": 1, "property": [{ "name": "**CityName**", "val": "Bologna" }, { "name": "**KPICode**", "val": "KPI_A" }, { "name": "**KPIValue**", "val": "25" }] }, { "id": 2, "property": [{ "name": "**CityName**", "val": "Bologna" }, { "name": "**KPICode**", "val": "KPI_B" }, { "name": "**KPIValue**", "val": "15" }] }, { "id": 3,

	Primo approccio	Secondo approccio
	{ "name": "**KPI_C**", "val": "8" }] }, ...	"property": [{ "name": "**CityName**", "val": "Bologna" }, { "name": "**KPICode**", "val": "KPI_C" }, { "name": "**KPIValue**", "val": "7" }] }, { "id": 4, "property": [{ "name": "**CityName**", "val": "Roma" }, { "name": "**KPICode**", "val": "KPI_A" }, { "name": "**KPIValue**", "val": "22" }] }, ...] }, ...
PRO	**Contenimento della ridondanza delle informazione nelle istanze UD:** - per ogni KPI è sufficiente una proprietà - le ulteriori proprietà associate ai KPI, nel caso dell'esempio "CityName", occorrono una sola volta	**Scalabilità**: - gli UD supportano l'invio di qualsiasi KPI, purché compaia nella lista di codici associata alle sue proprietà - l'aggiunta di nuovi KPI è rapida e poco dispendiosa in quanto basta aggiungere i valori alla lista di codici
CONTRO	**Poca scalabilità**: - quali e quanti diversi KPI possono essere inviati tramite messaggi che istanziano l'UD è fisso e predefinito a livello di ontologia	**Elevata ridondanza delle info**: - per ogni KPI sono necessarie 2 proprietà: una per fornirne il nome, una per il valore - le ulteriori proprietà associate ai KPI, nel caso dell'esempio "CityName", vengono ripetute per ogni KPI

Volendo privilegiare un approccio scalabile, è stata adottato il secondo.
Una descrizione più dettagliata di come è stato implementato sarà fornita dei paragrafi successivi contestualmente alla presentazione degli UrbanDataset relativi all'invio di KPI e Indicatori Urbani.

3.2.2 UrbanDataset PlatformStatus

L'UrbanDataset *Platform Status* è stato ideato per abilitare l'invio periodico di informazioni da una piattaforma software ad un'altra che ne debba tenere monitorato lo stato di attività; ad esempio, da una Smart City Platform (SCP) a una inter-Smart City Platform (iSCP), ma anche da una qualsiasi Solution a una SCP.
Il punto di partenza, e poi scenario di test, per la progettazione di questo UrbanDataset, è stato il caso studio pilota "Da SCP-Casaccia a inter-SCP" (si veda cap.2); tale caso studio prevede che una SCP invii, periodicamente, informazioni sul suo stato a una inter-SCP presso la quale è registrata.
Il flusso di dati è descritto nella seguente figura.

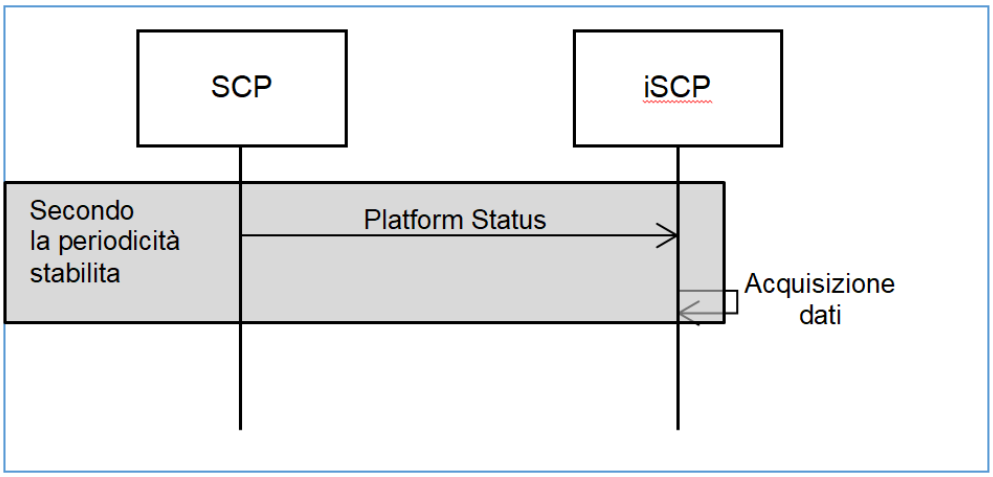

Figura 16 - Diagramma di sequenza UML del caso studio "iSCP"

Nello specifico, i **requisiti** individuati grazie all'analisi del caso studio sono i seguenti:

- la piattaforma a cui si riferiscono le informazioni deve essere identificata tramite identificatore e nome
- l'informazione relativa allo stato di funzionamento della piattaforma deve poter essere espressa sia come valore codificato che come testo libero
- l'attività della piattaforma deve essere descritta in termini di: numero di Solution e utenti gestiti, numero di UrbanDataset acceduti e prodotti
- l'UrbanDataset deve includere le informazioni necessarie per collocare la piattaforma nell'area geografica da essa gestita.

Coerentemente, l'UrbanDataset è stato definito come segue:

Tabella 2 – Intestazione dell'UrbanDataset "Platform Status"

UrbanDataset	**Platform Status**
Scopo	Informazioni dello stato di funzionamento della piattaforma all'istante di generazione dell'UrbanDataset.
Categoria / Sottocategoria	Public Safety Policy Emergency Response/Communication
	SPECIFICAZIONE DELL'URBANDATASET
UrbanDatasetId	PlatformStatus -1.0
UrbanDatasetUri	https://smartcityplatform.enea.it/SCPSWebLibrary/urbandataset?name=PlatformStatus
UrbanDatasetName	Platform Status

Tabella 3 – Contestualizzazione dell'UrbanDataset "Platform Status"

	Unità di Misura:	Formato	Descrizione / Esempio
Timestamp	Adimensionale	Date-Time aaaa-mm-ggThh:mm:ss	Tempo di generazione dell'UrbanDataset (determina anche l'istante temporale di validità delle informazioni fornite)
Timezone	Adimensionale	String	Es. UTC, UTC+01, UTC-05
Coordinates	Adimensionale	WGS84 (World Geodetic System)	Coordinate del centro geometrico dell'area gestita dalla piattaforma.
Producer	Adimensionale	String	Identificatore del Sistema che ha prodotto i dati
		PROPRIETÀ DELL'URBANDATASET	
	Unità di Misura:	**Formato**	**Descrizione / Esempio**
PlatformID	Adimensionale	String	Identificatore della piattaforma software
PlatformName	Adimensionale	String	Nome della piattaforma software
LogCode	Adimensionale	String (vincolata a lista di codici)	Codice ch-e indica lo stato dell'"oggetto" monitorato
LogDescription	Adimensionale	String	Descrizione testuale dello stato dell'oggetto monitorato
DataDescription	Adimensionale	String	Descrizione testuale dei dati inviati
TownName	Adimensionale	String	Nome esteso del Comune gestito dalla piattaforma (es. Roma, Anguillara Sabazia, ...)
ZIPCode	Adimensionale	String	Codice di Avviamento Postale(CAP) della zona gestita dalla piattaforma
ProvinceCode	Adimensionale	String	Codice Provincia a cui appartiene l'area gestita dalla piattaforma
TotalManagedUsersCount	Adimensionale	Integer	Numero degli utenti gestiti
TotalManagedVerticalsCount	Adimensionale	Integer	Numero delle applicazioni verticali gestite
TotalAccessedUDCount	Adimensionale	Integer	Numero totale di UrbanDataset acceduti
TotalProducedUDCount	Adimensionale	Integer	Numero totale di UrbanDataset prodotti

La lista di codici a cui è stata vincolata la proprietà "*LogCode*" è stata resa disponibile in formato Genericode ed è riportata in Appendice B.1.

3.2.3 UrbanDataset per KPI SmartLighting

Il caso studio Smart Lighting "PELL" (Public Energy Living Lab, piattaforma che raccoglie i consumi elettrici da tutte le municipalità aderenti per offrire un servizio di monitoraggio e benchmarking) ha messo innanzitutto in evidenza la necessità di distinguere gli indicatori comunemente denominati Key Performance Indicator (KPI), ovvero indicatori ottenuti dalla combinazione ed elaborazione di diversi dati relativi al contesto urbano, da informazioni più elementari (ad esempio il numero di punti luce dell'impianto di pubblica illuminazione di una città) che sono stati denominati Indicatori Urbani.

Ha quindi consentito l'identificazione dei KPI e degli Indicatori Urbani relativi all'illuminazione pubblica che, una piattaforma di gestione degli impianti di pubblica illuminazione, sarebbe

potenzialmente in grado di calcolare e fornire a una SCP o una generica Solution che ne faccia richiesta.
Al fine di supportare l'invio di queste informazioni, sono stati progettati e resi disponibili i due UrbanDataset di seguito descritti:

- UrbanDataset *POD KPI PELL*
- UrbanDataset *City Public Lighting Indicators*

Entrambi sono stati definiti coerentemente con l'approccio alla gestione degli UrbanDataset relativi ai KPI precedentemente illustrato.

UrbanDataset PODKPIpell

L'UrbanDataset *POD KPI PELL* è stato ideato per consentire, ad una piattaforma di gestione degli impianti di pubblica illuminazione, l'invio di Key Performance Indicator (KPI) calcolati a livello di POD; l'invio può avvenire verso una SCP o una generica Solution che ne faccia richiesta.
Il flusso di dati è descritto in Figura

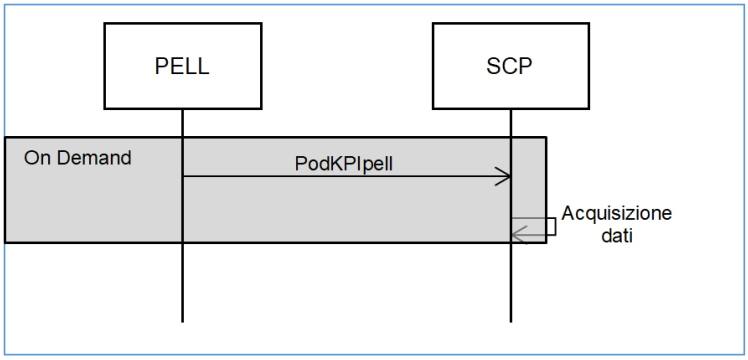

Figura 17 - Diagramma di sequenza UML del caso studio Smart Lighting "PELL" - invio KPI

Nello specifico, i seguenti requisiti individuati grazie all'analisi del caso studio sono i seguenti:

- ogni KPI inviato deve essere riferito a un POD, identificato tramite il proprio codice, e una città, identificata tramite codice ISTAT; può inoltre essere indicato il riferimento ad una zona omogenea[1]
- l'insieme dei KPI supportati dall'UrbanDataset deve essere scalabile nel tempo (deve ciò essere possibile aggiungere nuovi KPI, purché abbiano le caratteristiche indicate al punto precedente)
- in una stessa istanza dell'UrbanDataset deve essere possibile riportare informazioni relative a più POD, anche di città diverse.

L'UrbanDataset progettato al fine di supportare queste esigenze è descritto nella seguente tabella:

Tabella 4 - Intestazione dell'UrbanDataset "POD KPI PELL"

UrbanDataset	**POD KPI PELL**
Scopo	Fornire KPI calcolati a livello di POD appartenenti a impianti di illuminazione pubblica
Categoria / Sottocategoria	BuiltEnvironment/LandUseAndManagement
	SPECIFICAZIONE DELL'URBANDATASET
UrbanDatasetId	PODKPIpell-1.0
UrbanDatasetUri	https://smartcityplatform.enea.it/SCPSWebLibrary/urbandataset?name=PODKPIpell

[1] Area che necessita di uguali prestazioni illuminotecniche per quanto riguarda l'illuminazione artificiale al fine di garantire la sicurezza della circolazione veicolare o pedonale in primis o per altre esigenze.

UrbanDataset	**POD KPI PELL**
UrbanDatasetName	POD KPI PELL

Tabella 5 – Contestualizzazione dell'UrbanDataset "POD KPI PELL"

		CONTESTUALIZZAZIONE DELL'URBAN DATASET	
	Unità di Misura:	**Formato**	**Descrizione / Esempio**
Timestamp	Adimensionale	Date-Time aaaa-mm-ggThh:mm:ss	Tempo di generazione dell'UrbanDataset
Timezone	Adimensionale	String	Es. UTC, UTC+01, UTC-05
Coordinates	Adimensionale	WGS84 (World Geodetic System)	Coordinate del centro geometrico dell'area per la quale sono forniti i KPI.
Producer	Adimensionale	String	Identificatore del Sistema che ha prodotto i dati
		PROPRIETÀ DELL'URBANDATASET	
	Unità di Misura:	**Formato**	**Descrizione / Esempio**
PellKPICode	Adimensionale	String (vincolata a lista di codici)	KPI tecnici specifici del PELL IP
KPIValue	Adimensionale	Double	Valore calcolato per il KPI specifico
EvaluationIndicator	Adimensionale	String	Valutazione del valore del KPI/indicatore es. 'ok', 'fuori soglia', 'non disponibile'
TownCode	Adimensionale	String (vincolata a lista di codici)	Codice ISTAT del comune
PODID	Adimensionale	String	Codice POD che identifica univocamente il punto di prelievo
PLHomogeneousAreaID	Adimensionale	String	Identificatore della zona omogenea
Start Period	Adimensionale	Date-Time aaaa-mm-ggThh:mm:ss	Data/Ora di inizio del periodo di rilevazione (es. 2017-10-03T 14:00:00). Questo è compreso nell'intervallo di misurazione.
End Period	Adimensionale	Date-Time aaaa-mm-ggThh:mm:ss	Data/Ora di fine periodo di rilevazione (es. 2017-10-03T14:30:00). Questo istante NON è compreso nell'intervallo di misurazione.

Si noti che, coerentemente con l'approccio descritto precedentemente, il modello dell'UrbanDataset è tale per cui ogni KPI inviato richiede, a livello di istanza, una riga di UrbanDataset. Questo approccio può generare un'elevata ridondanza di alcune informazioni (codice ISTAT, codici identificativi del POD e della Zona Omogenea vengono ripetuti in ogni riga, quindi per ogni KPI inviato); tuttavia è stato scelta questa soluzione poiché è stato ritenuto che questo "svantaggio" sia ampiamente compensato dalla scalabilità del modello che consente di inviare qualsiasi KPI purché sia riferito ad una coppia città-POD.
La lista di codici a cui è stata vincolata la proprietà "*PellKPICode*" è stata resa disponibile in formato Genericode in Appendice B.2.

Anche la lista di codici a cui è stata vincolata la proprietà "*TownCode*" è stata resa disponibile in formato Genericode; a causa della numerosità dei valori, la tabella qui riportata ne include solo

alcuni a titolo di esempio. Si precisa che, come indicato nel campo "Agency", i valori utilizzati provengono dalla lista ufficiale fornita dall'ISTAT[2].

Tabella 6 - Tabella di esempio della lista di codici "TownCode"

Nome sintetico	TownCode
Versione	1.0
URI	TownCode.gc
Agency	ISTAT
Codice	**Valore**
01001001	Agliè
01001002	Airasca
01001003	Ala di Stura
01001004	Albiano d'Ivrea
01001006	Almese
...	....

UrbanDataset CityPublicLightingIndicators

L'UrbanDataset CityPublicLightingIndicators può essere usato da una piattaforma di gestione degli impianti di pubblica illuminazione per inviare indicatori urbani relativi a impianti di illuminazione pubblica calcolati a livello di città (o di una sua parte censita); l'invio può avvenire verso una SCP o una generica Solution che ne faccia richiesta.
Il flusso di dati è descritto nella seguente figura.

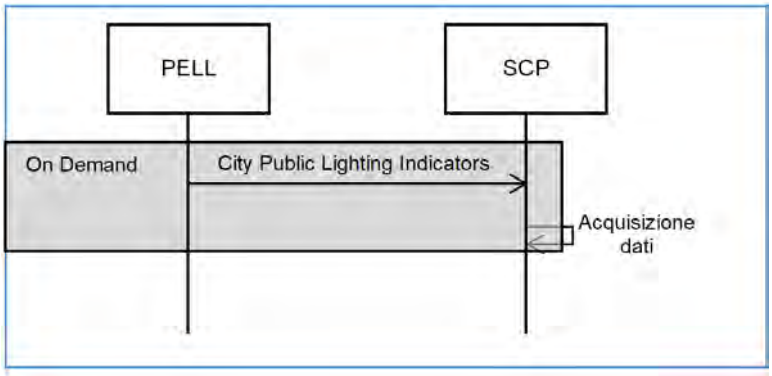

Figura 18 - Diagramma di sequenza UML del caso studio Smart Lighting "PELL" - invio indicatori urbani

Nello specifico, i seguenti **requisiti** individuati grazie all'analisi del caso studio sono i seguenti:

- ogni indicatore inviato deve essere riferito ad una città, identificata tramite codice ISTAT
- l'insieme degli indicatori supportati dall'UrbanDataset deve essere scalabile nel tempo (deve ciò essere possibile aggiungere nuovi indicatori, purché siano riferibili ad una città)
- in una stessa istanza dell'UrbanDataset non deve essere possibile riportare informazioni relative a più città.

L'UrbanDataset progettato al fine di supportare queste esigenze è descritto nella seguente tabella.

[2] https://www.istat.it/storage/codici-unita-amministrative/Elenco-comuni-italiani.xls

Tabella 7 - Intestazione dell'UrbanDataset "City Public Lighting Indicators"

UrbanDataset	City Public Lighting Indicators
Scopo	Fornire indicatori relativi a impianti di illuminazione pubblica calcolati a livello di città (o della sua parte censita). In una stessa istanza di questo UD non possono essere presenti indicatori relativi a città diverse
Categoria / Sottocategoria	BuiltEnvironment/LandUseAndManagement
	SPECIFICAZIONE DELL'URBANDATASET
UrbanDatasetId	CityPublicLightingIndicators-1.0
UrbanDatasetUri	https://smartcityplatform.enea.it/SCPSWebLibrary/urbandataset?name=CityPublicLightingIndicators
UrbanDatasetName	City Public Lighting Indicators

Tabella 8 - UrbanDataset "City Public Lighting Indicators"

		CONTESTUALIZZAZIONE DELL'URBANDATASET	
	Unità di Misura:	**Formato**	**Descrizione / Esempio**
Timestamp	Adimensionale	Date-Time aaaa-mm-ggThh:mm:ss	Tempo di generazione dell'UrbanDataset
Timezone	Adimensionale	String	Es. UTC, UTC+01, UTC-05
Coordinates	Adimensionale	WGS84 (World Geodetic System)	Coordinate del centro geometrico dell'area per la quale sono forniti gli indicatori.
Producer	Adimensionale	String	Identificatore del Sistema che ha prodotto i dati
		PROPRIETÀ DELL'URBANDATASET	
	Unità di Misura:	**Formato**	**Descrizione / Esempio**
CityPLIndicatorCode	Adimensionale	String (vincolata a lista di codici)	Indicatore urbano su illuminazione pubblica
TownCode	Adimensionale	String (vincolata a lista di codici)	Codice ISTAT del comune
Start Period	Adimensionale	Date-Time aaaa-mm-ggThh:mm:ss	Data/Ora di inizio del periodo di rilevazione (es. 2017-10-03T 14:00:00). Questo è compreso nell'intervallo di misurazione.
End Period[3]	Adimensionale	Date-Time aaaa-mm-ggThh:mm:ss	Data/Ora di fine periodo di rilevazione (es. 2017-10-03T14:30:00). Questo istante NON è compreso nell'intervallo di misurazione.

La lista di codici a cui è stata vincolata la proprietà "*TownCode*" è quella mostrata in tabella 5 poiché la proprietà è la medesima usata dall'UrbanDataset PellKPICode.
La lista di codici a cui è stata vincolata la proprietà "*CityPLIndicatorCode*" è stata resa disponibile in formato Genericode ed è riportata in Appendice B.3

[3] La misurazione relativa ad un certo periodo si ferma all'istante immediatamente precedente a quello indicato dal timestamp di fine periodo. Questo implica che il timestamp di fine periodo del periodo N deve essere uguale al timestamp di inizio periodo del N+1, ma le misure relative a questo istante devono essere considerate solo per il periodo N+1.

3.3 Evoluzione UrbanDatasetGateway

La comunicazione via web service RESTful *UrbanDatasetGateway*, prevede due nuovi metodi:

- *lastRequest*: per recuperare l'ultimo UrbanDataset prodotto in una certa collaborazione (ciò è particolarmente utile nella connessione tra SCP/iSCP e le dashboard realizzate con il componente SCP-DASH, si veda cap.4);
- *searchingRequestByProperty*: evoluzione del metodo searchingRequest, che permette di raffinare la ricerca ulteriormente, ricercando tra le proprietà dell'UD quelle che fanno match con la proprietà data (questo metodo può risultare molto utile in diversi ambiti, sia nel caso PELL per il recupero degli UrbanDataset secondo un dato codice ISTAT (si veda par.3.2.3), sia nella definizione delle dashboard realizzate con il componente SCP-DASH, si veda cap.4).

Riportiamo in questo paragrafo le descrizioni riprese dalla specifica aggiornata *SCPS Communication 2.0* [7].

3.3.1 *lastRequest* REST method

Il metodo "lastRequest" permette di richiedere l'ultimo UrbanDataset generato tramite una chiamata REQUEST/RESPONSE.

