QROWD—A Platform for Integrating Citizens in Smart City Data Analytics

Luis-Daniel Ibáñez, Eddy Maddalena, Richard Gomer, Elena Simperl, Mattia Zeni, Enrico Bignotti, Ronald Chenu-Abente, Fausto Giunchiglia, Patrick Westphal, Claus Stadler, Gordian Dziwis, Jens Lehmann, Semih Yumusak, Martin Voigt, Maria-Angeles Sanguino, Javier Villazán, Ricardo Ruiz, and Tomas Pariente-Lobo

Abstract Optimizing mobility services is one of the greatest challenges Smart Cities face in their efforts to improve residents' wellbeing and reduce CO_2 emissions. The advent of IoT has created unparalleled opportunities to collect large amounts of data about how people use transportation. This data could be used to ascertain the quality and reach of the services offered and to inform future policy—provided cities have the capabilities to process, curate, integrate and analyse the data effectively. At the same time, to be truly 'Smart', cities need to ensure that the data-driven decisions they make reflect the needs of their citizens, create feedback loops, and widen participation. In this chapter, we introduce QROWD, a data integration and analytics platform that seamlessly integrates multiple data sources alongside human, social and computational intelligence to build hybrid, automated data-centric workflows. By doing so, QROWD applications can take advantage of the best of both worlds: the accuracy and scale of machine computation, and the skills, knowledge and expertise of people. We present the architecture and main components of the platform, as well as its usage to realise two mobility use cases: estimating the modal split, which refers to trips people take that involve more than one type of transport, and urban auditing.

L.-D. Ibáñez (✉) · R. Gomer
University of Southampton,Southampton, UK
e-mail: l.d.ibanez-gonzalez@soton.ac.uk

E. Maddalena · E. Simperl
King's College London,London, UK

M. Zeni · E. Bignotti · R. Chenu-Abente · F. Giunchiglia
University of Trento,Trento, Italy

P. Westphal · C. Stadler · G. Dziwis · J. Lehmann
Institute for Applied Informatics (InfAI),Leipzig, Germany

S. Yumusak
KTO Karatay University,Karatay, Turkey

S. Yumusak · M. Voigt
AI4BD AG, Zürich,Zürich, Switzerland

M.-A. Sanguino · J. Villazán · R. Ruiz · T. Pariente-Lobo
ATOS Research and Innovation,Madrid, Spain

P. K. Singh et al. (eds.), *Sustainable Smart Cities*, Studies in Computational Intelligence 942, https://doi.org/10.1007/978-3-031-08815-5_16

1 Introduction

In a world dominated by huge societal and environmental shifts, urban mobility is likely to remain one of the most pressing challenges cities need to tackle over the next decades. The UN 2030 development agenda states as one of its goal the provision of "access to safe, affordable, accessible and sustainable transport systems for all". In terms of environmental impact, road transport represents nearly 30% of the CO_2 emissions in Europe and the US; reducing this share is critical to fight global warming, calling for novel approaches and tools to improve urban mobility. Furthermore, commuting time spent on the road has been shown to have substantial effects on productivity and wellbeing; For example, according to the European Commission, traffic congestion costs the EU economy more than 100 billions every year.[1]

The advent of IoT has created unparalleled opportunities to collect large amounts of data about how people use transportation, real-time positions of public transport; traffic cameras and meters; weather reports, etc. However, to be truly *'Smart'*, city authorities must find ways to ensure that they make sense of all this data to drive their decisions, and that decisions reflect the needs and expectations of the people they serve [9]. This may include focus groups, co-design workshops, or ideas competitions, as well as crowdsourced data collection activities in which citizens can share data about their own transport patterns via mobile phones, wearables and other sensored devices to improve and provide feedback on existing services [5].

In this chapter, we introduce the QROWD platform, a data integration and analytics platform designed to include citizens in the data value chain of Smart Cities. It incorporates advanced interlinking and analysis capabilities for different sources of data, including human computation to train and validate algorithms, alongside means to crowdsource data collection and feedback. The platform is designed to develop and deploy arbitrary hybrid workflows that bring together the accuracy and performance of machine computation with human skills, knowledge and expertise that machines cannot emulate. In addition to the QROWD platform, we report on the design, implementation of two end-to-end mobility use cases deployed in the city of Trento, Italy: the estimation of the modal split, *i.e.*, the share of citizens that use each available mode of transport; and the auditing of infrastructure location and information data. We also report on the results and lessons learned from their deployment on the city of Trento, Italy.

