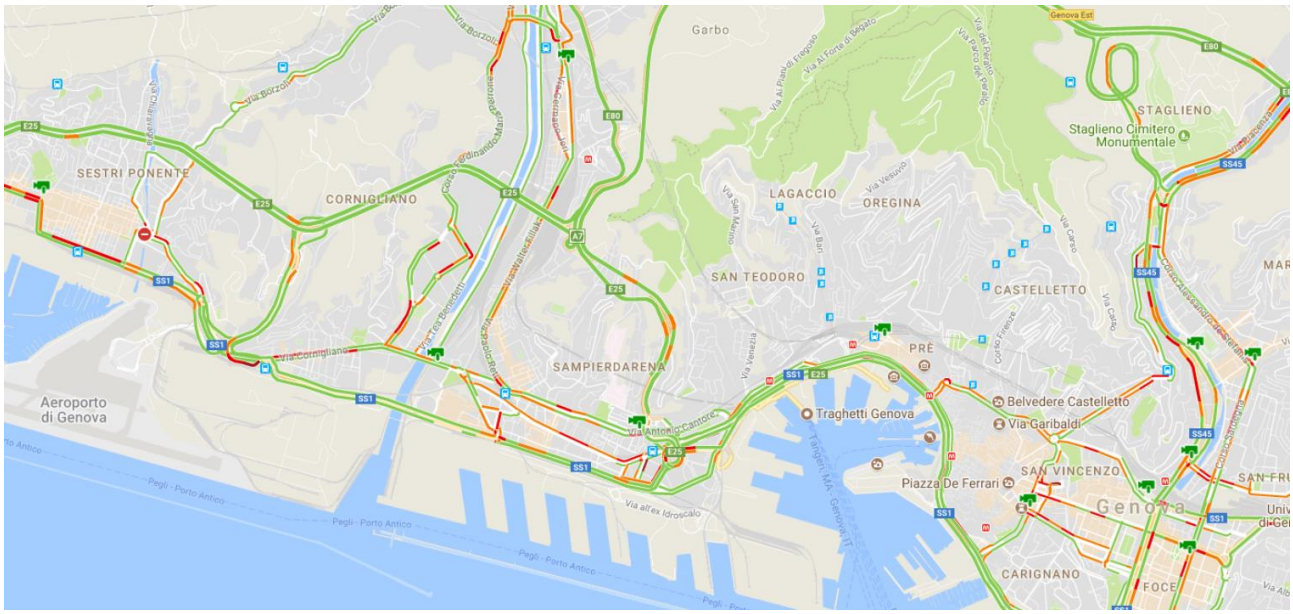
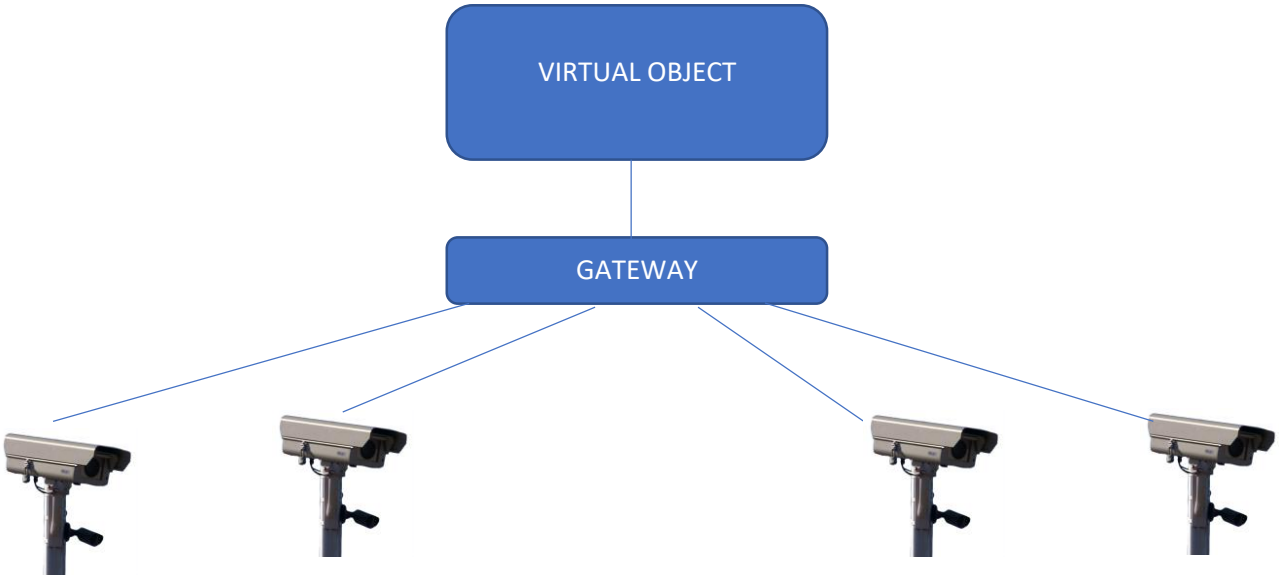
Cameras VO – Specifiche