Tabella 9 – lastRequest REST method

	lastRequest method
Servizio	[URL_BASE]/UrbanDatasetGateway
Metodo	[URL_BASE]/UrbanDatasetGateway/lastRequest
Tipo Chiamata	POST
	HEADER
Content-Type	application/json
Accept	application/json
Authorization	Bearer [JWT-TOKEN] Dove [JWT-TOKEN] = token di autenticazione per l'accesso ai dati, rilasciato dal metodo di login.
	BODY
Parametri	resource_id=[string/resource_id] identifica univocamente un UrbanDataset prodotto da una specifica Solution producer (sintassi definita nella specifica SCPS Collaboration)
Esempio	{ "resource_id":"SCP-1_SmartBuildingCasaccia-3_SmartBuildingAnomalies-1.0_20180125120000" }

	lastRequest method
	RETURN
Success	{ “code”: “03”, “message”: “Request-Response Successful”, “dataset”: { "UrbanDataset": { … } } } Dove [{ "UrbanDataset": { … } }] è la rappresentazione JSON di un singolo UrbanDataset secondo la specifica SCPS Information.
Failure	{ “code”: “[CODE]”, “message”: “[MSG]”, “dataset”: } Dove [CODE] e [MSG] assumono specifici valori dipendentemente dal tipo di errore riscontrato (si veda la lista completa dei codici di ritorno in Appendice A della specifica SCPS Communication 2.0 [7]).

Il WS, nel metodo *lastRequest*, deve effettuare una serie di controlli:

1. il token JWT deve essere valido e da esso si può recuperare il ruolo dello user mittente;
2. lo user deve essere abilitato a produrre quell’UD;
3. il resource_id deve essere consistente in ognuna delle 4 parti che lo compongono (la sintassi del *resource_id* è definita nella specifica SCPS Collaboration 2.0 [6]).

3.3.2 *searchingRequestByProperty* REST method

Il metodo “*searchingRequestByProperty*” permette di ricercare uno o più UrbanDataset tramite una chiamata REQUEST/RESPONSE, fornendo l’identificatore della risorsa, con raffinamento spazio-temporale della ricerca a livello di context (elemento di contestualizzazione UrbanDataset presente nel formato), ricercando tra le proprietà dell’UD quelle che fanno match con la proprietà data.
Questo metodo è un’evoluzione del metodo *searchingRequest*.

Tabella 10 – searchingRequestByProperty REST method

	searchingRequestByProperty method
Servizio	[URL_BASE]/UrbanDatasetGateway
Metodo	[URL_BASE]/UrbanDatasetGateway/searchingRequestByRequest
Tipo Chiamata	POST
	HEADER
Content-Type	application/json
Accept	application/json
Authorization	Bearer [JWT-TOKEN]

	searchingRequestByProperty method
	Dove [JWT-TOKEN] = token di autenticazione per l'accesso ai dati, rilasciato dal metodo di login.
	BODY
Parametri	resource_id=[string/resource_id] identifica univocamente un UrbanDataset prodotto da una specifica Solution producer (sintassi definita nella specifica SCPS Collaboration)
	property_label=[STRING] Nome della proprietà su cui si dovrà effettuare la ricerca.
	property_value=[STRING] Valore della proprietà su cui si dovrà effettuare la ricerca.
	period_start*=[DATETIME] Data e ora da cui si vuole specificare la partenza di un intervallo temporale. Nel caso sia assente, la finestra temporale include tutti i valori possibili.
	period_end*=[DATETIME] Data e ora da cui si vuole specificare la fine di un intervallo temporale. Nel caso sia assente, la finestra temporale include tutti i valori possibili da period_start fino al momento in cui viene effettuata la chiamata.
	center_latitude*=[COORDINATE_WGS84] Latitudine del centro su cui si effettuerà la ricerca spaziale.
	center_longitude*=[COORDINATE_WGS84] Longitudine del centro su cui si effettuerà la ricerca spaziale.
	distance*=[DISTANCE_WGS84] Raggio del cerchio, espresso in metri, su cui si effettuerà la ricerca spaziale.
	* parametri opzionali
	RETURN
Success	{ "code": "03", "message": "Request-Response Successful", "dataset": [{ "UrbanDataset": { … } }, { … }] } Dove [{ "UrbanDataset": { … } }, { … }] è la rappresentazione JSON di uno o più UrbanDataset secondo la specifica SCPS Collaboration.

	searchingRequestByProperty method
Failure	{ “code”: “[CODE]”, “message”: “[MSG]”, “dataset”: [ ] } Dove [CODE] e [MSG] assumono specifici valori dipendentemente dal tipo di errore riscontrato (si veda la lista completa dei codici di ritorno in Appendice A della specifica SCPS Communication 2.0 [7]).

Il WS, nel metodo *searchingRequestByProperty*, deve effettuare una serie di controlli:

1. il token JWT deve essere valido e da esso si può recuperare il ruolo dello user mittente;
2. lo user deve essere abilitato a produrre quell’UD;
3. il *resource_id* deve essere consistente in ognuna delle 4 parti che lo compongono (la sintassi del *resource_id* è definita nella specifica SCPS Collaboration);
4. l’UD deve essere valido secondo il json-schema (la sintassi degli UD è definita nella specifica SCPS Collaboration);
5. l’UD deve indicare nell’elemento “context/producer/ID” il corretto ID del client

N.B. nel caso più tipico in cui il WS sia pubblicato sulla SCP,

- il *resource_id* è dato dalla SCP nel momento in cui si definisce la produzione di un UD;
- il producer/ID corrisponde al solutionID.

Esempio: la Solution “WebGIS Casaccia” invoca il metodo “*searchingRequestByProperty*” per recuperare dalla Smart City Platform “Smart Village Casaccia” gli UrbanDataset “Building Energy Consumption”, la cui proprietà “BUILDINGNAME” è uguale al valore “Edificio A”, prodotti nell'ultima settimana dalla Solution "Smart Building Casaccia".

L’implementazione e il test dei due nuovi metodi verrà realizzata nella Linea di Attività 1.21 (anno 2021).

4 Progettazione e Sviluppo SCP-DASH

Il componente SCP-DASH è un modulo software che permette di ottenere una o più dashboard di visualizzazione dati associate a un prototipo SCP e, quindi, anche alla inter-SCP agente su scala nazionale.
In questo capitolo viene descritta la progettazione e lo sviluppo relativi al componente SCP-DASH.

4.1 Progettazione SCP-DASH

La progettazione della dashboard per SCP/iSCP è avvenuta parallelamente alla definizione del caso studio pilota descritto nel cap.2; anche riguardo questo componente si è definita una serie di casi particolari di dati, basati su UrbanDataset reali e simulati, e si è delineata un'ipotesi di visualizzazione in modo tale da focalizzare fin da subito l'obiettivo a cui tendere.
In altre parole, se il caso studio pilota "Da SCP-Casaccia a inter-SCP" fornisce una descrizione generale onnicomprensiva, i casi studio per SCP-DASH sono più specifici del servizio finale da realizzare e ipotizzano alcune soluzioni per la visualizzazione dei dati sia su scala urbana che su scala nazionale, progettando un set di viste da poter riutilizzare, abbinandole agli UrbanDataset desiderati.
Particolarmente importante, in questa direzione, è stato lo studio sullo "Stato dell'Arte Servizi di Visualizzazione Dati" condotto nella LA 1.19 (annualità 2019, si veda il relativo report [14] al cap.7).

4.1.1 Caso Studio "view immediata di proprietà"

Questo è considerato il caso più semplice, presente sulla SCP-Casaccia, si tratterebbe di:
- recuperare periodicamente un singolo UrbanDataset;
- visualizzare alcune proprietà.

Esempio: UrbanDataset "WeatherCondition"
Viene inviato ogni 6 ore, contiene le seguenti informazioni meteo (1 linea):

TEMPERATURA DELL'ARIA. (AIRTEMPERATURE) 10.9 degreeCelsius

NOME DELLA STAZIONE METEO CHE HA FORNITO I DATI. (METEOSTATIONNAME) Bracciano-Lungolago

DATI SULLE PRECIPITAZIONI. (RAINFALL) 0.0 millimetrePerHour

A questo punto, si dovrà configurare una tabella che
- recupera l'ultimo UD e prende la prima linea di valori;
- visualizza le proprietà mappate nel file di configurazione.

Tabella 11 – View immediata di proprietà

Meteo ultime 6 ore		
Stazione Meteo	Bracciano-Lungolago	
Temperatura	10.9	degreeCelsius
Precipitazioni	0.0	millimetrePerHour
		Aggiornato alle 12:15:07 del 23/03/2020

Note:
Si preveda un tooltip a comparsa che riporti la descrizione del campo
(p.es. "Temperatura dell'aria rilevata esternamente" su "Temperatura").
Il titolo della tabella sarà ricavato dal file di configurazione.

“Aggiornamento alle [TIMESTAMP]” è processato a runtime.

4.1.2 Caso Studio “view di un calcolo”

In questo caso, abbiamo sempre una visualizzazione tabellare
ma il dato da visualizzare è il risultato di un calcolo algoritmico.

Esempio: UrbanDataset “Building Electric Consumption”; si vuole ottenere una media che sia risultato della fusione di 2 o più “Building Electric Consumption”.
Viene inviato 1 volta al giorno, contiene le seguenti informazioni su N palazzi monitorati:

IDENTIFICATORE DEL PALAZZO. (BUILDINGID) 1

ETICHETTA ASSOCIATA AL PALAZZO. (BUILDINGNAME) Edificio A

CONSUMO ENERGIA ELETTRICA. (ELECTRICCONSUMPTION) 119.31175 kiloWatthour

A questo punto, si dovrà configurare una tabella che
- recupera gli UD in input;
- raggruppa gli input discriminando in base all’edificio;
- calcola la media dei consumi;
- visualizza le proprietà mappate nel file di configurazione.

Tabella 12 – View di un calcolo

Consumo elettrico medio quotidiano su Ultima Settimana			
Edificio A	ENEA Bologna	710.123	kiloWatthour
Edificio B	ENEA Bologna	810.123	kiloWatthour
F40	ENEA Casaccia	910.123	kiloWatthour
			Aggiornato alle 12:15:07 del 23/03/2020

Note:
Si preveda un tooltip a comparsa che riporti la descrizione del campo
(p.es. “Edificio A monitorato da ENEA Bologna” su “Edificio A”).
Il titolo della tabella sarà ricavato dal file di configurazione.
“Aggiornamento alle [TIMESTAMP]” è processato a runtime.

4.1.3 Caso Studio “Basic Line”

In questo caso, abbiamo una visualizzazione grafica con una o più curve che indicano un andamento (solitamente temporale) di un dato singolo.
Esempio: UrbanDataset “Building Electric Consumption”
Viene inviato 1 volta al giorno, contiene le seguenti informazioni su 2 palazzi monitorati:

IDENTIFICATORE DEL PALAZZO. (BUILDINGID) 1

ETICHETTA ASSOCIATA AL PALAZZO. (BUILDINGNAME) Edificio A

CONSUMO ENERGIA ELETTRICA. (ELECTRICCONSUMPTION) 119.31175 kiloWatthour

A questo punto, si dovrà configurare un Chart che
- ogni giorno recupera gli UD degli ultimi N giorni;
- discrimina le linee per proprietà (per palazzo in questo caso);
- preleva N valori, associandoli al giorno corrispondente;
- le label del titolo e degli assi sono pre-configurati.

Viene qui riportato, a solo titolo di esempio, il chart “Basic Line” di HighCharts.

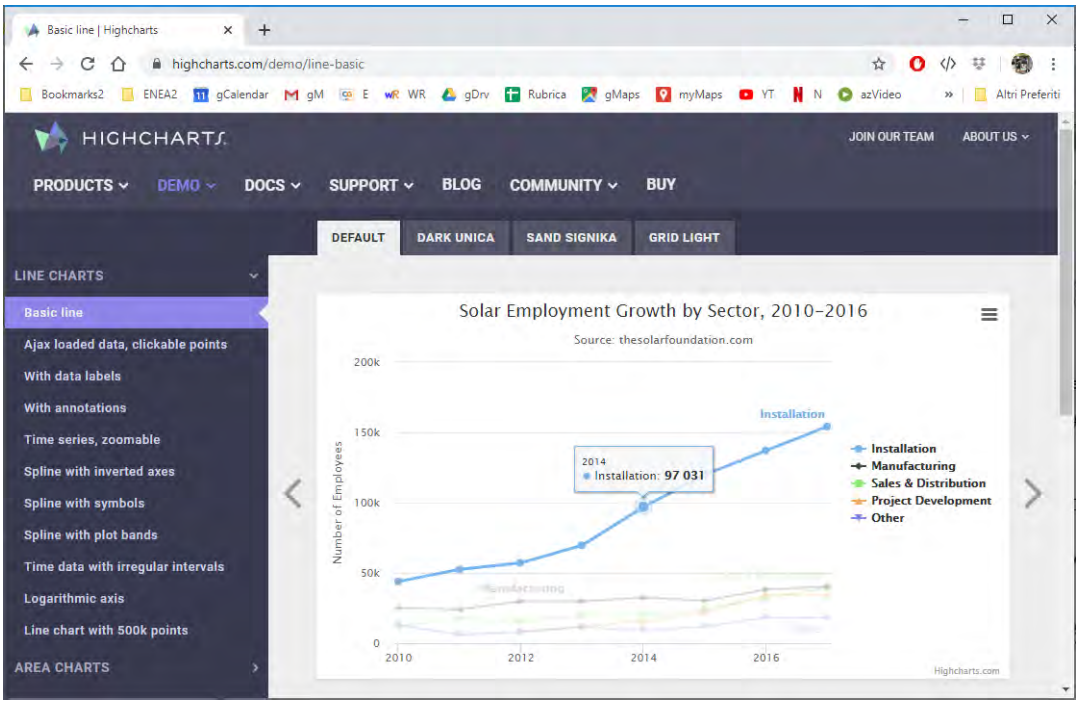

Figura 19 - Chart “Basic Line” di HighCharts

4.1.4 Caso Studio “Dual Axis”

In questo caso, abbiamo una visualizzazione grafica doppia, con curva e colonne, che indicano due andamenti (solitamente temporale) di 2 dati prelevati da uno o più Urban Dataset.

Esempio: UrbanDataset
“Home Aggregated Electric Production” e “Home Aggregated Electric Consumption”
Vengono inviati 1 volta al giorno, contengono le seguenti informazioni generiche:

SUPERFICIE CALPESTABILE. **(FLOORAREA)** 801.00

NUMERO OCCUPANTI. **(OCCUPANTNUMBER)** 29

CONSUMO ENERGIA ELETTRICA. **(ELECTRICCONSUMPTION)** 18.28 kilowattHour

A questo punto, si dovrà configurare un chart che
- ogni giorno recupera gli UD degli ultimi N giorni;
- se indicato, discrimina le linee per proprietà;
- preleva i valori, associandoli al giorno corrispondente;
- i consumi saranno rappresentati con le colonne, le produzioni con la curva;
- le label del titolo e degli assi sono pre-configurati.

Viene qui riportato, a solo titolo di esempio, il chart “Dual axes” di HighCharts.

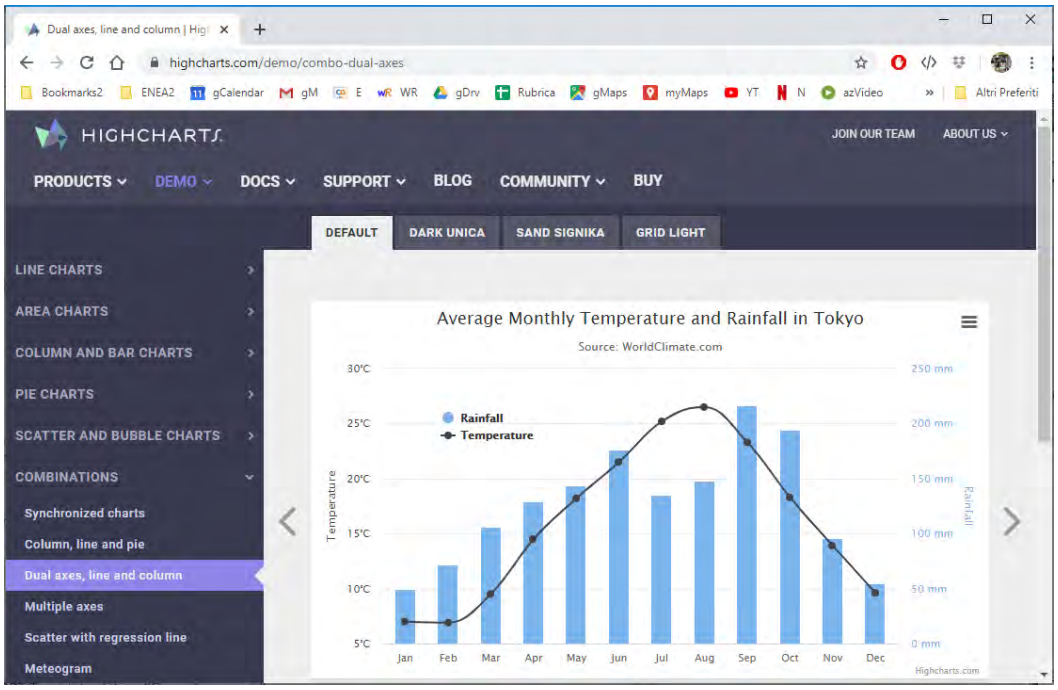

Figura 20 - Chart “Dual axes” di HighCharts

4.1.5 Caso Studio “Map Info”

Si tratta di visualizzare, in questo particolare caso sulla mappa, una serie di KPI inviati da sorgenti che sono geograficamente distribuite.

Esempi:

1) Caso Studio “Platform Status”: in questo caso ogni SCP invia quotidianamente un UD sullo stato corrente di funzionamento;

2) Caso Studio “KPI x iSCP”: in questo caso le sorgenti sono altre SCP che inviano alla iSCP ricevente:
- KPI su CO2 o PM10;
- KPI su consumo energetico per persona in ambito Building;

3) Caso Studio “KPI x PELL”:
- KPI consumo energetico giornaliero massimo misurato: è un indicatore che riporta il confronto tra il consumo misurato giornalmente e il consumo massimo teorico che si avrebbe se l’impianto funzionasse alla massima potenza durante il periodo di funzionamento stabilito dai tempi di attivazione e disattivazione del timer astronomico.

Possiamo ipotizzare riguardo questi esempi che gli UrbanDataset:
- vengano calcolati ogni giorno, sul giorno prima;
- vengano calcolati per ogni sorgente dati con un UrbanDataset apposito;
- che l’UD una volta esportato venga inviato alla SCP;
- che la SCP-DASH recuperi gli UD dalla SCP per la visualizzazione su mappa.

Viene qui riportato, a solo titolo di esempio, il chart “Advanced lat/long” di HighCharts. N.B. per la SCP-DASH si è ovviamente impostata la mappa dell’Italia (si veda cap.4).

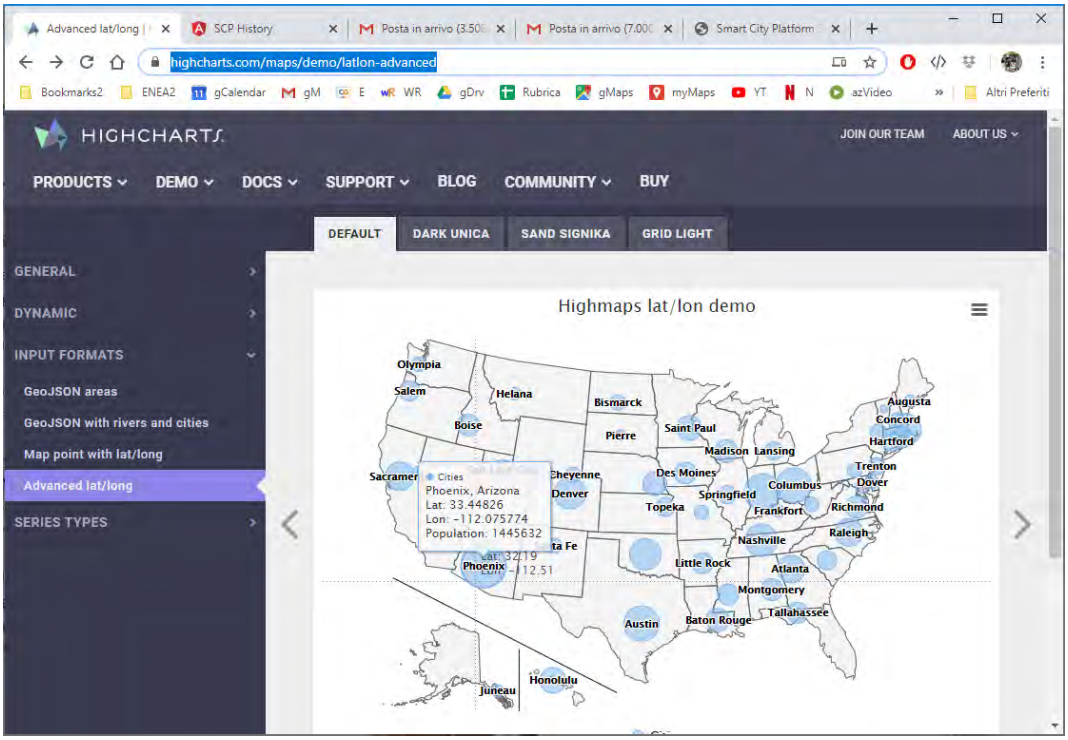

Figura 21 - Chart “Advanced lat/long” di HighCharts

4.2 Sviluppo SCP-DASH

4.2.1 Requisiti

Partendo dai casi studio per SCP-DASH presentati nel precedente paragrafo, è stato richiesto (a una software house esterna) di definire l'architettura di massima che descrivesse:

- come configurare i Report (composti da un insieme di tabelle e grafici);
- come gestire i meccanismi di aggregazione e data fusion;
- come recuperare i dati (UrbanDataset) dalla SCP, in adesione alle specifiche SCPS;
- come visualizzare i dati nei Report finali dell'interfaccia web, con un meccanismo che permetta un aggiornamento istantaneo;
- quale tecnologia utilizzare in fase di visualizzazione dati (tecnologia che abbia una ricca gallery di chart e schemi, per una eventuale estensione futura della dashboard, ad esempio highcharts.com).