This chapter extends previous work published in the 5th International Smart Cities Workshop, colocated with Web Conference 2019, where we presented the design and implementation of the modal split and mobility infrastructure use cases; and i-Log, a crowdsensing mobile application (i-Log) to collect data from personal mobile phones in an unobtrusive, responsible way [17]. This chapter introduces the following novel contributions:

[1] https://ec.europa.eu/transport/themes/urban/urban_mobility_en.

- The QROWD platform. A data integration and analytics architecture, compatible with the FIWARE set of open standards,[2] in use in more than 100 cities in the world, which supports the design, development and deployment of hybrid human-machine workflows for data collection, curation, integration and analysis.
- A linked-data-enabled, big sensor data storage component (QROWDDB)
- Extensions of the FIWARE open standard data models for transport and mobility to include contributions and feedback from citizens and manage potential rewards or compensation.
- Guidelines to help Smart City managers design human computation tasks as part of hybrid workflows.
- Rewriting of the use cases design following the guidelines
- Extension of the technical details of the machine components developed for the use cases. The transport mode classifier used to analyse data contributed by citizens through the i-Log app and the interlinking component that uses semantic technologies to integrate heterogeneous data sources.

The remainder of the chapter is organised as follows: Sect. 2 reviews previous frameworks for Smart Cities aimed at leveraging IoT devices and those that tackle the human and citizen perspective. Section 3 describes the QROWD platform and architecture. Section 4 introduces the the guidelines for designing and implementing tasks for citizens and crowdworkers, and how to architect them as *crowdsourcing services* for their integration with machine processes. Section 5 describes the tools for acquiring data from pre-existing static and dynamic sources and crowdsourcing services included with the QROWD platform for data collection. Section 6 introduces the data models we developed to facilitate the inclusion of citizens, and the QROWDDB, a component to manage data related with citizens and their contributions in a privacy preserving way. Section 7 describes the data integration capabilities of the QROWD Platform. Section 8 describes how we used the QROWD Platform to develop and pilot two urban mobility use cases: generating and curating mobility infrastructure data and estimating modal split. We report on the results and lessons learned from the pilots.

2 Related Work

2.1 IOT for Smart Cities

Several frameworks have been developed to help Smart Cities harness the power of IoT infrastructures and sensor devices [10]. They can be classified according to their provision of the following capabilities [22] (1) data acquisition (2) semantic interoperability (3) real-time data analysis (4) application development support

[2] https://www.fiware.org/developers/.

SmartSantander created the first experimental test facility for the research and experimentation of architectures, key enabling technologies, services and applications for the IoT in the context of a Smart City. It provides the data acquisition and application development support dimensions [27]. The semantic interoperability dimension was considered in the SPITFIRE EU project, that developed vocabularies to integrate descriptions of sensors and things with the LOD cloud; Semantic entities as an abstraction for things with high-level states inferred from embedded sensors; Semi-automatic generation of semantic sensor descriptions; Efficient search for sensors and things; all on top of an unified service infrastructure. Another platform that focuses on the semantic dimension is OpenIoT, that leverages the W3C Semantic Sensor Network ontology to annotate data from sensor streams, providing also a toolkit for filtering, selecting them and visualizing them [21].

STAR-CITY integrates data of heterogeneous variety, velocity and volume, and combines description logic, rule-based reasoning, machine learning inferences and stream based correlation to provide spatio-temporal analysis of traffic conditions for their diagnosis and prediction. Its main application scenario is real-time data analysis and event detection of traffic events [15]. CityPulse aims at facilitating knowledge extraction from Smart City environments, using a combination of large scale data stream processing modules and adaptive decision support, while providing application development support, demonstrated by the development of a prototype adaptive travel planner app [22].

The IOTManager is a versatile and resilient framework capable to store and rearrange data collected by IoT sensors [2]. The main contribution is the provision of a software that could be easily deployed by public organisations and Smart City managers, without being tied to "platform-as-a-service" contracts with large providers. The framework is demonstrated with a case study on traffic controllers and weather stations.

Deepint introduces the concept of "City-as-a-platform" [8]. The main innovation is the introduction of wizards to easily create and deploy AI models for common Smart City tasks. Deepint is demonstrated with the implementation of a Crowd-counter to monitor pedestrian traffic levels by using video cameras in the City of Melbourne.