Il virtual object “cameras vo” salva ed elabora le immagini provenienti dalle telecamere installate sul territorio per rilevare il numero di accessi a determinate aree e segnalare quando gli accessi superano un determinato limite,

Oggetti reali



Le telecamere che invieranno i dati al virtual object saranno quelle installate nel territorio del comune di Genova per rilevare il traffico. Le webcam registrano una nuova immagine ogni tre minuti. Tutte le immagini delle telecamere sono accessibili direttamente tramite un servizio REST (GET). Tutte le telecamere sono interrogate da un gateway che si occuperà di inviare i dati al virtual object tramite una POST REST.



Gateway

Il gateway è un loop sempre attivo che si occupa ogni 3 minuti di:

- Interrogare il servizio per visualizzare i metadati di tutte le telecamere presenti sul territorio, il timestamp dell'ultimo aggiornamento e il link per recuperare l'immagine.
  URL GET A http://www.iononrischioclout.comune.genova.it/back_end/webcam.php?checkimage
- Interrogare tutte le telecamere presenti e recuperare l'immagine da inviare al virtual object. Esempio GET a http://mobilitypoint.comune.genova.it/WebCams/Images/15.jpg come su classe esempio.
- Inviare al VO l'immagine serializzata in base64 (come su classe esempio) e i metadati con una POST al servizio new data del VO. Inserendo tutti i dati in un JSON. Il formato del JSON si può recuperare effettuando una GET al servizio getInfo del virtual object.

Virtual object

Inizializzazione.

Caricare nel servlet context le properties presenti nel file di properties. In particolare URL VORegister e URL Orchestratore.

Inizializzare il logger su un rolling file appender.

Inizializzare il VO inviando al VORegister:

- Nome
- Indirizzo IP/URL su cui è in ascolto la invoke action
- Hashmap azioni invocabili-descrizione
- Hashmap eventi-descrizione
- Proprietà

Elenco azioni invocabili su VO

Nessuna.

Elenco eventi inviabili da VO

NUM_MAX_ACCESS_EXCEEDED

Modulo comunicazione real object – virtual object.

CLASSE VORealObjectCommunication, classe annotata JAX-RS che contiene:

- FUNZIONE NEW DATA: Espone funzione annotata JAX-RS con metodo POST 'newData' chiamabile dagli oggetti reali.

esempio POST: *http://localhost:8080/WeatherStationVO/rest/VORealObjectCommunication/newData*

- FUNZIONE GET INFO: Espone funzione annotata JAX-RS con metodo GET 'getInfo' chiamabile dagli oggetti reali per ricavare informazioni sull'oggetto virtuale (formato dati inviabile su new data)

Modulo comunicazione Spreadsheet Space.

Sviluppare lo Spreadsheet Space connection module nel virtual object..

- Alla ricezione di un nuovo messaggio 'newData' il virtual object istanzia il thread SSSConnection che chiama il metodo updateSSSData(String payload). Dove il payload è il JSON ricevuto dal client.
- updateSSSData legge il payload inserendo i dati in una struttura dati locale e legge l'id della telecamera.
- Se è la prima volta che riceve i dati della telecamera crea una nuova view chiamata (cameradata_cameraID) dove cameraID è l'ID della stazione meteo contente un range di tipo 'table' (a dimensione variabile). La view deve essere esposta al proprio account e all'account 'giancarlo.camera@spreadsheetspace.net'.
- Se è già stata creata una view per quella telecamera bisogna aggiornare la tabella contente i dati di quella telecamera.
- I dati da inserire in ogni riga sono timestamp, immagine serializzata in stringa base 64.
- Il metodo ritorna un messaggio di successo in caso di buona riuscita dell'operazione o di errore in caso di problemi. Il messaggio deve essere loggato sul file di log impostato.

Modulo core.

CLASSE onReceiveNewData

Estende classe **thread*** e contiene:

- FUNZIONE (finta) per il processing dell'immagine e rilevamento del numero degli accessi che invoca una notifyEvent(..) se è stato superato il limite di accessi consentiti.
- FUNZIONE notifyEvent () : Client REST che chiama metodo POST notify event dell'orchestratore per notificare una particolare condizione verificatasi sul virtual object (NUM_MAX_ACCESS_EXCEEDED)

Modulo comunicazione virtual object – orchestratore.

Nessuna funzione invoke action disponibile.