Il meccanismo di configurazione dei Report deve astrarre i casi studio per SCP-DASH permettendo di implementare non solo i casi proposti ma anche i casi "simili", fornendo così una metodologia di creazione dashboard che non richieda ulteriore sviluppo di codice ma si limiti alla configurazione di quali Urban Dataset saranno recuperati e come saranno visualizzati.

Il componente software SCP-DASH permette le seguenti fasi di interazione:

1. Configurazione

Configurazione manuale su file di testo, o configurazione tramite interfaccia web, del file (o dei file) che descrivono 1 o più Report che dovranno essere visualizzati nella Dashboard; per ogni Report dovrà essere possibile descrivere tabelle o grafici, potendo indicare gli UrbanDataset di input e come mappare le relative proprietà con quelle dei grafici risultanti.

2. Login Utente

Utilizzando la bancadati preesistente "scps-registry", dovrà essere possibile, tramite l'interfaccia della dashboard, effettuare:

- Registrazione utente di tipo "Solution";
- Login utente di tipo "Solution" o "Administrator";

dipendentemente dall'utente la view dei Report potrebbe variare, a seconda dei permessi impostati sulle Solution e degli Urban Dataset recuperati.

3. Visualizzazione Report

Si sono ipotizzate più voci nel menù principale, in cui ogni voce corrisponde a un preciso Report.
Accedendo dunque a una delle voci del menù si potrà accedere a un intero Report composto da un insieme di tabelle e grafici.

4.2.2 Implementazione Software

Il sistema proposto è stato realizzato con le tecnologie indicate da ENEA.
In particolare, l'architettura software è così composta:

- Componenti client-side: il client HTML è realizzato in Angular/Bootstrap e adotta lo stile Angular Material. Nelle fasi di implementazione o modifica del client sono utilizzati Node.js e npm.
- Componenti server-side: l'applicazione web è realizzata in tecnologia Java, eseguita all'interno dell'application server Tomcat. Espone le proprie funzionalità tramite web service RESTful che codificano le informazioni in JSON. E' compreso un componente utilizzabile per definire la frequenza di aggiornamento di ogni grafico. E' inoltre incluso un componente utilizzabile per integrare ElasticSearch.
- La persistenza dei dati, resa tramite JPA/Hibernate, è realizzata su MySQL.

Le interfacce del sistema sono altamente configurabili, per cui sono state adattate al fine di adottare un layout simile all'applicazione SCP-GUI come richiesto. Le interfacce sono già responsive e quindi visualizzabili correttamente su device con risoluzioni e form factor eterogenei, come ad es. PC, tablet e smartphone. Il nuovo layout adotterà analogo comportamento. Il client HTML è compliant ai requisiti di accessibilità (livello AA). Il nuovo layout è stato sviluppato al fine di mantenere tale compatibilità.
La gestione utenti del componente è modulare, per cui è possibile personalizzare i flussi di registrazione e login, adottando i servizi di identity provider già in essere o accedendo direttamente a database esterni (ad es. Registry). Il sistema fornisce degli strumenti che permetteranno al personale ENEA di definire autonomamente nuovi report, secondo le ulteriori esigenze che potranno emergere.

Il componente SCP-DASH è configurabile tramite file .properties.
E' possibile indicare:

- Le pagine che mostrano i report. Ogni pagina corrisponde ad una voce di menu ed è composta da una etichetta, mostrata nella interfaccia HTML, e l'identificatore del report visualizzato al suo interno.
- Il menu Help.
- I dati di accesso all'identity provider.
- I dati di accesso all'UD gateway.

La definizione della configurazione dei report, elemento fondamentale per la specificazione di tutti gli input necessari, è realizzata tramite file JSON, che il personale ENEA può modificare manualmente.

Il sistema di gestione degli utenti del componente è configurato al fine di fornire le funzionalità di:

- registrazione utenti con ruolo SOLUTION.
- login nel sistema per utenti con ruolo ADMINISTRATOR e SOLUTION.

Queste funzionalità utilizzano l'identity provider già in essere ed accedono al database Registry.
L'accesso ai grafici del report è profilato rispetto all'utente loggato nel sistema e all'UD indicato nel datasource del grafico stesso. Ad esempio, se l'UD di input è dichiarato OPENDATA, il grafico risulta accessibile a tutti gli utenti, indipendentemente dal loro ruolo; in caso contrario il grafico del report è visibile a un utente solo se tale utente ha il permesso di accedere al relativo UD di input.

Al fine di dimostrare le funzionalità del componente, sono stati predisposti due report andando ad utilizzare tutte le tipologie di tabelle e grafici supportati (così come definiti nei casi studio e richiesti in fase di progettazione). Tali report sono dei "mock", ovvero basati su **dati simulati**, ma permettono di testare il funzionamento in maniera efficace utilizzando UrbanDataset di dati preparati appositamente (che, nella versione finale, saranno generati automaticamente e periodicamente, così come spiegato nei par.2.4 e par.2.5).

Nella LA 1.21 (annualità 2021) relativa alla sperimentazione finale, si configurerà il report finale per la dashboard della inter-SCP, in modo tale utilizzi in input UrbanDataset contenenti informazioni recuperate dalla SCP-Casaccia così come il caso studio pilota descrive.

In Appendice C viene riportato il listato completo di un file di configurazione report che illustra come ogni grafico/tabella (chart) abbia una configurazione apposita associata a uno specifico UrbanDataset.

Seguono alcune figure contenenti gli screenshot dei due report attualmente simulati nel componente SCP-DASH. Non si ritiene di fornire altra descrizione, in quanto i grafici sono per loro natura auto-esplicativi e riprendono le descrizioni definite nei casi studio per SCP-DASH (par. 4.1).

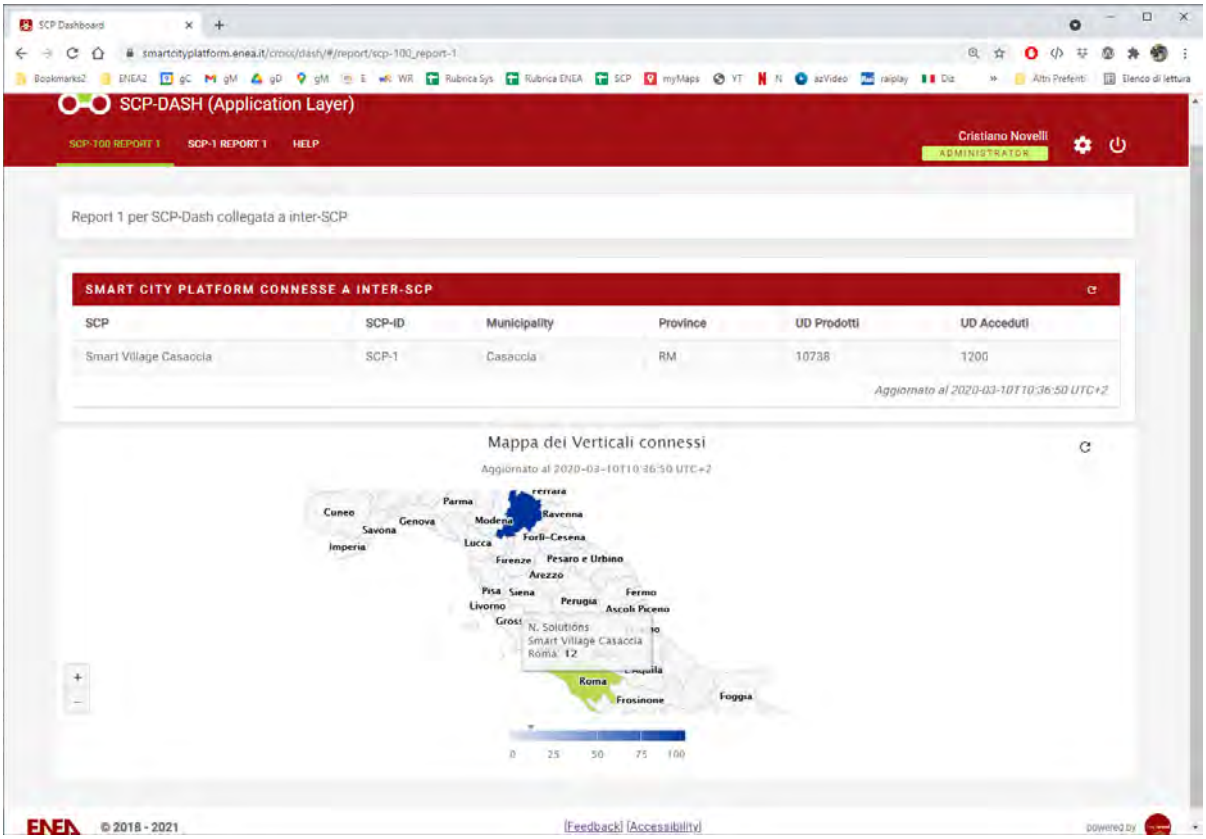

Figura 22 - SCP-DASH Report Screenshot 1

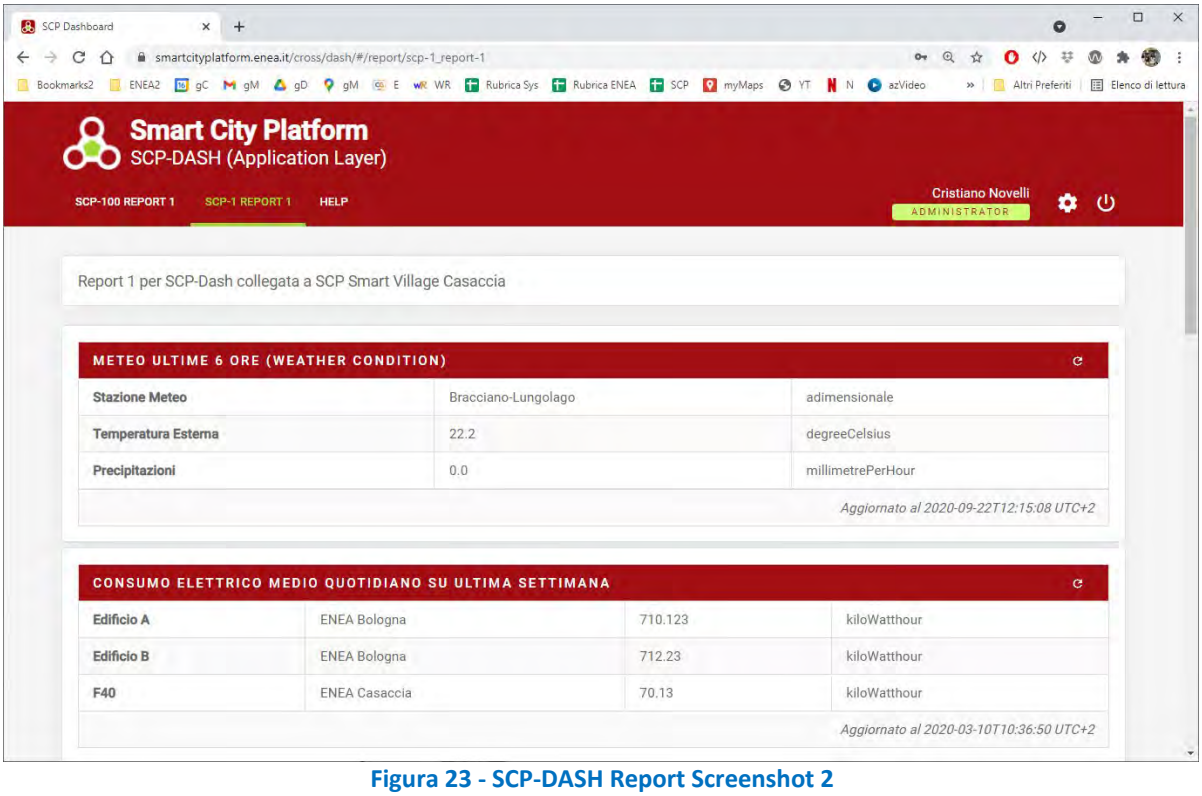

Figura 23 - SCP-DASH Report Screenshot 2

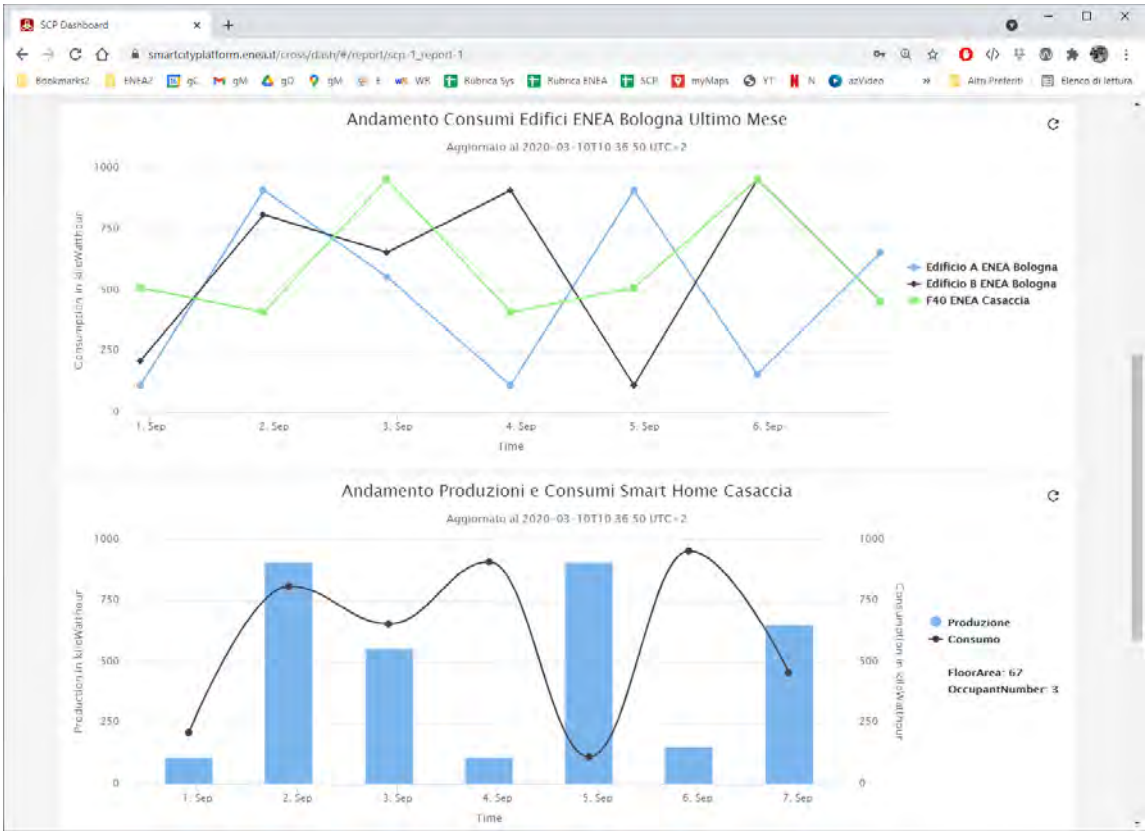

Figura 24 - SCP-DASH Report Screenshot 3

5 Integrazione SCP-PELL

Questo capitolo descrive il risultato dell'integrazione della piattaforma SCP con la piattaforma PELL seguendo lo studio di fattibilità condotto nella Linea di Attività 1.19 (annualità 2019).

5.1 Architettura SCP-PELL

Riprendiamo dallo studio di fattibilità eseguito nella LA 1.19 l'ipotesi finale di integrazione architetturale che è schematizzata nella seguente figura e che ha costituito l'obiettivo a cui tendere.

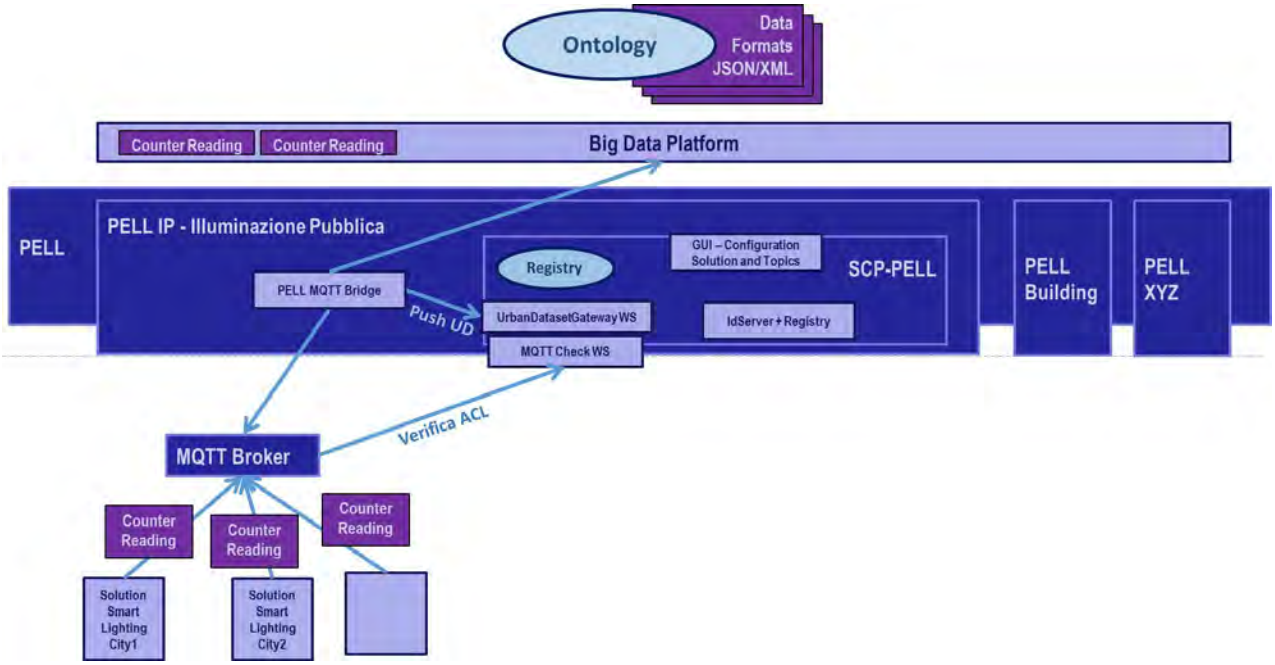

Figura 25 - Architettura PELL IP integrato con SCP-PELL

Nello schema architetturale si evince come sia stata prevista un'istanza di SCP apposita per il PELL, denominata SCP-PELL; essa ora permette 3 azioni principali nella piattaforma PELL:

1. configurare tramite GUI le Solution verticali che invieranno gli UrbanDataset associati a specifici "topic" MQTT;
2. verificare i permessi tramite il web service "MQTT Check";
3. ricevere tutti gli UrbanDataset ricevuti tramite Broker MQTT, che il PELL MQTT Bridge invierà al web service UrbanDatasetGetaway, con chiamata al metodo push, in pieno rispetto delle specifiche SCPS.

Queste 3 azioni sono esattamente le 3 fasi previste dallo studio di fattibilità, che nel prossimo paragrafo, descriveremo più dettagliatamente, facendo un riepilogo degli interventi di sviluppo software effettuati.

5.2 Interventi Sviluppo Software

Riportiamo dallo studio di fattibilità (LA 1.19) le tre fasi previste per l'integrazione SCP-PELL, i diversi step che le compongono e sottolineiamo dove sono richiesti gli interventi di sviluppo software, lato SCP.

Fase 1 Configurazione

1. Creazione di un account
2. Creazione di 1 o N Solution **[intervento sw]**
3. Creazione della produzione/accesso al topic **[intervento sw]**

Fase 2 Verifica (plug-in Mosquitto del Broker MQTT)

1. Autenticazione User(*checkUser*) **[intervento sw]**
2. Autenticazione Admin (*checkAdmin*) **[intervento sw]**
3. Verifica degli accessi in scrittura (*checkACL*) **[intervento sw]**

Fase 3 Invio Dati (PELL MQTT Bridge)

1. login (username, password)
2. Recupero del resource_id (*getResourceId*) **[intervento sw]**
3. Invio "Counter Reading" tramite push.

Per attuare tale integrazione, per alcuni di questi task, sono stati necessari alcuni interventi sul prototipo SCP/iSCP che elenchiamo nei prossimi paragrafi, facendo riferimento alla fase/task suddetti.

5.2.1 Creazione di 1 o N Solution

Il primo intervento riguarda la Fase1, Task2, "Creazione di 1 o N Solution".
Si tratta di una modifica della SCP-GUI necessaria per permettere l'associazione di un singolo account utente più di una sorgente verticali (mentre prima l'associazione era 1-1). Questa eventualità, per esempio con un unico gestore dell'illuminazione pubblica associato a diversi comuni, è qualcosa che potrà accadere spesso nella relativa gestione PELL.

La modifica non investe la banca dati Registry della SCP, in quanto era stata già prevista nel modello relazionale questa logica. Ciò che si è dovuto aggiornare è stata l'interfaccia della SCP-GUI per permettere di:

- creare una nuova Solution associandola a un account esistente;
- presentare all'utente una lista delle Solution a lui collegate;
- modificare l'url associato ad ogni Solution, se presente;
- cancellare una Solution;

seguendo l'approccio funzionale CRUD (Create, Read, Update, Delete) che la SCP-GUI ha adottato nelle diverse sezioni di gestione.

Riteniamo interessante mostrare questa implementazione con uno screenshot della inter-SCP che mostra questa funzione sviluppata (figura seguente). Si noti come tutti gli interventi software che vengono effettuati sul prototipo SCP/iSCP favoriscono non solo l'istanza che ha fatto nascere la particolare esigenza (in questo caso SCP-PELL) ma anche tutte le altre istanze SCP che sfruttano il componente oggetto di intervento.