None of these frameworks considers the integration of citizens or human computation in general as part of their toolkit. The potential of human sensing for Smart Cities was first studied in [6]. For the particular case of social media, it presents a methodology to extract the perceptions that may be relevant to Smart City initiatives from social media updates, validated on dataset of tweets geolocalised in New York City. Beyond social media, [9] proposed a re-imagination of the role of citizens in Smart Cities, highlighting the importance of supporting them in playing an active role in urban innovation, from the crowdsourcing of initial ideas, to facilitating their involvement in the realisation of community projects.

TCitySmartF outlines a Smart Cities roadmap from the technological, social, economic and environmental point of view. It puts both residents and urban dynamics at the forefront of the development with participatory planning and interaction for the robust community- and citizen-tailored services. It also includes connections to

other cities, in order to create a region or country-wide network of data that could be used to implement further policies and share technical knowledge [13]. They define a high level architectural design that but do not provide an implementation.

2.2 Mobile Crowdsensing and Crowdsourcing

The field of Mobile Crowdsensing studies the wide variety of sensing models by which individuals collectively share data and extract information to measure and map phenomena of common interest, and for which several platforms have been discussed in the literature (see for example the survey on [14]).

A similar line of work studies the efforts of making mobile crowdsourcing useful for Smart Cities. Note that mobile crowdsourcing can be used for other types of problems, for example, to perform task within a private organisation (like an University campus), or for realising food or package delivery. The particular application for Smart Cities has been surveyed by Kong et al. [12]. According to their classification, the QROWD platform is a technology enabler for the use of mobile crowdsourcing in the context of Smart Cities.

Both fields are related to our work as one of the goals of the QROWD platform is the integration of the human factor, and citizens in particular, in the data value chain of a Smart City. The QROWD platform includes its own crowdsensing application (the i-Log app) that implements best practices from literature, and enables the flow of collected data towards data analytics components, and back to citizens for further curation. In this subsection, we review other frameworks that highlight a citizen-centric approach.

A discussion of the challenges for sustainable people-centric sensing is presented in [24], highlighting the importance of being energy-efficient for user's devices guaranteeing the privacy and security of people's data, and putting in place appropriate incentive mechanisms. Vol4All [28] enables ideas exchange and crowdsourcing by facilitating citizens' involvement in the realization of community projects. Volunteering actors (initiators, participants, stakeholders) can easily interact via the Vol4All platform which enables volunteering opportunities dynamic sharing, evolution and monitoring. Vol4All includes a point-system to incentivize participation in volunteering activities, and a number of tools for monitoring volunteering activity and analyse the result of specific campaigns.

OrganiCity [11] provides an Experimentation as a Service (EaaS) framework that aims at providing a scalable platform to manage city services and of a co-creation environment including citizens. It includes technical components to integrate different data sources and to host co-creation experiments that empower citizens at different stages of the urban service lifecycle. TO this end, it provides a set of co-creation tools:(1) SensiNact studio: aims at helping coders working with data streams from deployed data assets without the need to learn about the Organicity APIs.(2) TinkerSpace: Toolkit for creating mobile services—Apps—without the need for extensive software training or experience. Providing (3) Smartphone Experimentation: A

complementary framework that facilitates experimenter to gather and process data from the sensors and communication interfaces of the smartphones of volunteers and use them to run experiments.

Both Vol4All and OrganiCity restrict citizen participation to tools for contributing data. The QROWD platform extends this in two ways: first, it enables the harnessing of human computation, not only from citizens, but also from paid crowdworkers; second, human computation is not limited to data collection, but also integrated into all the other steps of the data value chain, providing the building blocks to create hybrid data flows comprised of several human and machine processing steps.

CitySpeed is an application and server to collect, manage and provide access to vehicular speed data. Participating citizens download a mobile application that monitors the speed of their vehicles [4]. The proposed mobile-phone based monitoring was found to match the speed as collected by the ECU units of a set of test vehicles. The whole system was piloted on two cities in Brazil. The theoretical framework for a similar application is described on [19]. CitySpeed could be re-factored as a component of the QROWD platform, giving the additional advantage of managing the incentives for the citizens that wish to participate.

3 The QROWD Platform and Architecture

The QROWD platform is designed to seamlessly connect human computation tasks (HCTs) with machine analytics process, reducing the friction for developers and enabling the continuous improvement of data and services. From an architectural point of view, it is divided into five sets of components, as shown in Fig. 1.