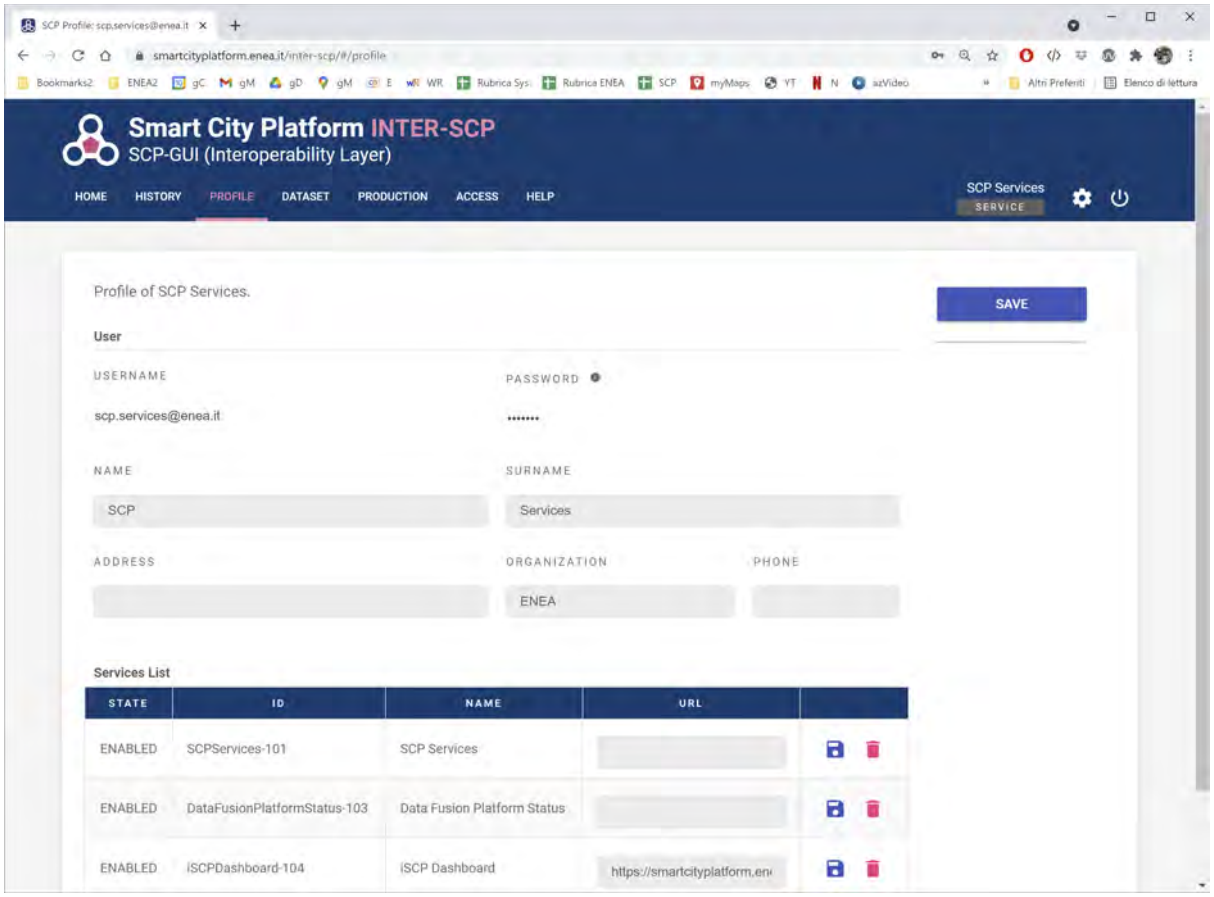

Figura 26 - Associazione di 3 Services per un singolo account utente

Nella lista dei Services (che come sappiamo vengono trattati come Solution interne della SCP) possiamo notare i moduli che sono stati individuati e configurati per il caso studio pilota (si veda par.2.6).

5.2.2 Creazione della produzione/accesso associati a un topic

Questo intervento riguarda la Fase1, Task3, "Creazione della produzione/accesso al topic".
Questa modifica è quella più delicata e rappresenta il cuore dell'integrazione: grazie a questo intervento, infatti, è possibile associare un topic MQTT a una produzione di UrbanDataset verso la SCP-PELL.
Durante i lavori di sviluppo è stata fatta un'ulteriore riflessione: perché non astrarre maggiormente il concetto di "SCP-Bridge" e di integrazione al protocollo MQTT?
Si è deciso quindi di non chiamare "topic" l'informazione da associare al recupero della risorsa tramite bridge, ma di definirla come generica informazione "extra" che,

- nel caso dell'integrazione SCP-PELL avrebbe assunto esattamente la valenza del *topic*;
- nel caso della comunicazione interoperabile tra SCP e inter-SCP avrebbe assunto come valore il *resource_id* dell'UrbanDataset remoto da recuperare.

Ciò ha permesso di astrarre ulteriormente il discorso di definizione dello SCP-Bridge arrivando infine a concepire lo SCP-Scheduler, grazie al quale è possibile configurare un SERVICE che svolga le funzioni previste dallo SCP-Bridge e molti altri SERVICE (come abbiamo visto nella descrizione del caso studio pilota nel cap.2).

L'intervento ha riguardato la rimodulazione della tabella *communication_endpoint* del registry (dove vengono ora salvati sia gli endpoint destinazione che il bridge deve raggiungere, sia l'informazione "extra" che identifica la risorsa da recuperare.

Quindi, riassumendo l'intervento, tramite la SCP-GUI modificata, è possibile inserire la configurazione, immagazzinata nella banca dati Registry modificata, che permette a un modulo software di recuperare una risorsa ben precisa da un sistema remoto:

- nel caso dell'integrazione SCP-PELL, il modulo software corrisponde al "PELL Bridge", che abbiamo visto nell'architettura dell'integrazione (par.5.1), e il sistema remoto è il Broker MQTT;
- nel caso della comunicazione tra inter-SCP e SCP, il modulo software corrisponde a un SERVICE (p.es. "Get Platform Status" descritto nel par.2.3), che abbiamo visto (ovvero il caso studio pilota, descritto nel cap.2), e il sistema remoto è la piattaforma urbana SCP che deve inviare l'UD alla inter-SCP.

Si noti che nessun intervento di modifica software al prototipo SCP è effettuato per gestire un caso specifico ma l'intento è quello di effettuare modifiche e miglioramenti che possano essere utili in una casistica di applicazioni della SCP più ampia possibile (abbiamo constatato ciò anche con la definizione dello SCP-Scheduler, valido sia per la inter-SCP su scala nazionale, così come per la SCP su scala urbana).
Riportiamo uno screenshot relativo all'interfaccia SCP-GUI che ora permette di configurare le produzioni con specificazione del bridge (nell'esempio è stata configurata una produzione di test che permette agli UrbanDataset di tipo "Whatever" inviati al MQTT Broker dalla solution "TestSolution" di essere recuperati tramite PELL Bridge utilizzando le credenziali di accesso al broker e il topic specificato).

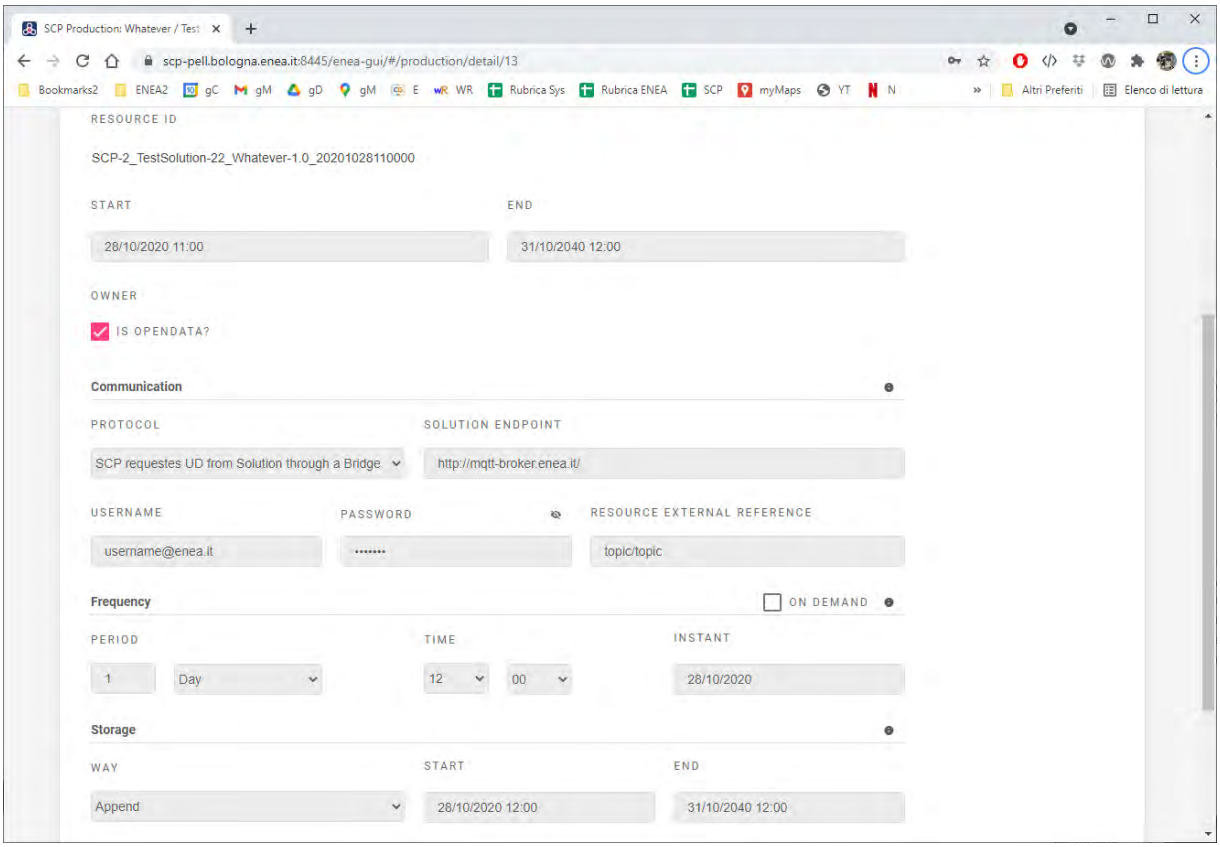

Figura 27 - Creazione di una Produzione con "Bridge"

I test relativi alla Fase 1 dell'integrazione hanno avuto esito positivo.

Nella LA 1.21 (annualità 2021) si estenderà questa modifica anche alla definizione dell'accesso con "bridge", inteso come la funzionalità di un SERVICE di poter accedere a uno o più UrbanDataset locali per inviarli a sistemi remoti (anche con questo pattern sarà utilizzato lo SCP-Scheduler).
A questo punto, la casistica relativa alle possibili combinazioni dei "pattern di comunicazione" (si veda la specifica SCPS Communication [7], cap.2) sarà completata e pienamente supportata nel prototipo SCP.
Riassumiamo qui di seguito la casistica dei possibili pattern di comunicazione che è già memorizzata nella tabella *communication_supported* del Registry:

1) 'SOLUTION/SERVICE pushes UD': pattern “push” in cui la Solution funge da client e la SCP funge da server per l’invio diretto di UrbanDataset (è il pattern “classico” più utilizzato);
2) 'SOLUTION/SERVICE accesses UD': pattern “request/response” in cui la Solution funge da client e la SCP funge da server per l’accesso diretto a UrbanDataset (è il pattern “classico” più utilizzato);
3) 'SERVICE pushes UD from SOLUTION': pattern “request/response” in cui la Solution funge da server e la SCP funge da client (tramite SERVICE) per l’accesso a UrbanDataset remoto (p.es. PELL Bridge) e push locale alla SCP chiamante;
4) 'SERVICE accesses UD and push it to SOLUTION': pattern “push” in cui la Solution funge da server e la SCP funge da client (tramite SERVICE) per l’accesso locale a UD e l’invio alla Solution.

5.2.3 Verifica ACL con plug-in Mosquitto

Questo intervento è relativo all’intera Fase 2 che prevede la verifica dei permessi di produzione per permettere a una Solution di scrivere UrbanDataset su uno specifico topic verso il broker MQTT.
Il broker MQTT utilizza un plug-in di Mosquitto che effettua tre azioni:

1. Autenticazione User(*http_checkUser_uri*) [post]
2. Autenticazione Admin (*http_checkAdmin_uri*) [post]
3. Verifica delle ACL (*http_aclcheck_uri*)[post]

Queste tre azioni sono descritte più dettagliatamente nello studio di fattibilità (nel report LA 1.19, par.4.4).

L’intervento ha richiesto lo sviluppo del web service “MQTT Check” che espone i tre metodi previsti, corrispondenti alle azioni che il Broker MQTT richiamerà:

1. *checkUser(username, password*);
2. *checkAdmin(username, password*);
3. *checkACL(username, clientId, topic, acc*);

la cui descrizione è stata già provvista nello studio di fattibilità.

Il web service è stato sviluppato in due progetti:
- il componente server da installare nella SCP (a fianco del già presente WS UrbanDatasetGateway);
- il componente client per effettuare tutti i test del web service;

per massimizzare la compatibilità, le tecnologie utilizzate sono le stesse usate per lo sviluppo dell’*UrbanDatasetGateway*, ovvero il linguaggio Java/J2EE, con IDE Eclipse, su Apache Tomcat 8.
Una volta sviluppato e installato il web service, il team che lavora sulla piattaforma PELL ha proceduto con il completamento dell’integrazione, andando a testare le chiamate del plugin del Broker MQTT verso il web service MQTT Check.

I test relativi alla Fase 2 dell’integrazione hanno avuto esito positivo.

5.2.4 Recupero del resource_id

Questo intervento riguarda la Fase3, Task2, “Recupero del resource_id”.

Riprendiamo dallo studio di fattibilità la descrizione dell’intervento:

- creare l’account di tipo Administrator per il PELL MQTT Bridge;
- implementare nel WebService *UrbanDatasetGateway* il metodo *getResourceId(jwt-token, topic)* che, dato il topic, ritorna il *resource_id* relativo.

Lato piattaforma SCP, il metodo è stato sviluppato all’interno dell’*UrbanDatasetGateway* e testato.
Lato piattaforma PELL, la chiamata al metodo è stata integrata nel PEL Bridge e testata con successo.

Si noti che questa serie di modifiche non solo hanno permesso l'integrazione tra SCP e PELL ma hanno anche dotato il prototipo SCP di un nuovo protocollo di trasporto con MQTT.

5.2.5 Persistenza

Oltre agli interventi di sviluppo software pianificati nello studio di fattibilità condotto nella LA 1.19 ed eseguiti con successo nella LA 1.20 in oggetto, è stato risolto anche il seguente punto aperto:

"questa prima integrazione comporta una ridondanza dei dati in due sistemi (piattaforma Big Data e SCP-UD-DB basato su ElasticSearch). Si può pensare in futuro, in caso di integrazione avvenuta con successo, a una convergenza anche su questo fronte."

Tale convergenza è stata già risolta: gli UrbanDataset che vengono ricevuti dalla SCP-PELL vengono inviati alla piattaforma big data basata su tecnologia Apache Hadoop utilizzando la medesima interfaccia basata su Elastic Search. In questo modo non vi è ridondanza dei dati e l'integrazione è divenuta ancor più completa andando ad utilizzare lo stesso sistema di persistenza dei dati.

5.3 SCP-PELL

La seguente figura è uno screenshot della SCP-PELL, installata, configurata e testata in ognuna delle tre fasi descritte nel paragrafo precedente, in combinazione con i componenti MQTT Broker e PELL Bridge.

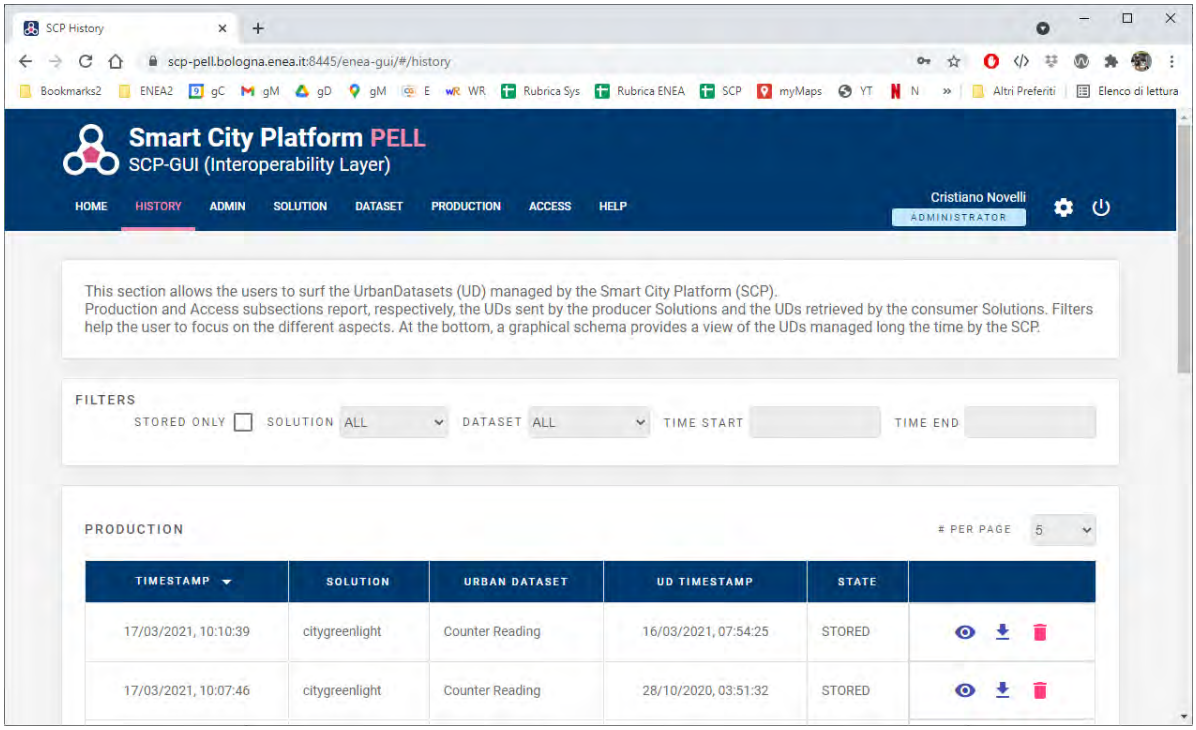

Figura 28 - SCP installata e configurata per il PELL

Si noti che, nella sezione "history" in figura, sono correttamente memorizzate le ricezioni di due UrbanDataset "Counter Reading" (relativi ai consumi di energia elettrica) utilizzando il meccanismo con broker MQTT, PELL e SCP-PELL.

Abbiamo descritto nel par.5.2 gli interventi software relativi al prototipo SCP per l'integrazione con il PELL. Gli interventi software relativi alla piattaforma PELL sono descritti nel report "Piattaforma PELL: dati statici e dinamici PELL IP e dati statici PELL Edifici Scuole" [17].

6 Guida di Adesione al Framework

In questo capitolo descriviamo la guida di adesione al framework SCP, pubblicata online sul sito ufficiale del progetto [18]. Riportiamo integralmente la sotto-sezione "*Adesione con una SCP alla inter-SCP*" che è l'unico percorso di adesione (dei tre disponibili) relativo alla piattaforma inter-SCP agente su scala nazionale.

Questo percorso di adesione prevede che una SmartCityPlatform (SCP) di una città/distretto si debba connettere alla inter-SmartCityPlatform (abbreviato in "inter-SCP" o "iSCP"), piattaforma ICT sviluppata da ENEA, agente su scala nazionale (si veda l'architettura relativa alla comunicazione inter-city tra SCP e iSCP nella sezione "Architettura" del sito web) allo scopo di produrre o accedere a UrbanDataset (formato dati comune per rappresentare KPI o dati relativi alla gestione Smart City).

6.1 Requisito: Accordo formale tra le parti

Per avviare questo percorso è necessario stabilire un accordo formale tra le parti coinvolte:

- Referente della SmartCityPlatform (SCP);
- Referente della inter-SmartCityPlatform (inter-SCP), ovvero ENEA.

Al momento, ogni SCP definita con il prototipo di ENEA, su server di ENEA, è automaticamente idonea per avviare la connessione alla inter-SCP. Eventuali SCP sviluppate da terze parti (ma che comunque abbiano adottato le specifiche SCPS) o SCP installate su server esterni alla rete ENEA, saranno valutate prima di intraprendere il percorso di adesione per connettere una SCP urbana alla inter-SCP agente su scala nazionale. Per richiedere tutte le informazioni del caso, utilizzare il contatto che si trova nel menù principale in [Aiuto] > [Feedback] (presente anche nel footer di ogni pagina).

Prima di intraprendere questo percorso di adesione, è necessario che la SCP in oggetto abbia effettuato con successo i precedenti due percorsi di adesione:
1. Adesione con una nuova SCP;
2. Adesione con una Solution a una SCP.

L'accordo formale esprime, tra le altre cose, la lista degli UrbanDataset che la SCP invierà alla inter-SCP (p.es. l'UD "Platform Status").

Una volta che l'accordo formale è raggiunto e i requisiti sono stati verificati, la municipalità (o società) che gestisce la SCP dovrà compiere una serie di passi per instaurare la comunicazione con la inter-SCP utilizzando le specifiche SCPS (ovvero i passi seguenti).

Si noti che, nel caso in cui la SCP sia il prototipo SCP di ENEA, ed esso sia installato e configurato su server ENEA, per abilitare la comunicazione tra SCP e inter-SCP, TUTTI I PASSI SEGUENTI sono già configurati e implementati di default nei prototipi SCP e inter-SCP di ENEA per tutti gli UrbanDataset concordati nell'accordo formale tra le parti.

6.2 Passo 1: Adesione all'Architettura di Riferimento

La comunicazione interoperabile SCPS-based tra SCP e inter-SCP (iSCP) è equivalente alla comunicazione tra Solution e SCP così come descritta attraverso l'architettura di riferimento nel capitolo 2 della specifica SCPS Functional 2.0. Si richiede dunque, come primo passo informale, di leggere tale specifica, allo scopo di condividere l'architettura di riferimento e i concetti chiave (come SCP, Solution, UrbanDataset, Ontologia, Registry, ecc.)

Si noti che la comunicazione inter-city tra SCP e iSCP (N SCP connesse a 1 iSCP) è analoga alla comunicazione intra-city tra Solution e SCP (N Solution connesse a 1 SCP): in entrambi i casi si utilizzano le specifiche SCPS per implementare la comunicazione interoperabile.

Vi è però una differenza architetturale:
- nella comunicazione intra-city, tra Solution e SCP, è la Solution ad inviare gli UD alla SCP;
- nella comunicazione inter-city, tra SCP e inter-SCP, è la inter-SCP a recuperare gli UD dalla SCP.
Vedremo come ciò sarà configurato nel Passo 4, relativo alla configurazione della collaborazione.

6.3 Passo 2: Progettazione dei flussi dati

L'accordo formale preso con ENEA (si veda REQUISITO 0) prevede che la SCP coinvoltà invierà uno o più UrbanDataset alla inter-SCP.

Gli UrbanDataset che le SCP inviano alla inter-SCP sono definiti da ENEA.

Diversamente, il referente della SCP:
- dovrà valutare se tutti gli UrbanDataset richiesti sono realizzabili, ovvero se tutti i dati e KPI che dovranno essere inviati alla inter-SCP sono dati recuperabili o KPI calcolabili;
- nel caso gli UD richiesti non siano realizzabili o siano insufficienti, potrà contribuire al processo di modifica e/o definizione di nuovi UrbanDataset.

Nel caso di modifica o creazione di UrbanDataset, dovranno essere coinvolti anche gli specialisti di settore (ENEA), individuando, per ciascun dataset:
- un nome,
- una categorizzazione (tra le categorie e sotto-categorie fornite),
- la finestra temporale su cui è aggregato il dato,
- le proprietà che lo compongono.

Questa progettazione viene solitamente fatta utilizzando due strumenti presenti nella sezione Strumenti:
- il Navigatore dell'Ontologia SCPS Web Library, con il quale si possono vedere gli UrbanDataset definiti, organizzati in categorie e sottocategorie;
- la Scheda Proposta UrbanDataset.

Al termine di questo passo, gli UrbanDataset necessari alla comunicazione saranno correttamente certificati e registrati da ENEA nell'Ontologia centrale.

N.B. la inter-SCP dovrà inoltre specificare nel proprio Registry quali UrbanDataset, tra quelli definiti nell'Ontologia, devono essere supportati. Questa configurazione viene svolta da ENEA come utente ADMINISTRATOR della inter-SCP, tramite la SCP-GUI, nella sezione DATASET.

6.4 Passo 3: Esportazione degli UrbanDataset

Una volta che gli UrbanDataset necessari alla comunicazione in oggetto, tra SCP e iSCP, sono correttamente definiti nell'Ontologia centrale (si veda Passo 2), sarà possibile effettuare il download dei relativi template in formato JSON o XML.

N.B. Il prototipo SCP di ENEA supporta, al momento, solo il formato JSON.

Per effettuare il download è possibile utilizzare il il Navigatore dell'Ontologia (SCPS Web Library), dopo aver individuato l'UrbanDataset desiderato, tramite i link "JSON Message Template" o "XML Message Template".

La struttura sintattica degli UrbanDataset, comune a entrambi i formati JSON e XML, è descritta nel dettaglio tramite il "Modello Astratto dei Dati" che fa parte della specifica SCPS Information 2.0.

I template JSON o XML degli UrbanDataset saranno utilizzati per rappresentare i dati nello scambio dati. La SCP, quindi, nell'ottica di abilitare la comunicazione con la inter-SCP, dovrà implementare l'esportazione dei

dati dal proprio sistema locale nel formato UrbanDataset, di cui possiede il template, e verificare che tale file sia conforme alle specifiche superando i test di validazione (Schema e Schematron).

N.B. il campo dell'UrbanDataset UrbanDataset/context/producer, che deve essere aggiornato nel template, verrà spiegato nel Passo 4.

6.5 Passo 4: Configurazione della Collaborazione

A questo punto, è necessario configurare la collaborazione tra la SCP e la inter-SCP (iSCP) secondo la specifica SCPS Collaboration 2.0.

Vi sono due configurazioni, relative a due collaborazioni, da effettuare:
- una lato SCP (Passo 4.1) e
- una lato inter-SCP (Passo 4.2).

6.5.1 Passo 4.1: Configurazione della Collaborazione lato SCP

Questa configurazione viene eseguita per permettere al modulo di esportazione dell'UrbanDataset (descritto al Passo 3), di essere identificato dalla SCP in oggetto come SERVICE (equiparabile a un utente di tipo SOLUTION) e così di poter pubblicare l'UD esportato sulla SCP, rendendolo disponibile al recupero da parte della inter-SCP.
Questa azione è fatta dall'utente ADMINISTRATOR della SCP tramite la SCP-GUI.
La configurazione della collaborazione implica la pubblicazione della risorsa con un resource_id univoco a cui la inter-SCP deve avere accesso.
Descriviamo sinteticamente in cosa consiste la configurazioe di questa collaborazione nel prototipo SCP (in modo tale possa essere da esempio per piattaforme sviluppate da terze parti):

a. la registrazione del SERVICE presso la SCP,
   N.B. ciò genera il *solution_id* da inserire negli UD nel campo *UrbanDataset/context/producer*, quando il SERVICE produce UD (nello stesso modo per le Solution verticali della SCP);
b. la specificazione dei permessi in produzione presso la SCP, esprimendo l'associazione "SERVICE – UrbanDataset", ovvero quali UrbanDataset saranno generati e pubblicati sulla SCP dal modulo SERVICE di esportazione e in quale modo (protocollo, frequenza di invio, storage, ecc.)
c. si noti che nella configurazione dell'associazione "SERVICE – UrbanDataset" (in produzione) viene automaticamente associato un resource_id, identificatore dell'UD prodotto dallo specifico SERVICE (o SOLUTIOIN), in uno specifico periodo temporale, per la specifica SCP (seguendo la convenzione sintattica descritta nella specifica SCPS Collaboration 2.0; tale resource_id viene utilizzato sia per effettuare la push dell'UD dal modulo SERVICE sul sistema locale (si veda Passo 5.1), sia per fornire il conseguente accesso alla inter-SCP sulla specifica risorsa (si veda il punto seguente) e sia per configurare la comunicazione sulla inter-SCP (si veda PASSI 4.2 e 5.2);
d. per permettere alla inter-SCP di accedere all'UD pubblicato, bisognerà definire l'accesso relativo (salvo l'UD non sia dichiarato OPENDATA); si noti che, in questo caso, la iSCP viene vista dalla SCP come una normale Solution/Piattaforma verticale, basterà quindi definire un'associazione "SOLUTION – UrbanDataset" (in accesso), ovvero quale UrbanDataset potrà essere acceduto dalla iSCP (che deve risultare registrata presso la SCP), e con quale protocollo.

A questo punto il SERVICE di esportazione è abilitato a pubblicare l'UrbanDataset che la iSCP potrà recuperare.

6.5.2 Passo 4.2: Configurazione della Collaborazione lato inter-SCP

Questa configurazione viene eseguita per permettere alla inter-SCP (iSCP) di recuperare l'UrbanDataset esportato e pubblicato dalla SCP (si vedano PASSI 3 e 4.1).
Questa azione è fatta dall'utente ADMINISTRATOR della inter-SCP (ovvero il responsabile di ENEA) tramite la SCP-GUI della inter-SCP.

La configurazione della collaborazione lato inter-SCP implica:

a. registrazione della SCP (vista come Solution/Piattaforma verticale) presso la iSCP,
   N.B. ciò permette di generare il *solution_id* da inserire negli UrbanDataset nel campo *UrbanDataset/context/producer*, quando la SCP produce UD per la iSCP;
b. specificazione presso la iSCP delle associazioni "SCP – UrbanDataset" (in produzione), ovvero quali UrbanDataset saranno generati dalla SCP (vista come Solution/Piattaforma verticale) alla iSCP e con quali parametri (protocollo, endpoint del web service, resource_id remoto (configurato al Passo 4.1), frequenza di produzione, modalità di storage, ecc.);
c. si noti che anche alla configurazione dell'associazione "SCP – UrbanDataset" (in produzione, verso la iSCP) viene automaticamente associato un resource_id, identificatore dell'UD prodotto da una specifica SCP in uno specifico periodo temporale, verso la inter-SCP; tale *resource_id* sarà quello utilizzato nella comunicazione (descritta al Passo 5).

N.B. la presenza di un diversi *resource_id* sembrerebbe una complessità aggiuntiva non necessaria. In realtà i due resource_id, anche se afferiscono idealmente agli stessi UrbanDataset, sono in realtà ben distinti:

- il *resource_id* descritto nel Passo 4.1 è l'identificatore dell'Urban_Dataset esportato e immagazzinato nella SCP in oggetto;
- il *resource_id* descritto nel Passo 4.2 è l'identificatore dell'Urban_Dataset recuperato e immagazzinato nella inter-SCP.

6.6 Passo 5: Implementazione della Comunicazione

Il passo finale è l'implementazione della comunicazione tra SCP in oggetto e la inter-SCP.

Vi sono due comunicazioni da implementare:
- una relativa al flusso dati "SERVICE - SCP" (Passo 5.1) e
- una relativa al flusso dati "SCP - iSCP" (Passo 5.2).

6.6.1 Passo 5.1: Comunicazione SERVICE - SCP

La SCP, dopo aver esportato i dati dal proprio sistema locale e averli rappresentati con il formato UrbanDataset (si veda Passo 3), e dopo aver preparato la configurazione della produzione da parte del modulo SERVICE (si veda Passo 4.1), dovrà pubblicare tale dati tramite il web service *UrbanDatasetGateway* sulla SCP di riferimento per renderli disponibili alla inter-SCP.
Il modulo SERVICE effettuerà dunque una chiamata al metodo *UrbanDatasetGateway.push* della SCP fornendo in input l'UrbanDataset esportato e il resource_id configurato appositamente (Passo 4.1).
L'*UrbanDatasetGateway* è un web service RESTful la cui interfaccia è descritta dettagliatamente nella specifica SCPS Communication 2.0.
Per richiamare tale web service la solution dovrà implementare un client implementando i metodi che dovrà utilizzare. Una volta implementato il client web service, nel caso l'invio o il recupero di UD sia configurato come periodico, la Solution dovrà automatizzazione il processo secondo la frequenza stabilita e concordata.
A questo punto l'UD è pubblicato e pronto per essere recuperato dalla iSCP.

6.6.2 Passo 5.2: Comunicazione SCP - iSCP

Il passo finale è l'implementazione della comunicazione tra SCP e inter-SCP.
Come abbiamo descritto, a questo punto la SCP esporta e pubblica l'UD concordato periodicamente (si veda Passo 5.1). A questo punto la iSCP, seguendo la configurazione della collaborazione preparata in 4.2, può prelevare l'UD periodicamente, tramite una chiamata presso la SCP al metodo L'UrbanDatasetGateway.lastRequest che restituisce l'ultimo UD prodotto per un particolare resource_id. La inter-SCP, a questo punto, automatizza il processo secondo la frequenza stabilita e concordata.

N.B. uno dei parametri richiesti è il *resource_id* configurato al Passo 4.2, che identifica univocamente l'UrbanDataset da recuperare. L'*UrbanDatasetGateway* è un web service RESTful la cui interfaccia è descritta dettagliatamente nella specifica *SCPS Communication 2.0*.

6.7 Conclusione Percorso di Adesione

In conclusione, una volta ultimati i 5 passi, il referente della SCP si impegna a:

1. monitorare periodicamente, tramite l'interfaccia grafica della SCP-GUI (nel caso del prototipo ENEA), la corretta produzione dell'UD da parte dei SERVICE interni e il corretto accesso agli UD da parte della inter-SCP;
2. qualora dal monitoraggio dovessero risultare delle anomalie, provvedere a ripristinare il meccanismo di produzione dell'UD da parte del SERVICE, secondo la periodicità stabilita;
3. mantenere funzionante il meccanismo di comunicazione SCP-iSCP per il tempo concordato (stabilito nell'accordo formale tra le parti);
4. effettuare eventuali aggiornamenti necessari al funzionamento del sistema, durante il periodo di collaborazione concordato.

Per ricevere assistenza sui percorsi di adesione, è possibile contattare via email il personale ENEA (dal menù principale, [AIUTO] > [FEEDBACK]).

7 Conclusioni

In questo rapporto tecnico è stata descritta la linea di attività LA 1.20 "*Sviluppo Piattaforma per la Governance dei Dati Urbani Energetici*" in cui si è svolta la progettazione e avviato lo sviluppo della piattaforma *inter-SmartCityPlatform* agente su scala nazionale.

I risultati più immediatamente tangibili sono **la definizione del caso studio pilota** e la prima versione di piattaforma **inter-SCP versione 0.1** completa di tutte le configurazioni descritte e una **iSCP-Dashboard** pronta per essere configurata per la visualizzazione dei dati.

Come abbiamo ribadito più volte, questa linea di attività (LA 1.20) si basa sul lavoro svolto nella precedente annualità (LA 1.19) che è stata di analisi, studio di fattibilità e studio dello stato dell'arte, e prepara la strada per l'ultima attività del triennio (LA 1.21) in cui si ultimerà lo sviluppo e si avvierà la sperimentazione finale.

Un'attività essenziale per implementare il caso studio pilota e avviare la sperimentazione sarà lo sviluppo dello SCP-Scheduler, le cui funzionalità, proprio grazie alla definizione del caso studio pilota, sono già state individuate.

Le uniche criticità riscontrate nella LA 1.20 sono state dovute a ritardi nella consegna dei risultati da parte di collaboratori esterni, ritardi dovuti essenzialmente all'emergenza pandemica. Ad ogni modo, sebbene con alcuni mesi di ritardo, possiamo comunque affermare che il lavoro è stato svolto secondo quanto programmato, raggiungendo tutti i risultati pianificati, in particolare la definizione del "caso studio pilota" è stata chiarita in ogni sua parte divenendo descrizione fondamentale per guidare lo sviluppo software.

L'obiettivo primario della prossima linea di attività LA 1.21 (terza e ultima del triennio, per l'annualità 2021) è l'implementazione completa del caso studio pilota, con i componenti SCP-Scheduler e SCP-Dash funzionanti, e con le diverse fasi del caso studio pilota (definizione UD, esportazione UD, trasmissione UD, DataFusion di UD, Visualizzazione Finale UD, descritte in questo report) automatizzate e in produzione su SCP-Casaccia e inter-SCP, funzionanti online su server ENEA. **L'implementazione del caso studio pilota è l'obiettivo minimo a cui tendere nella sperimentazione LA 1.21**.
Con questo non si vuole dire che lo sforzo richiesto relativo sia minimo, anzi, lo sviluppo dello SCP-Scheduler e dei diversi moduli SERVICE, nonché la configurazione dei report della dashboard sarà decisamente consistente e per questo, al momento, si ritiene di non poter fare di più nel tempo disponibile.
Ulteriori obiettivi, sviluppi futuri che NON sono stati pianificati in fase di stesura di questo progetto triennale ma che si vogliono comunque specificare per far meglio comprendere la direzione che l'intero progetto del framework SCP intende intraprendere, potrebbero essere i seguenti:

1. connessione di altre SCP alla inter-SCP: il meccanismo per la connessione è stato ideato, ovviamente, per essere replicato a tutte le altre SCP e piattaforme urbane che vogliono connettersi alla inter-SCP tramite specifiche SCPS; questo permetterà di verificare sulla inter-SCP la gestione di più contesti reali e, sulla iSCP-Dashboard associata alla inter-SCP, di abilitare servizi di confronto tra le varie città monitorate;
2. utilizzo degli UrbanDataset descriventi i KPI su scala nazionale (si veda par.3.2.3 e lavoro svolto dal Politecnico di Milano descritto nel report LA 1.24 [20]) per inviare dati urbani relativi alla gestione energetica, dalle SCP urbane alla inter-SCP; in questo modo potremo ottenere una vera e propria governance dei dati urbani, relativi alla gestione energetica, su scala nazionale.
3. realizzazione del caso studio "urbano" (descritto nel par.2.7) che permetterebbe di consolidare i componenti sviluppati e potenziare il prototipo SCP-Casaccia, in vista anche di replicare tali funzionalità su altre SCP urbane e così dimostrare, in modo meno tecnico e più diretto all'utente finale, l'efficacia del recupero di dati armonizzati in una piattaforma centrale, per mezzo di formati e protocolli condivisi, per effettuare un monitoraggio in tempo reale della della Smart City.

A quel punto, se il framework fosse composto da una rete minima di SCP in grado di abilitare servizi di benchmarking tra diverse città, si potrebbe ampliare ulteriormente e le esperienze ricavate nei diversi contesti reali permetterebbero di consolidare e migliorare l'approccio con l'obiettivo di una diffusione sul territorio nazionale.

Riferimenti bibliografici

[1] SCP Project website
https://smartcityplatform.enea.it/

[2] Smart City Platform Specification for interoperability layer, SCPS 2.0
https://smartcityplatform.enea.it/#/it/specification/

[3] C. Novelli, A. Frascella, Settembre 2020, "SCPS Functional 2.0"
https://smartcityplatform.enea.it/#/it/specification/functional/2.0/

[4] SCPS Ontology Web Tool
https://smartcityplatform.enea.it/SCPSWebLibrary/ontologyinfo

[5] A. Brutti, A. Frascella, N. Gessa, C. Novelli, Novembre 2020, "SCPS Information 2.0"
https://smartcityplatform.enea.it/#/it/specification/information/2.0/

[6] C. Novelli, Settembre 2020, "SCPS Collaboration 2.0"
https://smartcityplatform.enea.it/#/it/specification/collaboration/2.0/

[7] C. Novelli, A. Frascella, G. Santomauro, Settembre 2020, "SCPS Communication 2.0"
https://smartcityplatform.enea.it/#/it/specification/communication/2.0/

[8] RdS 2015-2017, Report RdS/PAR2018/020, RdS/PAR2017/040 e SCPS in RdS/PAR2017/103 fino a 108

[9] Specifiche SCPS 1.0, Dicembre 2018
https://smartcityplatform.enea.it/#/it/specification/1.0/

[10] Smart City Platform (SCP) Smart Village Casaccia
https://smartcityplatform.enea.it/casaccia/

[11] inter - Smart City Platform (iSCP) versione 0.1
https://smartcityplatform.enea.it/inter-scp/

[12] C. Novelli, A. Frascella, Settembre 2020, "SCPS Glossario 2.0"
https://smartcityplatform.enea.it/#/it/specification/glossary.html

[13] Definizione dell'UrbanDataset "*Platform Status*" nell'Ontologia 1.0
https://smartcityplatform.enea.it/specification/semantic/1.0/ontology/scps-ontology-1.0.owl#PlatformStatus

[14] Report RdS/PTR(2019)/007, capitolo 7, "Stato dell'Arte Servizi di Visualizzazione Dati"

[15] inter - Smart City Platform Dashboard (iSCP-Dashboard)
https://smartcityplatform.enea.it/inter-scp/dash/

[16] SCPS WebLibrary, applicazione web per la navigazione dell'Ontologia SCPS
https://smartcityplatform.enea.it/SCPSWebLibrary/ontologyinfo

[17] Blaso L., Brutti A., Caldera M., Clemente P., Fumagalli S., Giovinazzi S., Giuliani G., Gozo N., Leonardi G., Moretti F., Pizzuti S., Pollino M., Rosato V., Sylos Labini S., Zinzi M., Report RdS/PTR(2020)/016, "Piattaforma PELL: dati statici e dinamici PELL IP e dati statici PELL Edifici Scuole"

[18] C. Novelli, Sezione "Adesione" nello "SCP Project" website
https://smartcityplatform.enea.it/join/

[19] Milano M., Chesani F., Mello P., Patella M., Università di Bologna,
Linea di Attività 1.22, "Ontologia per la Governance dei Dati Urbani Energetici"

[20] Franzò S., Chiaroni D., Chiesa V., Frattini F., Politecnico di Milano,
Linea di Attività 1.24, "Inquadramento europeo dei KPI per dati urbani energetici su scala nazionale"

Appendice A – Replicabilità SCP/iSCP

In quest'appendice verranno descritti i vari passaggi che hanno portato, partendo dalla configurazione della piattaforma "Smart City Platform Smart Village Casaccia" (brevemente SCP-Casaccia), dapprima alla creazione di una piattaforma di riferimento, detta SCP-Template, e poi da essa alla generazione di ulteriori piattaforme (SCP-Bologna, SCP-PELL, SCP-DARE, SCP-COGITO, SCP-ReggioEmilia, SCP-Livorno e la stessa inter-SmartCityPlatform estesa) che condividono con la SCP-Template la stessa matrice di sviluppo.
L'ottimizzazione del processo di duplicazione della piattaforma SCP-Template, al fine di creare nuove istanze che siano facilmente riconfigurabili per diversi casi d'uso, e che siano agevolmente integrabili in nuovi ambienti candidati ad ospitarle, rientra in una dei sotto-task previsti dalla Linea di Attività per l'anno 2020, che nella fattispecie consiste nell'*"installare la piattaforma ICT, consolidarla e renderla maggiormente replicabile"*.

A.1 Contesto di Riferimento

Nella seguente figura schematica è illustrata l'architettura della SCP-Casaccia e il suo sviluppo rappresenta l'obiettivo finale raggiunto nel precedente triennio di progetto. Partendo dalla SCP-Casaccia, che fisicamente afferisce all'infrastruttura cloud del Data Center ENEA di Portici, sono state innanzitutto convogliate le sei macchine virtuali (Virtual Machine o brevemente VM) che compongono questa SCP in un'unica VM, la SCP-Template appunto, che afferisce al Data Center ENEA di Bologna. Questo passaggio è stato possibile perché la progettazione e lo sviluppo della SCP-Casaccia, accompagnati da una fase di raffinamento delle parti, si sono basati sull'idea di rendere modulari e indipendenti le sue diverse componenti. Una o più di una di esse, insistono infatti su una singola VM (ma a breve spiegheremo che non è requisito vincolante) e svolgono un compito preciso.