1. The **Crowdsourcing services** (bottom) component set is a repository of standalone HCTs. We detail how HCTs should be architected in Sect. 4.2.
2. The **Data generation and acquisition** (bottom right quadrant) includes a data storage component to host heterogeneous data sources that could be static (e.g., records of parking locations and fees) or dynamic (e.g. live streams of occupancy of said parking). In this component set, we also include machine components to perform data harvesting, extraction and semantization of data.
3. **Storage** (bottom left): Data acquired from citizens through crowdsourcing services and pre-existing data needs to be integrated for further analysis. This component set includes the QROWDDB, a Big Data Storage for personal big data generated by mobile devices of citizens, and the associated. We describe it in detail in Sect. 6.
4. **Data interlinking, fusion and analytics** (top right and top left): components that take as input data integrated in the QROWDDB and perform machine-based fusion and interlinking, or other data analytics.

To implement inter-component communication, we chose to adopt a technology stack consistent with the one promoted by the Open and Agile Smart Cities (OASC) alliance, a non-profit, international Smart City network that connects +140 Smart

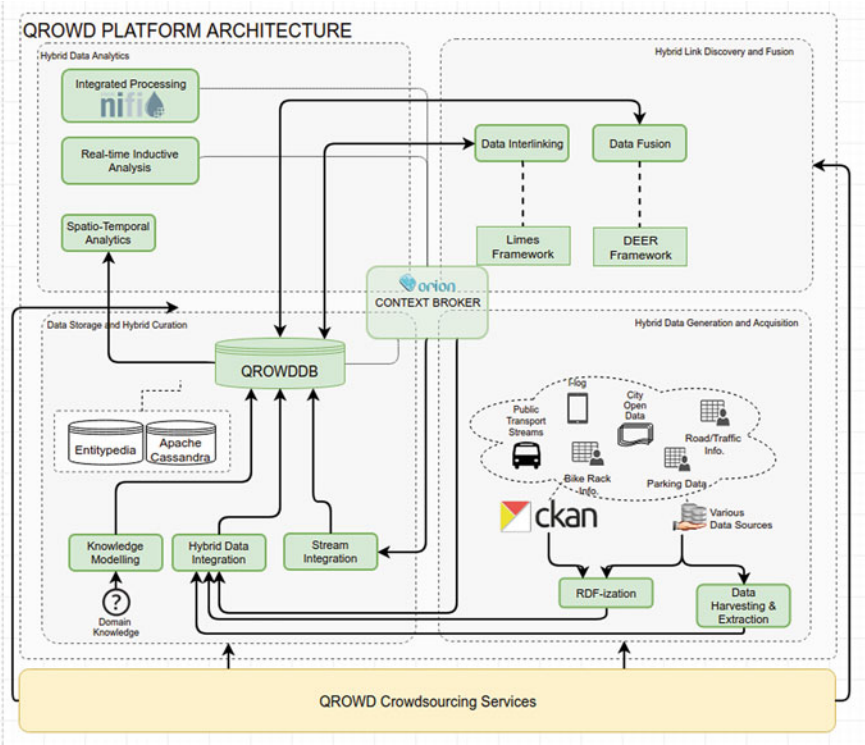

Fig. 1 QROWD Platform architecture

Cities globally organised in national networks from 27 countries and regions, aimed at establishing the Minimal Interoperability Mechanisms (MIMs) needed to create a Smart City market. The QROWD platform makes use of three core technologies:

1. Apache NiFi to host and execute data flows. The use of NiFi Is also consistent with the design decision of following the FIWARE architecture.
2. CKAN[3] is an open source, fully-featured, mature data portal and management solution that can be easily adapted and extended and provides an API. CKAN is used by hundreds of data publishers around the world and is the standard platform recommended by OASC to store datasets at rest. We use it as the repository for acquired data.
3. FIWARE context broker (Orion). Orion manages the entire lifecycle of context information including updates, queries, registrations and subscriptions. Context information consists on entities (e.g. a car) and their attributes (e.g. the speed or location of the car). Orion implements the NGSIv2 specification.[4] The QROWD

[3] https://ckan.org/.

[4] https://fiware.github.io/specifications/ngsiv2/stable/.

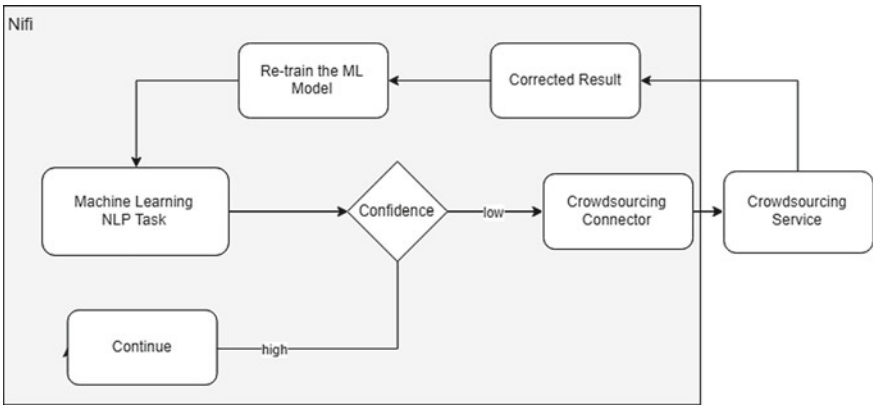

Fig. 2 Integration of machine data analytics task as Apache NiFi processor and human computation task deployed as crowdsourcing service

platform use Orion to manage streaming and sensor data, and to orchestrate message passing across different components.

To illustrate, consider the problem of annotating text with entities from a knowledge base. A machine learning model is trained to output annotations and a confidence value. Machine annotations may or may not be correct, therefore, humans could be recruited to validate them and provide new ones that could be used for re-training the model. How to seamlessly connect inputs and outputs of both types of processes? Fig. 2 illustrates how to do this with the QROWD platform, annotation is implemented as a a human computation task and deployed as a crowdsourcing service. The Machine Learning model is deployed as a NiFi processor calls the crowdsourcing service through a **crowdsourcing connector** whenever the confidence value of an annotation is under a configurable threshold. The corrected result is used as input to re-train the model.

4 Crowdsourcing Services

When designing a hybrid human-machine workflow, the first step is to have a clear separation between tasks to be executed by machines and Human Computation TAsks (HCTs). Designing HCTs is fundamentally different from designing a machine-only data pipeline. Designers need to consider an appropriate user interface, what incentive humans have to perform the tasks, and the general unpredictability of human behaviour. Lack of proper thinking about what, how and why a human engages in a task might lead to poor quality of results, or even no results at all. In the following, we

discuss how the QROWD platform helps Smart City managers and service providers with the design of HCTs and their implementation as services for their integration with machine data processing, as described in Sect. 3.

4.1 Design Guidelines for Human Tasks

Several frameworks have proposed a taxonomy of dimensions that need to be considered for general purpose crowdsourcing tasks literature to design effective and efficient human and crowdsourcing tasks [18, 23, 25]. However, they all overlook some important dimensions for an hybrid context: first: the characteristics and restrictions of devices required to fulfill the task; second, for data in motion or streams, human tasks need to output results at a certain velocity consistent with the processing speed of a machine component, suggesting the consideration of an *acceptable delay* dimension; third, depending on the type of contributors, one might only be able to tolerate a certain delay to assign tasks to them before they lose attention or consider the proposed incentives as insufficient for the time they invest. To fill this gap, we developed a guideline (Table 1) that combines the most important dimensions of previous frameworks and adds five new ones tailored to hybrid human-machine workflows. The first column of the guideline is the name of the dimension; the second one indicates either the source framework, or if it is introduced by us; the third column lists sample values for the dimension. To apply the guideline, HCTs designers must ask themselves for each dimension which value on the third column corresponds to the task being designed. In the following, we expand the questions associated with each dimension and the sample values. We proceed in the same order given in Table 1

- **What is going to be done?** This dimension refers to both what is required to the crowd and what goal the requester wants to achieve. In [18], possible values are information finding, verification and validation, and content creation. We add passive and active sensing as two activities often required in the crowdsensing context.
- **Who is carrying out the task?** The *who* represents the type of crowd. In some cases contributors are drawn from an undetermined group of people, meaning no assumptions regarding their skills can be made. However, contributors with particular skillsets are often required, such as polyglots for translation tasks, or citizens of a particular city for location-dependent tasks. Online crowdsourcing platforms also often implement strategies aimed at identifying the best or most reliable contributors, who may receive access to special benefits and privileges. We propose a novel set of values with respect to the literature: (1) Experts, if specific knowledge is required (2) Citizens, for location-dependent tasks (3) Whoever, if the task could be assigned to any crowdworker (4) Specific contributor, when the task can only be performed by a specific person, e.g., the verification of personal data provided by a citizen may only be done by the concerned citizen.