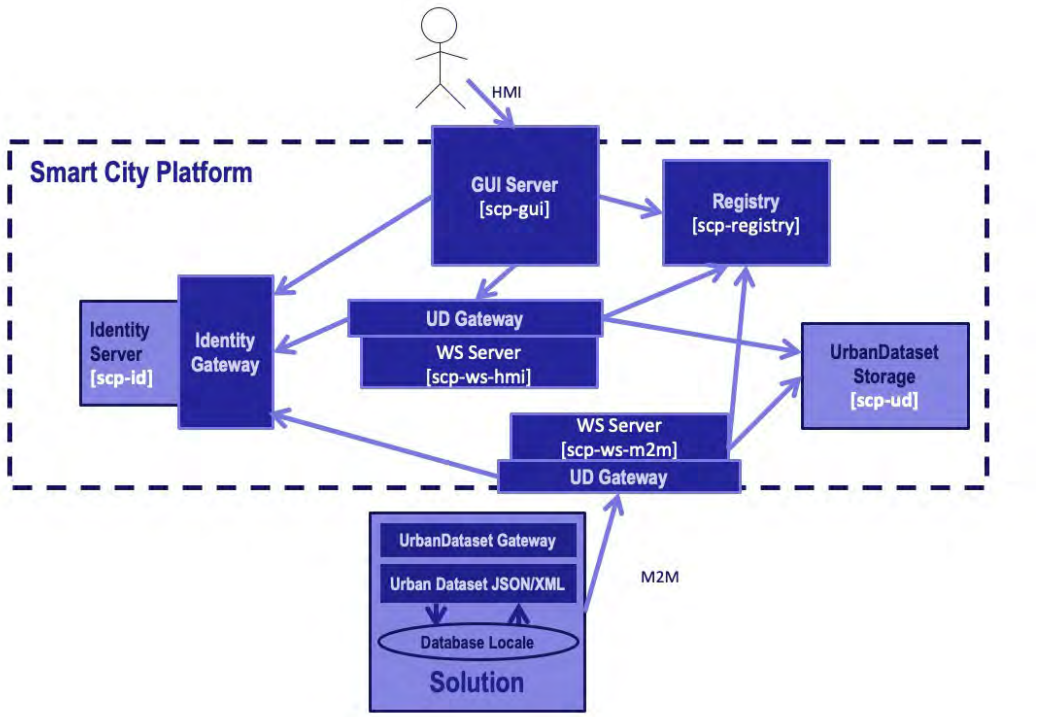

Figura 29 - Architettura Software SCP-Casaccia.

Per meglio illustrare il processo replicazione nelle prossime sezioni, riportiamo qui di seguito la descrizione di ognuna delle sei VM e il ruolo che svolge all'interno della SCP:

- la *scp-ws-m2m*, che in Figura 29 è rappresentata dall'omonima etichetta, è la VM che ospita fisicamente la componente server *UrbanDatasetGateway* che gestisce la ricezione e l'invio di dati

(UrbanDataset o brevemente UD) con le *Solution* (produttori o consumatori di dati) attraverso un canale automatizzato (Machine-To-Machine o più brevemente *M2M*);

- la *scp-ws-hmi*, in maniera analoga alla precedente, gestisce il canale Human-To-Machine-Interface (brevemente *HMI*) per l'interazione utente-piattaforma;
- la *scp-registry* contiene il *Registry*, ossia il database relazione, basato su *MySql,* che controlla le produzioni e gli accessi ai dati;
- la *scp-gui* espone la dashboard (GUI) su un opportuno sito web permettendo tramite web browser di gestire e visualizzare le produzioni o gli accessi ai dati, sia a utenti loggati (in funzione del ruolo che ricoprono) sia a liberi cittadini (nel caso di open data);
- la *scp-ud* ospita il database non relazionale (*Elasticsearch*) per l'archiviazione degli UrbanDataset;
- la *scp-id* integra sia l'Identity Server (*wso2is*) che la componente server *IdentityGateway* che permette l'interfacciamento tra il gestore delle autenticazioni *wso2is* e il resto delle componenti della SCP.

A.2 Preparazione della Macchina Virtuale

Una volta consolidata la SCP-Casaccia, il processo di replicazione delle sei macchine virtuali che compongono questa piattaforma nella singola VM che costituisce la SCP-Template si è tradotto nel preparare una nuova VM equipaggiata con tutto lo stack software della piattaforma originaria.
A tal fine, l'idea iniziale di progettare e sviluppare le diverse componenti della SCP-Casaccia, in forma logica come blocchi funzionali separati e, in forma fisica come oggetti software distinti, dove nella fattispecie ognuno di essi adempie al proprio compito fornendo uno o più servizi come web server, è tornata molto utile nella fase di replicazione.
Per meglio chiarire questo aspetto, riportiamo dapprima per ognuna delle componenti una breve descrizione del relativo servizio web, sottolineando le porte di comunicazione sulle quali insistono i servizi stessi:

- ***UrbanDatasetGateway***: è la componente il cui codice sorgente è sviluppato in *JAVA,* la sua applicazione avviene generando una webapp che si esegue sotto un web server *Apache Tomcat* ed espone dei servizi RESTful per la ricezione e l'invio di UrbanDataset sulle porte 8080 (per il protocollo http) e 8443 (per il protocollo sicuro https);

- ***IdentityGateway***: ha il codice sviluppato in *JAVA* dal quale si genera una webapp che viene eseguita sotto web server *Apache Tomcat.* Essa espone dei servizi RESTful, per la verifica e la gestione delle identità, interfacciandosi con l'Identity Server, sulle porte 8081 (http) e 8444 (https);

- ***GUI***: ha il codice sviluppato in *JAVA* dal quale si genera una webapp che viene eseguita sotto web server *Apache Tomcat*. Essa pubblica un sito web, utile nell'interazione *HMI*, sotto forma di cruscotto per la Graphical Users Interface (GUI) sulle porte 8082 (http) e 8445 (https);

- ***Registry***: è uno schema del DataBase (DB) relazionale *MySql*. Per eseguire l'accesso in lettura e scrittura a questo DB, si utilizzano le interrogazioni sulla porta standard di installazione di *MySql*, 3306 (http);

- ***Identity Server***: è la componente che gestisce le autenticazioni e basata sull'utilizzo del software *wso2is*. Essa espone dei servizi RESTful sulla porta 9443 (https);

- ***UrbanDataset DataBase***: è la componente che gestisce la memorizzazione degli UrbanDataset ed è basata sull'utilizzo del DB non relazionale *Elasticsearch*. Essa espone dei servizi RESTful sulla porta 9200 (http).

- ***MQTTCheck***: è la componente il cui codice sorgente è sviluppato in *JAVA,* la sua applicazione avviene generando una webapp che si esegue sotto un web server *Apache Tomcat* ed espone dei servizi RESTful che si interfacciano con un broker MQTT sulle porte 8083 (http) e 8446 (https);

- ***Dashboard***: è la componente il cui codice sorgente è sviluppato in *JAVA,* la sua applicazione avviene generando una webapp che si esegue sotto un web server *Apache Tomcat* ed espone dei servizi RESTful per visualizzare grafici e tabelle eseguendo data fusion degli UrbanDataset sulle porte 8087 (http) e 8447 (https);

Se si osservano le porte sulle quali ognuno dei servizi web viene esposto, si nota che sono tutte differenti tra loro. Tale circostanza ha permesso di poter installare tutte le componenti su una singola macchina virtuale assicurando la piena funzionalità in assenza di conflitti per servizi che insistono sulla stessa porta. In particolare, per i servizi che vengono eseguiti come webapp sotto web server *Apache Tomcat* verrà chiarito meglio nella sezione A.4 l'organizzazione e la configurazione di questo software.
Per la preparazione della SCP-Template si è trattato dunque di definire una MV sul data center di Bologna con lo stesso sistema operativo (*Ubuntu 16.04 LTS*) delle 6 macchine di Portici e su di esso copiare tutte le componenti. In particolare, nella directory di sistema /opt sono stati copiati:

- il web server *Apache Tomcat* nel quale sono confluite le componenti *UrbanDatasetGateway*, *IdentityGateway* e *GUI;*

- il web server di *Elasticsearch* per la componente *UrbanDataset DataBase*;

- il web server *wso2is* per la componente di identity server.

Per la componente *Registry* si è configurato il DB relazionale *MySql* come servizio di sistema sul quale è stato eseguito un dump dello schema *Registry* presente sulla SCP-Casaccia.
Nella directory /opt sono presenti anche degli script per caricare l'ambiente e avviare alcune delle componenti descritte sopra. Questi script verranno illustrati meglio nella sezione A.3. A titolo di esempio riportiamo il listato delle sottocartelle e dei file presenti nella directory /opt:

```
[USER]@scp-template:/opt$ ls -l --group-directories-first
total 36
drwxr-xr-x 3 root     root     4096 Oct 14  2019 apache-tomcat_server
drwxr-xr-x 5 [USER]   [USER]   4096 Nov 13  2019 elasticsearch_server
drwxr-xr-x 3 root     root     4096 Sep 23  2019 wso2is_server
-rwxr-xr-x 1 root     root       99 Sep 23  2019 setjava8.sh
-rwxr-xr-x 1 root     root      603 Mar 20  2020 shutdown-scp.sh
-rwxr-xr-x 1 root     root      185 Mar 18  2020 shutdown-scp-tomcat.sh
-rwxr-xr-x 1 root     root      711 Mar 20  2020 startup-scp.sh
-rwxr-xr-x 1 root     root      184 Mar 18  2020 startup-scp-tomcat.sh
```

Le tre componenti presenti nel server *Apache Tomcat* hanno bisogno inoltre di essere configurate in funzione della Smart City Platform alla quale appartengono. Per tale motivo, tutte quelle informazioni che variano da piattaforma a piattaforma (per esempio, l'hostname e la porta dell'identity server che *l'IdentityGateway* deve interrogare per verificare le credenziali di un utente) sono state parametrizzate e vengono lette dalle varie componenti su alcuni file di testo esterni alle componenti stessi e salvate nella directory /etc/scp (accessibile solo come root). A titolo di esempio riportiamo il listato di alcuni file di proprietà nella directory /etc/scp/:

```
[USER]@scp-template:/etc/scp$ ls -l --group-directories-first
total 72
drwxr-x--- 2 root root 4096 Mar 16 09:58 logs
-rwxr-xr-x 1 root root  298 Feb 24  2020 help.properties
-rwxr-xr-x 1 root root  459 Jun 18  2020 hmi_client.properties
-rwxr-xr-x 1 root root  744 Apr 29  2020 log4j.properties
-rwxr-xr-x 1 root root  407 Apr 29  2020 m2m_client.properties
-rwxr-xr-x 1 root root  453 Jun 18  2020 scp-gui_idg-client.properties
-rwxr-xr-x 1 root root  955 Sep 22  2020 scp-gui.properties
```

-rwxr-xr-x 1 root root 407 Sep 22 2020 **scp-gui_udg-client.properties**
-rwxr-xr-x 1 root root 974 Apr 27 2020 **scp-id.properties**
-rwxr-xr-x 1 root root 547 Apr 28 2020 **scp-registry.properties**
-rwxr-xr-x 1 root root 1165 Sep 17 2020 **scp-ws.properties**

Di seguito, un esempio del contenuto del file di proprietà per la componente *l'IdentityGateway*.

[USER]@scp-template:**/etc/scp**$ cat scp-id.properties
Copyright 2019 ENEA
This file is part of Smart City Platform
by Cristiano Novelli, SCC Group, ENEA Casaccia, Roma
####################
SCP Configuration
####################
SCP [USER] User
scp.[USER]=aaa.bbb@ccc.domain
################################
IdentityServer Configuration
################################
scpid.base.url=https://scp-template.bologna.enea.it
scpid.base.port=9443
Common WSO2 Admin Credentials
wso2.admin.credentials=XXX
HMI WSO2 Master Credentials
wso2.hmi.master.credentials=YYY
M2M WSO2 Master Credentials
wso2.m2m.master.credentials=ZZZ

A.3 Script di avvio delle VM

Ai fini delle replicabilità e della facilità di configurazione sono stati sviluppati degli script che permettono anzitutto il caricamento opportuno dell'ambiente di lavoro e poi l'avvio dei servizi descritti nella precedente sezione.
In particolare, per quanto riguarda l'ambiente, le uniche librerie da caricare per far funzionare le varie componenti della SCP sono quelle del *JAVA Development Kit* (JDK), nello specifico della versione 8. A tale scopo, lo script *setjava8.sh* in /opt esporta nella variabile d'ambiente $PATH, i riferimenti del JDK. Di seguito le linee di codice dello script:

[USER]@scp-template:**/opt**$ cat setjava8.sh
#!/bin/sh
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/jre
export PATH=$JAVA_HOME/bin:$PATH

L'esecuzione di questo script è invocata all'interno di un altro script, sempre presente nella directory /opt, che è startup-scp.sh. Come è intuitivo capire dal nome stesso del file, questo script avvia tutti i programmi che definiscono le componenti stesse della SCP previo caricamento dell'ambiente. Per meglio chiarire, riportiamo le linee di codice dello script:

[USER]@scp-template:**/opt**$ cat startup-scp.sh
#!/bin/sh
PWD=/opt
APACHE_TOMCAT_SERVER=apache-tomcat_server
APACHE_TOMCAT_VER=apache-tomcat-8.5.15
WSO2IS_SERVER=wso2is_server
WSO2IS_VER=wso2is-5.5.0
ELASTICSEARCH_SERVER=elasticsearch_server
ELASTICSEARCH_VER=elasticsearch-6.4.0
#set java
. $PWD/setjava8.sh
#start tomcat
$PWD/$APACHE_TOMCAT_SERVER/$APACHE_TOMCAT_VER/bin/startup.sh > /dev/null 2>&1 &
#start wso2
$PWD/$WSO2IS_SERVER/$WSO2IS_VER/bin/wso2server.sh -Dsetup > /dev/null 2>&1 &
#start elasticsearch
sudo -u [USER] sh -c 'PWD=/opt; . $PWD/setjava8.sh; ELASTICSEARCH_SERVER=elasticsearch_server; ELASTICSEARCH_VER=elasticsearch-6.4.0; $PWD/$ELASTICSEARCH_SERVER/$ELASTICSEARCH_VER/bin/elasticsearch > /dev/null 2>&1 &'

Come si evince dal codice, viene eseguito innanzitutto lo script setjava8.sh. Dopodiché, una volta settato l'ambiente, si procede nell'ordine all'avvio (come utente root) del web server *Apache Tomcat* e dell'identity server *wso2is* e (come utente [USER]) del db non relazionale Elasticsearch.

Questo script costituisce il motore per avviare l'intero sistema di componenti. Nel caso della SCP-Casaccia, in ogni MV questo script conteneva la sola parte di codice per avviare la singola componente ivi ospitata, mentre, nel caso della SCP-Template, essendo le componenti raccolte in un'unica MV, analogamente si è proceduto a raggruppare in un unico file tutti i comandi di avvio.

Per automatizzare infine l'avvio delle componenti al boot della MV, questo script è stato configurato all'esecuzione automatica attraverso la definizione e l'abilitazione di un servizio di sistema. Questa procedura ha anche il vantaggio di rendere più robusta l'intera SCP in caso di riavvio accidentale del sistema operativo stesso.

A.4 Organizzazione Server Apache Tomcat

Il web server *Apache Tomcat* permette di esporre dei servizi web sui protocolli http e https attraverso delle webapp. Nel caso specifico della SCP-Template, le webapp di riferimento sono relative alle tre componenti *UrbanDatasetGateway*, *IdentityGateway* e *GUI*. Per meglio chiarire la struttura organizzativa del web server riportiamo dapprima il contenuto della directory /opt/apache-tomcat_server/apache-tomcat-8.5.15/

```
[USER]@scp-template:/opt/apache-tomcat_server/apache-tomcat-8.5.15$ ls -l
total 144
drwxrwxr-x 2 root root  4096 Oct 14  2019 bin
drwxrwxr-x 7 root root  4096 Apr 27 17:30 conf
drwxrwxr-x 2 root root  4096 Oct 14  2019 lib
-rwxrwxr-x 1 root root 57092 Oct 14  2019 LICENSE
drwxrwxr-x 2 root root 16384 Apr 27 11:36 logs
-rwxrwxr-x 1 root root  1723 Oct 14  2019 NOTICE
-rwxrwxr-x 1 root root  7064 Oct 14  2019 RELEASE-NOTES
-rwxrwxr-x 1 root root 15946 Oct 14  2019 RUNNING.txt
drwxrwxr-x 2 root root  4096 Oct 14  2019 temp
drwxrwxr-x 7 root root  4096 Oct 14  2019 webapps
drwxr-xr-x 3 root root  4096 Apr 24 09:32 webapps_dash
drwxrwxr-x 3 root root  4096 Apr 23 09:49 webapps_gui
drwxrwxr-x 8 root root  4096 Apr 28  2020 webapps_identitygateway
drwxrwxr-x 3 root root  4096 Dec 17  2019 webapps_mqttcheck
drwxrwxr-x 8 root root  4096 Mar 17 11:30 webapps_urbandatasetgateway
```

Come si nota, per ognuna delle componenti, stata creata una sottodirectory (con prefisso "webapps_" seguito dal nome della componente) contenente il relativo file Web application ARchive (WAR). Per ognuno di questi file, una volta avviato il server Tomcat, viene eseguito il deployment e quindi l'avvio della relativa webapp. L'orchestrazione dell'intero web server, in maniera tale che una webapp sia esposta su una coppia di porte http/https differente da quella di un'altra webapp avviene nel file di configurazione *servex.xml* presente nella directory *conf*. Per brevità non riportiamo il suo contenuto, ma è semplicemente l'opportuna fusione dei file *server.xml* provenienti dai server Apache Tomcat implementati sulle singole macchine virtuali della SCP-Casaccia

A.5 Gestione dei certificati

Come accennato, al fine di rendere più sicuro l'intero sistema dei servizi esposti dalla SCP, si è proceduto, ove necessario o opportuno, all'uso del protocollo HTTPS (HyperText Transfer Protocol over Secure Socket Layer). In altre parole, per le comunicazioni da e verso la SCP sulla rete, si è utilizzato un protocollo di connessione criptata tramite crittografia asimmetrica. A tal, fine, l'integrità dello scambio di dati tra le parti viene garantito attraverso l'uso di certificati digitali installati su ogni singola macchina virtuale della piattaforma. In termini operativi, un certificato è costituito da un insieme di file (bundle) rilasciati da un soggetto terzo di fiducia (Certificate Authority, brevemente anche CA) che viene salvato su un server e che attestano l'identità del proprietario. Seguendo le linee guida che definiscono lo standard di installazione di

un certificato, questo bundle, una volta ottenuto da una CA, è salvato nel path /etc/ssl/certs, sotto una directory che prende il nome della macchina server sulla quale è si trova. Ad esempio, sulla SCP-Template, abbiamo la seguente lista di file:

```
[USER]@scp-template:/etc/ssl/certs$ ls -l scp-template_bologna_enea_it_xxxxxxxx/
total 48
-rw-r--r-- 1 [USER] [USER] 1818 Apr  8  2020 DigiCertCA.crt
-rw-r--r-- 1 [USER] [USER] 1202 Apr  8  2020 INSTALL_INSTRUCTIONS.en.txt
-rw-r--r-- 1 [USER] [USER] 1214 Apr  8  2020 INSTALL_INSTRUCTIONS.es.txt
-rw-r--r-- 1 [USER] [USER] 1349 Apr  8  2020 INSTALL_INSTRUCTIONS.fr.txt
-rw-r--r-- 1 [USER] [USER] 1408 Apr  8  2020 INSTALL_INSTRUCTIONS.it.txt
-rw-r--r-- 1 [USER] [USER] 1241 Apr  8  2020 INSTALL_INSTRUCTIONS.lt.txt
-rw-r--r-- 1 [USER] [USER] 2408 Apr  8  2020 scp-template_bologna_enea_it.crt
-rw-r--r-- 1 [USER] [USER] 1017 Apr  8  2020 scp-template_bologna_enea_it.csr
-rw-r--r-- 1 [USER] [USER] 1704 Apr  8  2020 scp-template_bologna_enea_it.key
-rw-r--r-- 1 [USER] [USER] 2370 Apr  8  2020 scp-template_bologna_enea_it.pem
-rw-r--r-- 1 [USER] [USER] 4637 Apr  8  2020 scp-template_bologna_enea_it.pfx
```

I servizi che sono esposti sulla rete pubblica della SCP, come *l'UrbanDatasetGateway* e la *GUI* utilizzano sempre il protocollo https. Altri servizi, come l'*IdentityGateway*, l'*MQTTCheck* e l'*Identity Server*, sebbene non abbiano diretta connessione con l'esterno della rete della SCP, utilizzano lo stesso il protocollo https e quindi sfruttano lo stesso i certificati installati sulle macchine. In particolare, per quanto riguarda i servizi summenzionati, essendo delle webapp eseguite sotto un server *Apache Tomcat* si è trattato operativamente di configurare all'interno del file *server.xml* un'opportuna di istruzione per caricare il certificato salvato sulla macchina. Nella fattispecie, viene utilizzato il certificato che nella directory sopra descritta è espresso nel formato *.pfx*. Per esempio, nel caso dell'*UrbanDatasetGateway* il connettore (sulla porta 8443) che viene definito nel file *server.xml* per utilizzare l'https è espresso nel seguente modo:

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"
       maxThreads="150" scheme="https" SSLEnabled="true" clientAuth="false" sslProtocol="TLS"
       keystoreFile="/etc/ssl/certs/scp-certificate.pfx"
       keystorePass="xxxxxx"
       keystoreType="PKCS12" />
    <!-- Define an AJP 1.3 Connector on port 8009 -->
<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />
```

Come si nota, la variabile keystoreFile, che carica il file del certificato, non è un collegamento diretto al file *.pfx* che è situato nella stessa cartella, ma punta a un link che è sua volta una copia al suddetto file:

```
root@scp-template:~# ls -l /etc/ssl/certs/scp-certificate.pfx
-rw-r--r-- 1 [USER] [USER] 4637 Apr 14  2020 /etc/ssl/certs/scp-certificate.pfx
```

Questo accorgimento ci ha permesso di tenere separata la definizione del certificato dal suo path, il quale contiene anche il nome della originale della macchina host (ad es. /etc/ssl/certs/scp-template_bologna_enea_it_xxxxxxxx/). In questo modo, è possibile replicare più facilmente la SCP, e in particolare il server *Apache Tomcat*, senza dover apportate ulteriori modifiche all'interno del file *sever.xml*, andando di volta in volta, ad aggiornare ogni occorrenza del path del certificato con quello corretto. Inoltre, questa soluzione ritorna utile anche nel momento in cui un certificato scade e occorre caricarne uno nuovo (con conseguente nuovo path) semplicemente cancellando il file scp-certificate.pfx e ricrearlo come copia dal .pfx del nuovo bundle.
L'utilizzo del certificato in formato .pfx è risultato molto importante per la configurazione della SCP ma non viene direttamente fornito dall'ente certificante, bensì è un file che viene generato dal bundle iniziale attraverso i seguenti comandi da shell:

```
[USER]@scp-template:~$ openssl x509 -in [nomecertificato].crt -out [nomecertificato].pem -outform PEM

[USER]@scp-template:~$ openssl pkcs12 -export -out [nomecertificato].pfx -inkey [nomecertificato].key -in [nomecertificato].crt -certfile
DigiCertCA.crt
```

Questi due comandi, che usano il tool *openssl*, creano nell'ordine prima il certificato in formato .pem e poi da questi il .pfx. Il certificato in formato .pem è a sua volta importante perché viene utilizzato per configurare manualmente attraverso la sua interfaccia l'Identity Server *wso2is.*

A.6 Configurazione dei Database

Nell'ottica di rendere disponibili a nuovi riutilizzi i due DataBase (DB) che costituiscono la SCP - ossia *MySql* e *Elasticsearch* (ES) - sono previste due procedure di inizializzazione. Più in dettaglio, ricordiamo che *MySql* contiene il *Registry*, cioè il sistema nel quale sono definite tutte le produzioni e tutti gli accessi possibili ai dati da parte delle solution su una particolare SCP. Inoltre, esso registra tutte le transazioni – in lettura e scrittura - che vengono effettuate sull'altro database, *Elasticsearch*, cioè il sistema che archivia propriamente i dati UrbanDataset (UD).

A.6.1 Inizializzazione MySql

Per quanto riguarda il primo dei due database, *MySql*, vengono eseguite nell'ordine tre operazioni, le quali, sotto forma di query, modificano altrettante tabelle del *Registry*:

- La prima query agisce sulla tabella *scps-registry.admin_smartcityplatform* e aggiorna i dati della singola nuova istanza di SCP che viene creata. In particolare, da parte di ENEA viene assegnato un ID alla SCP per identificarla univocamente nell'insieme di tutte le altre piattaforme. Ad esempio alla SCP-Template è stato assegnato l'ID "99" e la tabella si presenta nella seguente forma.

```
[USER]@scp-template:~$ mysql -u [user] -p -e 'SELECT * FROM `scps-registry`.admin_smartcityplatform;'
Enter password:
+----+----------+---------+----------+-------+
| id | name     | city    | province | cap   |
+----+----------+---------+----------+-------+
| 99 | Template | Bologna | Bologna  | 40129 |

+----+----------+---------+----------+-------+
```

- La seconda query, che modifica la tabella *scps-registry.communication_endpoint* genera l'endpoint *dell'UrbanDatasetGateway*, cioè l'URL sul quale le solution possono interrogare il web server con le opprtune chiamate di *test, login, isAlive, push, basicRequest, ecc.* Ad esempio sulla SCP-Template il campo *endpoint* della suddetta tabella è:

```
[USER]@ scp-template:~$ mysql -u [user]-p -e 'SELECT * FROM `scps-registry`.communication_endpoint;'
Enter password:
+----+-----------------------------------------------------------------------------------+----------+----------+-------+
| id | endpoint                                                                          | username | password | extra |
+----+-----------------------------------------------------------------------------------+----------+----------+-------+
|  1 | https://scp-template.bologna.enea.it:8443/webservices/rest/UrbanDatasetGateway/ | NULL     | NULL     | NULL  |
+----+-----------------------------------------------------------------------------------+----------+----------+-------+
```

Come si evince, la stringa che costituisce questo endpoint è l'unione di diverse parti. Alcune sono statiche, altre invece sono dinamiche e dipendono dalla particolare SCP istanziata. In generale, salvo diverse indicazioni, il protocollo https, la porta 8443 e il path dove è ubicato *l'UrbanDatasetGateway* non variano. L'unica parte dell'URL che viene di volta in volta modificata è il nome della SCP con la quale essa viene registrata sul Domain Name System (DNS), pubblico o locale (a seconda dell'uso), per essere ricercata sulla rete senza passare per l'indirizzo IP che le viene assegnato.

- La terza ed ultima query modifica l'unica *collaboration* esistente su una nuova SCP qualora venga instanziata, ed è riferita all'unica *solution* che resta sempre definita sulla piattaforma, cioè a quella di test e che è detta appunto *TestSolution.* In particolare si aggiorna la prima e unica entry della tabella *scps-registry.collaboration*, modificando i campi *smartcity_id* e *resource_id*, sostituendo le occorrenze dell'ID della SCP-Template (99) con il nuovo valore che viene assegnato alla nuova piattaforma. Ad esempio sulla SCP-Template, la tabella appena menzionata poco sopra ha la seguente struttura:

```
[USER]@scp-template:~$ mysql -u [user]-p -e 'SELECT * FROM `scps-registry`.collaboration;'
```

```
Enter password:
+----+--------------+--------------+------------+--------------------+--------------------+-----------------------------------------------------------+
| id | smartcity_id | dataset_id   | solution_id | collaboration_start | collaboration_end   | resource_id                                               |
+----+--------------+--------------+------------+--------------------+--------------------+-----------------------------------------------------------+
|  1 |           99 | Whatever-1.0 |          1 | 2020-04-09 11:00:00 | 2040-04-09 12:00:00 | SCP-99_TestSolution-1_Whatever-1.0_20200409110000 |
+----+--------------+--------------+------------+--------------------+--------------------+-----------------------------------------------------------+
```

A.6.2 Inizializzazione Elasticsearch

Relativamente al secondo tipo di DB, cioè *Elasticsearch*, sono stati implementati una serie di script shell che nell'ordine: cancellano la collezione esistente di UD, ne ricreano una nuova vuota e ridefiniscono il mapping, ossia lo schema-json che i nuovi dati devono avere quando vengono salvati. Questi script, che si trovano in /opt/elasticsearch_server/script, sono:

```
[USER]@scp-template:/opt/elasticsearch_server/script$ ls -lrt
total 44
-rwxr-xr-x 1 [USER] [USER]  161 Sep 17  2020 create_index.sh
-rwxr-xr-x 1 [USER] [USER]  181 Sep 17  2020 delete_index.sh
-rwxr-xr-x 1 [USER] [USER] 8001 Sep 17  2020 create_index_mapping.sh
```

Ad esempio lo script per creare una nuova collezione di dati, che nella terminologia di *Elasticsearch* viene chiamato indice, si chiama appunto create_index.sh e ha la seguente struttura:

```
[USER]@scp-template:/opt/elasticsearch_server/script$ cat create_index.sh
#!/bin/sh
es_ip="localhost"
es_port="9200"
es_url="http://"$es_ip:$es_port
index="[NOMEINDICE]"
index_url=$es_url/$index
curl -w "\n" -X PUT $index_url?pretty
```

Allo stesso modo, sono definiti gli script delete_index.sh e create_index_mapping.sh, che per brevità non riportiamo.

A.7 Test con client WS

Per verificare la corretta installazione di una nuova SCP viene effettuata una batteria di test che consiste nell'interrogare la SCP attraverso i due canali *M2M* e *HMI*.
Per quanto riguarda il primo canale di comunicazione, *l'M2M, si* utilizza un client che simula le richieste per conto della *TestSolution* ed in particolare si eseguono nell'ordine delle chiamate RESTful ai seguenti servizi:

- *test:* per verifcare che l'*UrbanDatasetGateway* sia attivo;
- *login:* per verificare che credenziali della *TestSolution* siano valide, che l'*IdentityGateway* e l'*Identity Server* siano attivi;
- *isAlive:* per verificare che il token di autenticazione ottenuto dalla *login* per la *TestSolution* sia corretto;
- *push:* per verificare che l'invio di nuovo UD avvenga con successo;
- *specificRequest* e/o *basicRequest:* per verificare se sia possibile recuperare in lettura l'UD appena inserito e quindi che i servizi di *Registry* e *Elasticsearch* risultino funzionanti;

Rispetto al secondo canale di comunicazione, l'*HMI*, i test di funzionalità vengono effettuati sulla GUI tramite web browser. In particolare, ci si autentica tramite le credenziali della *TestSolution* e si eseguono le operazioni di visualizzazione e download dell'UD caricato con il canale *M2M*.

A.8 SCP-Template per SCP-Light e SCP-Full

Una volta consolidata la struttura della SCP-Template, è stato possibile replicare questa Smart City Platform in diverse nuove istanze. Per la creazione fisica di una nuova piattaforma è dunque sufficiente eseguire uno snapshot della MV contenente la SCP-Template e copiarla sul nuovo ambiente virtuale dove si vuole una

replica della piattaforma. L'assegnazione di un hostname e la configurazione di rete della nuova macchina sono a carico del gestore del sistema ospitante.
In modo specifico, dopo che dalle sei macchine virtuali della SCP-Casaccia (Data Center ENEA di Portici) è stata preparata la SCP-Template (Data Center ENEA di Bologna) secondo le operazioni illustrate nelle sezioni precedenti, la procedura di replica ha visto nascere due generi di SCP:

- *SCP-Light*: è un'istanza di Smart City Platform concentrata in una singola macchina virtuale, esattamente come la SCP-Template;
- *SCP-Full*: è un'istanza di Smart City Platform distribuita su un pool di macchine virtuali, generalmente sei, esattamente come la SCP-Casaccia.

Il primo genere di SCP risulta utile in ambiti applicativi dove non è richiesto un esoso contributo dal punto di vista delle prestazioni computazionali, permettendo al tempo stesso un ridotto costo di gestione.
Il secondo genere di SCP è invece utile per garantire la modularità delle componenti permettendo anche in fasi successive all'installazione la possibilità di espansione di singole parti della piattaforma stessa o l'integrazione di nuovi moduli indipendenti.
In Figura 30 è illustrata l'intera catena del processo di replicazione che è stato eseguito. Al tipo *SCP-Full* appartiene la SCP-Bologna, nata appunto per testare la replica multi server. Al tipo *SCP-Light* appartengono invece la SCP-PELL (presso ENEA-Bologna), SCP-DARE (presso CINECA), SCP-COGITO (ENEA-Portici) e SCP-ReggioEmilia (presso ENEA-Bologna), create per altrettanti progetti di ricerca e/o applicazioni sperimentali.
Una menzione particolare è riservata per le due istanze iSCP e SCP-ID-Common. Esse sono entrambe piattaforme del tipo *light* ma con particolari funzioni. La prima delle due, (iSCP) svolge il ruolo di Inter-SCP. In sostanza è la piattaforma nazionale che da un lato opera come una normale SCP che archivia i dati, da un altro si comporta come una Solution che accede a tutte le altre piattaforme interrogandole e raccogliendo i dati ivi contenuti. La seconda piattaforma (SCP-ID-Common) svolge il ruolo di Identity Server globale e verrà meglio chiarito il suo compito e la sua replica nella successiva sezione.

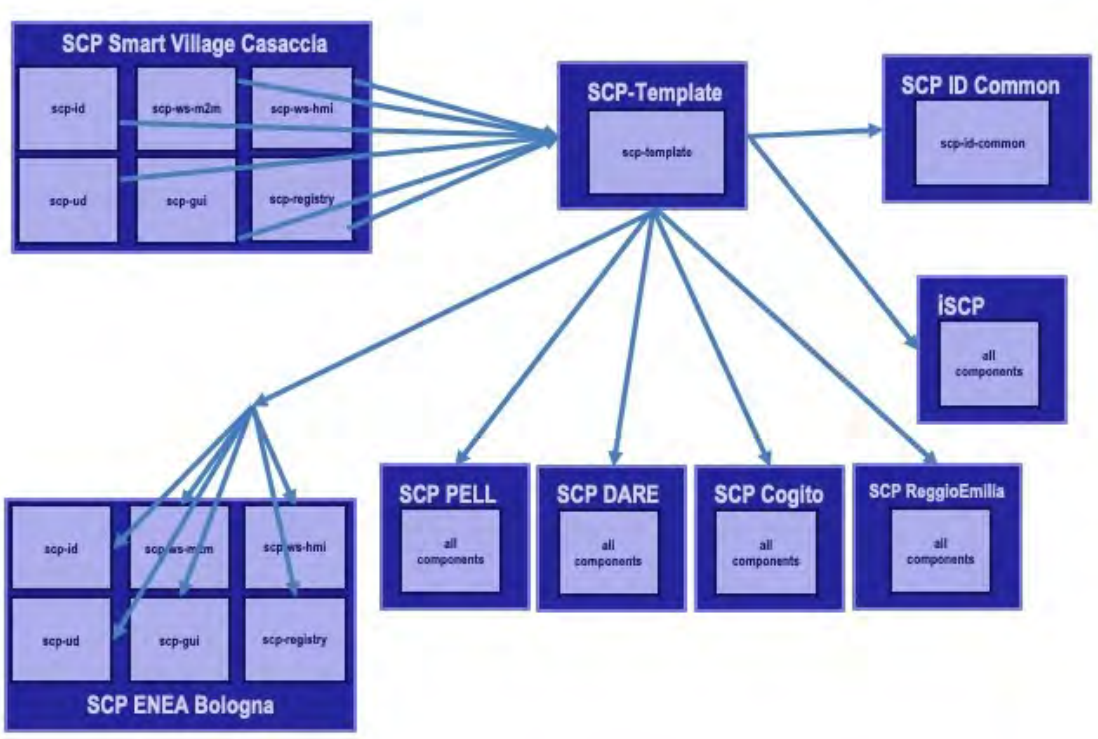

Figura 30 - Processo di Replicazione SCP

A.9 Replica della SCP-ID-Common

La piattaforma SCP-ID-common è stata creata per centralizzare la gestione delle identità di una generico utente registrato presso una o più SCP. In questo modo si permette l'autenticazione di un utente su più piattaforme con una singola coppia di credenziali. In altre parole, l'idea è quella di fornire un sistema di Single Sign On (SSO), snellendo la procedura di registrazione (viene effettuata una volta sola) sull'Identity Server garantendo l'accesso su più SCP. Per il processo di replica è stato sufficiente generare una nuova SCP nella quale sono stati lasciati attivi l'*IdentityGateway,* l'*Identity Server* e il *Registry*. Per agganciare (facoltativamente) le SCP istanziate a questo nuovo servizio di gestione globale delle utenze è bastato spegnere su di esse l'Identity Provider locale e far puntare i servizi come *UrbanDatasetGateway*, *GUI* ecc. all'indirizzo di questa nuova macchina. Per questa operazione è tornato molto utile l'aver parametrizzato le informazioni relative a questi servizi nei file di proprietà descritti nella sezione A.2. Tuttavia, da questa nuovo modo di gestire le identità è nata una criticità che è stata prontamente risolta.
In Figura 31 è rappresentata la situazione di un utente che si registra da due piattaforme distinte, prima da SCP-X e poi da SCP-Y, entrambe allacciate alla SCP-ID-Common. Con la prima richiesta l'utente viene registrato sull'Identity Server della SCP-ID-Common e viene aggiunto un record nel database "*DB Utenti*" dell'Identity Server.

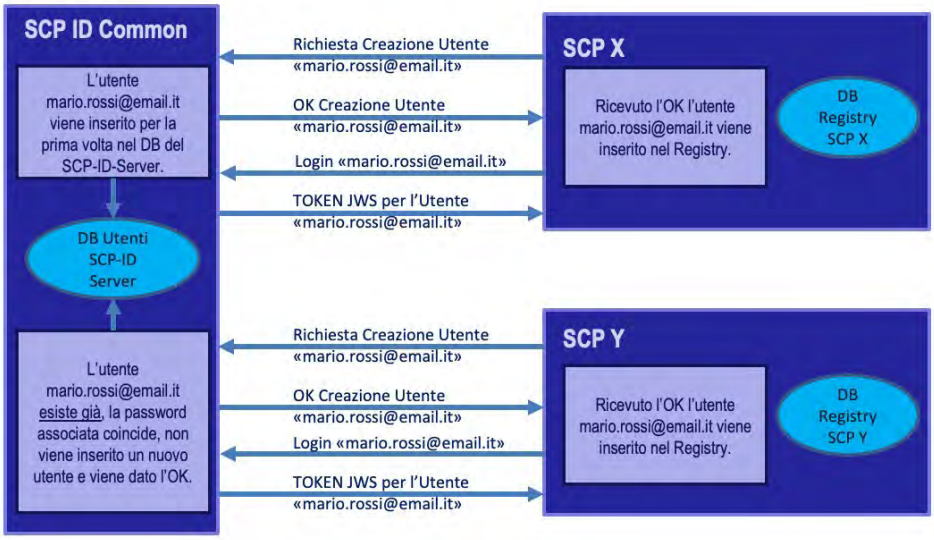

Figura 31 - SCP-ID-Common, illustrazione criticità

Nel secondo tentativo, verificando che l'utente esiste già non viene eseguito un nuovo inserimento. In questo modo i DB *Registry* di entrambe le SCP contengono il nome utente (nel formato email) registrato in esse. Contemporaneamente, il *"DB Utenti"* della SCP-ID-Common contiene gli account di tutti gli utenti (sia di SCP-X che di SCP-Y). Una criticità verrebbe così scaturita nel momento in cui si decide per la cancellazione di un utente da parte di una delle due SCP (per es. SCP-X), perché essa cancellerebbe tale account nel *"DB Utenti"* della SCP-ID-Common, creando una situazione inconsistente nei *Registry* delle altre SCP (per es. SCP-Y), ovvero il nome utente risulterebbe ancora presente ma non più valido.
Per ovviare a questo problema, come illustrato in Figura 32, sul *Registry* della SCP-ID-Common è stata aggiunta una tabella apposita, utilizzata solo su questa piattaforma, che si chiama *scps-registry.user_platforms*. Con questa tabella si tiene traccia degli utenti registrati e anche delle piattaforme su cui essi sono stati registrati. Ad esempio, per la username dell'utente test.solution, la situazione sul *Registry* della della SCP-ID-Common è la seguente:

```
[USER]@scp-id-common:~$ mysql -u [user] -p -e 'SELECT * FROM `scps-registry`.user_platforms;'
Enter password:
+---------------------------------+----------+
| username                        | platform |
+---------------------------------+----------+

    . . .
```

test.solution@enea.it	SCP-1
test.solution@enea.it	SCP-2
test.solution@enea.it	SCP-3
test.solution@enea.it	SCP-5
test.solution@enea.it	SCP-99

. . .
+---------------------------------+----------+

In altre parole, per ogni utente saranno presenti su questa tabella tante username ripetute per altrettante SCP sulle quali questi è stato registrato. Ciò garantisce che l'account verrà effettivamente cancellato dall'Identity Server della SCP-ID-Common solo nel momento in cui quel nome utente non è utilizzato in alcuna piattaforma, e quindi, come nel caso dell'utente test.solution, non ci sarà più alcuna entry nella tabella con il suo nome. Questa modifica al *Registry* ha previsto una leggera modifica anche ai metodi *signUp* e *delete* sull'*IdentityGateway* che eseguono, rispettivamente, la registrazione e la cancellazione di un utente. In particolare, la modifica ha previsto l'aggiunta di un parametro ulteriore in fase di richiesta che comunica l'identificativo della SCP chiamante all'*IdentityGateway*.

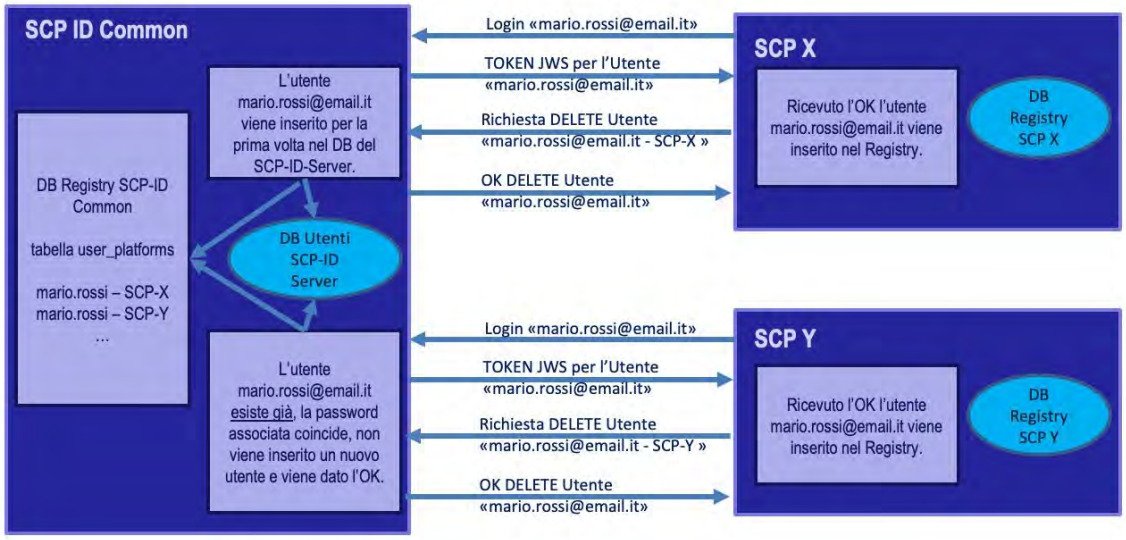

Figura 32 - SCP-ID-Common, risoluzione criticità

Appendice B – Liste di Codici

In questa Appendice vengono riportate alcune liste di codici che sono risultato del lavoro relativo alle nuove definizioni di UrbanDataset (descritto nel par.3.2).

B.1 Tabella codici "LogCode"

La lista di codici a cui è stata vincolata la proprietà "*LogCode*" è stata resa disponibile in formato Genericode ed è la seguente.

Tabella 13 - Tabella di sintesi della lista di codici "LogCode"

Tabella di sintesi della lista di codici "LogCode"	
Nome sintetico	LogCode
Versione	1.0
URI	LogCode.gc
Agency	ENEA
Codice	**Valore**
info	Messaggio informativo
warning	Messaggio di allerta
error	Messaggio di errore

B.2 Tabella codici "PellKPICode"

La lista di codici a cui è stata vincolata la proprietà "*PellKPICode*" è stata resa disponibile in formato Genericode ed è la seguente.

Tabella 14 - Tabella di sintesi della lista di codici "PellKPICode"

Tabella di sintesi della lista di codici "PellKPICode"	
Nome sintetico	PellKPICode
Versione	1.0
URI	PellKPICode.gc
Agency	ENEA
Codice	**Valore**
PowerDensity	Indica se la potenza elettrica installata della sorgente per m2 di superficie è all'interno dell'intervallo ammesso per la classe illuminotecnica, definita dallo standard vigente (UNI 11248), riferita alla zona omogenea considerata. Per zona omogenea si considera un'area che necessita di uguali condizioni luminose per garantire la sicurezza della circolazione veicolare, pedonale e la fruizione degli spazi.
Technological	Indica la qualità dell'efficienza luminosa della sorgente adottata, cioè la quantità di luce prodotta in relazione alla potenza elettrica fornita; questo indicatore consente il confronto della prestazione luminosa della sorgente adottata rispetto al valore minimo previsto per la medesima tecnologia (CAM).
Dimming	Restituisce un'indicazione sui risparmi energetici derivanti dall'utilizzo di strategie di dimming, statiche o adattive, rispetto al caso in cui l'impianto

	preso in considerazione funzioni sempre alla massima potenza, secondo i dati inseriti nel data model PELL IP.
BAT	Restituisce, sulla base del calcolo illuminotecnico relativo alla zona omogenea, un'indicazione circa il vantaggio che si potrebbe ottenere in termini energetici se si utilizzasse una BAT in sostituzione della sorgente installata. La BAT costituisce la migliore tecnologia sul mercato, considerando l'impianto a piena potenza.
BAU	Restituisce, sulla base del calcolo illuminotecnico relativo alla zona omogenea, un'indicazione circa il vantaggio in termini energetici derivanti dall'utilizzo della sorgente installata rispetto all'utilizzo della BAU che costituisce la tecnologia usualmente adottata, affiancata anche dall'utilizzo di sistemi di riduzione del flusso in modalità adattiva.

B.3 Tabella codici “CityPLIndicatorCode”

La lista di codici a cui è stata vincolata la proprietà "*CityPLIndicatorCode*" è stata resa disponibile in formato Genericode ed è la seguente.

Tabella 15 - Tabella di sintesi della lista di codici “CityPLIndicatorCode”

Tabella di sintesi della lista di codici “CityPLIndicatorCode”	
Nome sintetico	CityPLIndicatorCode
Versione	1.0
URI	CityPLIndicatorCode.gc
Agency	ENEA
Codice	**Valore**
PODCount	Numero totale di POD
ElectricPanelCount	Numero totale di Quadri Elettrici
LightSpotCount	Numero totale di Punti Luce
MunicipalLightSpotCount	Numero di Punti Luce di proprietà del Comune
NotMunicipalLightSpotCount	Numero di Punti Luce non di proprietà del Comune
NotMonitorableLightSpotCount	Numero di Punti Luce promiscui, ovvero non monitorabili
LightSpotPerCapitaCount	Numero di Punti Luce per abitante (calcolato sul numero totale di Punti Luce)
NextGenerationLightSpotCount	Numero di apparecchi co s truiti da 2013 in poi
MediumVoltageSystemPercentage	Percentuale di Quadri Elettrici in media tensione (calcolata sul numero totale di Quadri Elettrici)
SPDCount	Numero di Quadri Elettrici con SPD (calcolato sul numero totale di Quadri Elettrici)
GroundingSystemPercentage	Percentuale di Quadri Elettrici con impianto a terra (calcolata sul numero totale di Quadri Elettrici)
TotalNominalPower	Somma delle potenze nominali delle singole Sorgenti Luminose
AverageNominalPower	Potenza nominale media (calcolata sulla somma delle potenze delle singole Sorgenti Luminose)

Tabella di sintesi della lista di codici "CityPLIndicatorCode"

MercuryVaporLampPercentage	Percentuale di Sorgenti Luminose a Vapori di mercurio (calcolata sul numero totale di Sorgenti Luminose)
IncandescentLampPercentage	Percentuale di Sorgenti Luminose a Incandescenza (calcolata sul numero totale di Sorgenti Luminose)
CompactFluorescentLampPercentage	Percentuale di Sorgenti Luminose a Fluorescenza compatta (calcolata sul numero totale di Sorgenti Luminose)
TubolarFluorescentLampPercentage	Percentuale di Sorgenti Luminose a Fluorescenza tubolare (calcolata sul numero totale di Sorgenti Luminose)
HPSLampPercentage	Percentuale di Sorgenti Luminose a Sodio ad alta pressione (calcolata sul numero totale di Sorgenti Luminose)
LPSLampPercentage	Percentuale di Sorgenti Luminose a Sodio a bassa pressione (calcolata sul numero totale di Sorgenti Luminose)
MetalHalideLampPercentage	Percentuale di Sorgenti Luminose a Ioduri Metallici (calcolata sul numero totale di Sorgenti Luminose)
HalogenLampPercentage	Percentuale di Sorgenti Luminose Alogene (calcolata sul numero totale di Sorgenti Luminose)
LEDLampPercentage	Percentuale di Sorgenti Luminose a LED (calcolata sul numero totale di Sorgenti Luminose)
CrepuscularSystemPercentage	Percentuale di Quadri Elettrici con sistema di accensione crepuscolare (calcolata sul numero totale di Quadri elettrici)
ClockSystemPercentage	Percentuale di Quadri Elettrici con sistema di accensione con orologio (calcolata sul numero totale di Quadri elettrici)
AstronomicalClockSystemPercentage	Percentuale di Quadri Elettrici con sistema di accensione con orologio astronomico (calcolata sul numero totale di Quadri elettrici)
ManualSystemPercentage	Percentuale di Quadri Elettrici con sistema di accensione manuale (calcolata sul numero totale di Quadri elettrici)
RemoteControlSystemPercentage	Percentuale di Quadri Elettrici con sistema di accensione con telecontrollo (calcolata sul numero totale di Quadri elettrici)
PartializedSystemPercentage	Percentuale di Quadri Elettrici con parzializzazione tutta notte - mezza notte (calcolata sul numero totale di Quadri elettrici)
LuminousFluxReducingSystemPercentage	Percentuale di Quadri Elettrici con riduzione del flusso luminoso (calcolata sul numero totale di Quadri elettrici)
CentralizedLightFluxRegulationPercentage	Percentuale di Quadri Elettrici con regolazione del flusso luminoso centralizzata (calcolata sul numero totale di Quadri elettrici)

Tabella di sintesi della lista di codici "CityPLIndicatorCode"

PointToPointLightFluxRegulationPercentage	Percentuale di Quadri Elettrici con regolazione del flusso luminoso punto a punto (calcolata sul numero totale di Quadri elettrici)
TotalEnergy	Energia totale dell'impianto di Illuminazione Pubblica
PerCapitaEnergyConsumption	Consumo medio di energia per abitante
PerSquareMetreEnergyConsumption	Consumo medio di energia per metro quadro comunale
PerLightSpotConsumedEnergy	Consumo medio di energia per Punto luce
MercuryVaporLampInstalledPowerDensity	Densità di Potenza installata per le Sorgenti Luminose a Vapori di mercurio (valore medio calcolato per tipo sorgente)
IncandescentLampInstalledPowerDensity	Densità di Potenza installata per le Sorgenti Luminose a Incandescenza (valore medio calcolato per tipo sorgente)
CompactFluorescentLampInstalledPowerDensity	Densità di Potenza installata per le Sorgenti Luminose a Fluorescenza compatta (valore medio calcolato per tipo sorgente)
TubolarFluorescentLampInstalledPowerDensity	Densità di Potenza installata per le Sorgenti Luminose a Fluorescenza tubolare (valore medio calcolato per tipo sorgente)
HPSLampInstalledPowerDensity	Densità di Potenza installata per le Sorgenti Luminose a Sodio ad alta pressione (valore medio calcolato per tipo sorgente)
LPSLampInstalledPowerDensity	Densità di Potenza installata per le Sorgenti Luminose a Sodio a bassa pressione (valore medio calcolato per tipo sorgente)
MetalHalideLampInstalledPowerDensity	Densità di Potenza installata per le Sorgenti Luminose a Ioduri Metallici (valore medio calcolato per tipo sorgente)
HalogenLampInstalledPowerDensity	Densità di Potenza installata per le Sorgenti Luminose Alogene (valore medio calcolato per tipo sorgente)
LEDLampInstalledPowerDensity	Densità di Potenza installata per le Sorgenti Luminose a LED (valore medio calcolato per tipo sorgente)
OversizedElectricPanelPercentage	Percentuale di Quadri Elettrici che hanno più di 40 Punti Luce collegati.
MinLightSpotCount	Numero minimo di Punti Luce associati ad un Quadro Elettrico
MaxLightSpotCount	Numero massimo di Punti Luce associati ad un Quadro Elettrico
AverageLightSpotCount	Numero medio di Punti Luce associati ad un Quadro Elettrico
RemoteManagedElectricPanelPercentage	Percentuale di Quadri Elettrici con gestione remota
MinCorrelatedColorTemperature	Minimo valore di Temperatura di Colore Correlata
MaxCorrelatedColorTemperature	Massimo valore di Temperatura di Colore Correlata

Appendice C – File di configurazione report per SCP-DASH

In questa Appendice viene riportato un file di configurazione report per SCP-DASH. Riteniamo importante riportare l'intero listato JSON del file di configurazione, utilizzato per il primo mock di dashboard (si veda par.4.2.2) perché si evince come per ogni chart (grafico o tabella) vi è una configurazione apposita che però segue il formato ideato per descrivere le configurazioni (che viene anche validato con un apposito schema json), e che a ogni configurazione è associata un UrbanDataset specifico (tramite il suo *resource_id*) e un mapping delle proprietà in esso contenute con le caratteristiche di visualizzazione del chart.
Si prenda, per esempio, il chart con chartId=1: come si può leggere dalla descrizione presente nell'elemento stesso, esso corrisponde alla configurazione della tabella che permette una view istantanea (senza ulteriori rielaborazioni) dei dati meteo contenuti nell'UrbanDataset "WeatherCondition" prodotto dalla Solution "DecisionSupportSystem" (tabella mostrata nel par.4.2.2, figura "SCP-DASH Report Screenshot 2").


```
{
  "report": {
    "id": "scp-1_report-1",
    "label": "SCP-1 Report 1",
    "description": "Report 1 per SCP-Dash collegata a SCP Smart Village Casaccia",
    "position": 2,

    "charts": {
      "chart": [

        {
          "chartId": "1",
          "chartType": "table",
          "description": "Chart Table 1 con view istantanea dei dati Meteo.",
          "resourceId": "SCP-1_DecisionSupportSystem-11_WeatherCondition-1.0_20181001120000",
          "hoursFrequencyUpdating": 3,
          "title": "Meteo ultime 6 ore (Weather Condition)",
          "subtitle": "exp.subtitle(ud.UrbanDataset.context.timestamp, ud.UrbanDataset.context.timeZone)",
          "mapping": {
            "line": [
              {
                "lineId": 1,
                "labels": [
                  "Stazione Meteo"
                ],
                "values": [
                  "exp.tableValue(ud.UrbanDataset.values.line[0].property, 'MeteoStationName')"
                ],
                "description": "exp.tableLineDescription(ud.UrbanDataset.specification.properties.propertyDefinition, 'MeteoStationName')",
                "unit": "exp.tableLineUnitOfMeasure(ud.UrbanDataset.specification.properties.propertyDefinition, 'MeteoStationName')"
              },
              {
                "lineId": 2,
                "labels": [
                  "Temperatura Esterna"
                ],
                "values": [
                  "exp.tableValue(ud.UrbanDataset.values.line[0].property, 'AirTemperature')"
                ],
                "description": "exp.tableLineDescription(ud.UrbanDataset.specification.properties.propertyDefinition, 'AirTemperature')",
                "unit": "exp.tableLineUnitOfMeasure(ud.UrbanDataset.specification.properties.propertyDefinition, 'AirTemperature')"
              },
              {
                "lineId": 3,
                "labels": [
                  "Precipitazioni"
                ],
                "values": [
                  "exp.tableValue(ud.UrbanDataset.values.line[0].property, 'Rainfall')"
                ],
                "description": "exp.tableLineDescription(ud.UrbanDataset.specification.properties.propertyDefinition, 'Rainfall')",
                "unit": "exp.tableLineUnitOfMeasure(ud.UrbanDataset.specification.properties.propertyDefinition, 'Rainfall')"
```

```
        }
      ]
    }
  },

  {
    "chartId": "2",
    "chartType": "table",
    "description": "Chart Table 2 con totale dei consumi Building su una settimana.",
    "resourceId": "SCP-1_DFSWeeklyBuildingConsumption-2_Whatever-1.0_20180701120000",
    "hoursFrequencyUpdating": 3,
    "title": "Consumo elettrico medio quotidiano su Ultima Settimana",
    "subtitle": "exp.subtitle(ud.UrbanDataset.context.timestamp, ud.UrbanDataset.context.timeZone)",
    "mapping": {
      "line": [
        {
          "lineId": 1,
          "labels": [
            "Edificio A",
            "ENEA Bologna"
          ],
          "values": [
            "exp.tableValue(ud.UrbanDataset.values.line[0].property, 'Edificio-A')"
          ],
          "description": "exp.tableLineDescription(ud.UrbanDataset.specification.properties.propertyDefinition, 'Edificio-A')",
          "unit": "exp.tableLineUnitOfMeasure(ud.UrbanDataset.specification.properties.propertyDefinition, 'Edificio-A')"
        },
        {
          "lineId": 2,
          "labels": [
            "Edificio B",
            "ENEA Bologna"
          ],
          "values": [
            "exp.tableValue(ud.UrbanDataset.values.line[0].property, 'Edificio-B')"
          ],
          "description": "exp.tableLineDescription(ud.UrbanDataset.specification.properties.propertyDefinition, 'Edificio-B')",
          "unit": "exp.tableLineUnitOfMeasure(ud.UrbanDataset.specification.properties.propertyDefinition, 'Edificio-B')"
        },
        {
          "lineId": 3,
          "labels": [
            "F40",
            "ENEA Casaccia"
          ],
          "values": [
            "exp.tableValue(ud.UrbanDataset.values.line[0].property, 'F40')"
          ],
          "description": "exp.tableLineDescription(ud.UrbanDataset.specification.properties.propertyDefinition, 'F40')",
          "unit": "exp.tableLineUnitOfMeasure(ud.UrbanDataset.specification.properties.propertyDefinition, 'F40')"
        }
      ]
    }
  },

  {
    "chartId": "3",
    "chartType": "line-basic",
    "description": "Chart LineChart 1 per visualizzazione andamento consumi Building su ultimo mese.",
    "resourceId": "SCP-1_DFSMonthlyBuildingConsumption-3_Whatever-1.0_20180701120000",
    "hoursFrequencyUpdating": 6,
    "title": "Andamento Consumi Edifici ENEA Bologna Ultimo Mese",
    "subtitle": "exp.subtitle(ud.UrbanDataset.context.timestamp, ud.UrbanDataset.context.timeZone)",
    "axis": [
      "Time",
      "Consumption in kiloWatthour"
    ],
    "mapping": {
      "line": [
        {
          "lineId": 1,
```

```
            "labels": [
              "Edificio A",
              "ENEA Bologna"
            ],
            "values": [
              "exp.lineBasicValues(ud.UrbanDataset.values.line[0].property, 'name', 'val')"
            ],
            "description": "ud.UrbanDataset.values.line[0].description",
            "unit": "exp.tableLineUnitOfMeasure(ud.UrbanDataset.specification.properties.propertyDefinition, 'Edificio-A')"
          },
          {
            "lineId": 2,
            "labels": [
              "Edificio B",
              "ENEA Bologna"
            ],
            "values": [
              "exp.lineBasicValues(ud.UrbanDataset.values.line[1].property, 'name', 'val')"
            ],
            "description": "ud.UrbanDataset.values.line[1].description",
            "unit": "exp.tableLineUnitOfMeasure(ud.UrbanDataset.specification.properties.propertyDefinition, 'Edificio-B')"
          },
          {
            "lineId": 3,
            "labels": [
              "F40",
              "ENEA Casaccia"
            ],
            "values": [
              "exp.lineBasicValues(ud.UrbanDataset.values.line[2].property, 'name', 'val')"
            ],
            "description": "ud.UrbanDataset.values.line[2].description",
            "unit": "exp.tableLineUnitOfMeasure(ud.UrbanDataset.specification.properties.propertyDefinition, 'F40')"
          }
        ]
      }
    },

    {
      "chartId": "4",
      "chartType": "dual-axis",
      "description": "Chart LineChart 2 per visualizzazione andamento consumi/produzioni Smart Home (la linea1 è la curva lineare, la linea2 è l'istogramma, la linea3 sono proprietà comuni).",
      "resourceId": "SCP-1_DFSMonthlySmartHomeProductionConsumption-4_Whatever-1.0_20180701120000",
      "hoursFrequencyUpdating": 6,
      "title": "Andamento Produzioni e Consumi Smart Home Casaccia",
      "subtitle": "exp.subtitle(ud.UrbanDataset.context.timestamp, ud.UrbanDataset.context.timeZone)",
      "axis": [
        "Time",
        "Production in kiloWatthour",
        "Consumption in kiloWatthour"
      ],
      "mapping": {
        "line": [
          {
            "lineId": 1,
            "labels": [
              "UnixTimestamp",
              "Produzione",
              "FloorArea",
              "OccupantNumber"
            ],
            "values": [
              "exp.dualAxisValues(ud.UrbanDataset.values.line[0].property, 'name', 'val')",
              "exp.dualAxisCommonValues(ud.UrbanDataset.values.line[2].property, {'FloorArea','OccupantNumber'})"
            ],
            "description": "ud.UrbanDataset.values.line[0].description",
            "unit": "exp.tableLineUnitOfMeasure(ud.UrbanDataset.specification.properties.propertyDefinition, 'Smart-Home-Casaccia')"
          },
          {
            "lineId": 2,
```

```
            "labels": [
              "UnixTimestamp",
              "Consumo",
              "FloorArea",
              "OccupantNumber"
            ],
            "values": [
              "exp.dualAxisValues(ud.UrbanDataset.values.line[1].property, 'name', 'val')",
              "exp.dualAxisCommonValues(ud.UrbanDataset.values.line[2].property, {'FloorArea','OccupantNumber'})"
            ],
            "description": "ud.UrbanDataset.values.line[1].description",
            "unit": "exp.tableLineUnitOfMeasure(ud.UrbanDataset.specification.properties.propertyDefinition, 'Smart-Home-Centoce')"
          }
        ]
      }
    },

    {
      "chartId": "5",
      "chartType": "map",
      "description": "Chart Map delle Solution Verticali connesse alla SCP corrente.",
      "resourceId": "SCP-100_DFSSCPMAP-5_Whatever-1.0_20180701120002",
      "hoursFrequencyUpdating": 6,
      "title": "Mappa dei Verticali connessi",
      "subtitle": "exp.subtitle(ud.UrbanDataset.context.timestamp, ud.UrbanDataset.context.timeZone)",
      "mapping": {
        "line": [
          {
            "lineId": 1,
            "labels": [
              "Province",
              "N. Solutions",
              "SCP"
            ],
            "values": [
              "exp.mapValues(ud.UrbanDataset.values.line[0].property, {'Province', 'ManagedSolutions', 'SCPName'})"
            ],
            "description": "",
            "unit": "adimensional"
          },
          {
            "lineId": 2,
            "labels": [
              "Province",
              "N. Solutions",
              "SCP"
            ],
            "values": [
              "exp.mapValues(ud.UrbanDataset.values.line[1].property, {'Province', 'ManagedSolutions', 'SCPName'})"
            ],
            "description": "",
            "unit": "adimensional"
          }
        ]
      }
    }
  ]
},
```

```
 "layout": [
      {
        "rowId": 1,
        "chartId": "1",
        "rowSpan": 0.5
      },
      {
        "rowId": 2,
        "chartId": "2",
        "rowSpan": 0.5
      },
      {
        "rowId": 3,
        "chartId": "3",
        "rowSpan": 1
      },
      {
        "rowId": 4,
        "chartId": "4",
        "rowSpan": 1
      },
      {
        "rowId": 5,
        "chartId": "5",
        "rowSpan": 1
      }
    ]
  }
}
```


Nell'ultimo elemento "*layout*", ogni chart (grafico o tabella) definito in precedenza, viene collocato in una griglia a matrice composta da due colonne e un numero di righe illimitato, coprendo mezza riga o la riga intera. Si noti che, utilizzando device più piccoli (p.es. smartphone) i grafici verranno disposti su una singola colonna, permettendo così al report di essere adattivo con il device utilizzato per la visualizzazione.