Table 1 Guideline for human computation task design

Dimension	Based on	Sample Value (values in **bold** are those proposed by us)
What	Malone et al. [18]	Information finding Verification and validation Interpretation and analysis Content creation Surveys content access **Passive sensing** **Active sensing**
Who	Malone et al. [18]	**Expert** **Citizen** **Anyone** **Specific contributor**
Why—Motivation	Malone et al. [18], Smart et al. [25]	Economic Altruistic Hedonic Reputational Other
Why—Reward	Malone et al. [18], Smart et al. [25]	None Monetary Prize Fun Other
How	Malone et al.[18]	Collection Collaboration Context
Required skill	Quinn and Bederson	Visual recognition Language understanding Basic communication **Physical**
Required device	Novel	**PC** **Mobile** **None**
Device constraint	Novel	**Battery** **Storage** **Bandwidth** **CPU** **None**
Interaction limit	Novel	**[0-N]**
Acceptable question delay	Novel	**Immediate (seconds)** **Short (minutes)** **Medium (hours)** **Long (days)**
Acceptable resolution delay	Novel	**Immediate (seconds)** **Short (minutes)** **Medium (hours)** **Long (days)**

- **Why is this task being performed?** This dimension concerns the reason why a contributor would engage in the task. It is split into two dimensions: motivation, related to the intrinsic value for the contributor, e.g., is she doing it for the money (economical), for reputation, or for altruism?; and the concrete reward that she will get for completing the task.
- **How the task will be organised?** We consider three different ways of organising a task: (1) Collection: The task is partitioned in several independent micro-tasks that are then assigned to one or more contributors. Results are then collected an aggregated. (2) Collaboration: Contributors collaborate in solving the task. (3) Contest: contributors compete to better perform the task, rewards are higher for the winners of the contest.
- **What skill is required to complete the task?** Previous work considered visual recognition, language understanding and basic communication [23]. We added specific mobility requirements to accommodate location-based tasks. Note however that some tasks may have different skill requirements.
- **Do contributors require a device to complete the task?** Especially for sensing tasks, contributors might need to be in possession of a connected device.
- **Does the device have any constraints?** If a device is needed, it is important to consider constraints it might have, especially for the case that the device belongs to the contributor. We consider in our framework the basic constraints of an IoT device: battery, storage, bandwidth and CPU.

A further set of dimensions that need to be considered are those of *quality and aggregation*, that is, how individual contributions will be quality-checked and aggregated into a final result. Table 2 describes the necessary dimensions and values, that we mostly re-use from [23]. We added '*Formula*' to the aggregation dimension to refer to the algorithm that aggregates the set of individual contributions into a final result in order to include techniques beyond simple aggregation, such as clustering.

Once the guidelines have been applied, the next step is to implement the design choices in such a way that they can input and output to a workflow composed of machine and human processes. To this end, QROWD provides a specification of the high level components that a crowsourcing task needs to implement to become a crowdsourcing service and fit in the QROWD architecture.

4.2 Crowdsourcing Service Implementation Framework

Once a crowdsourcing task has been designed, the next step is to ensure that its input/output can be easily plugged from/to other data processes. QROWD proposes a framework for implementing human tasks as *crowdsourcing services*, that interact with other components of the QROWD platform. The elements of the framework are described in Fig. 3, together with their interactions with inputs, crowdsourcing channels and the QROWDDB.

Table 2 Quality and aggregation dimensions of a human task

Dimension	Based on	Sample Value
Quality control	Quinn and Bederson [23]	Output agreement Input agreement Economic models Defensive task design Redundancy Statistical filtering Multilevel review Automatic check Reputation system
Aggregation	Quinn and Bederson [23]	Collection Statistical processing of data Iterative improvement Active learning Statistical Search Iterative improvement None **Formula**
Task request cardinality	Quinn and Bederson [23]	One-to-one Many-to-many Many-to-one Few-to-one

The first element is a repository of **task and/or question templates**. A task template is a set of source codes, libraries, and resources (such as texts, images, appropriate handlers for the incoming data item) that define the logic of a human task, including aggregation and quality assurance methods chosen from the list defined in Table 2.

The second element is a **decision component** that handles the *who*, *why*, *reward*, *device constraints*, *interaction limit* and *acceptable delay* dimensions of the design guidelines. More precisely, a decision component must include:

- The list of contributors to the task
- A register of how many tasks have been assigned to each contributor and a counter of interactions
- Register of each contributor's answering time and its difference with respect to the acceptable delay
- If relevant, track of any device constraints associated to each contributor
- An assignment function that given a processing or generation request, instantiates a task template and decides to which contributor(s) assign it.

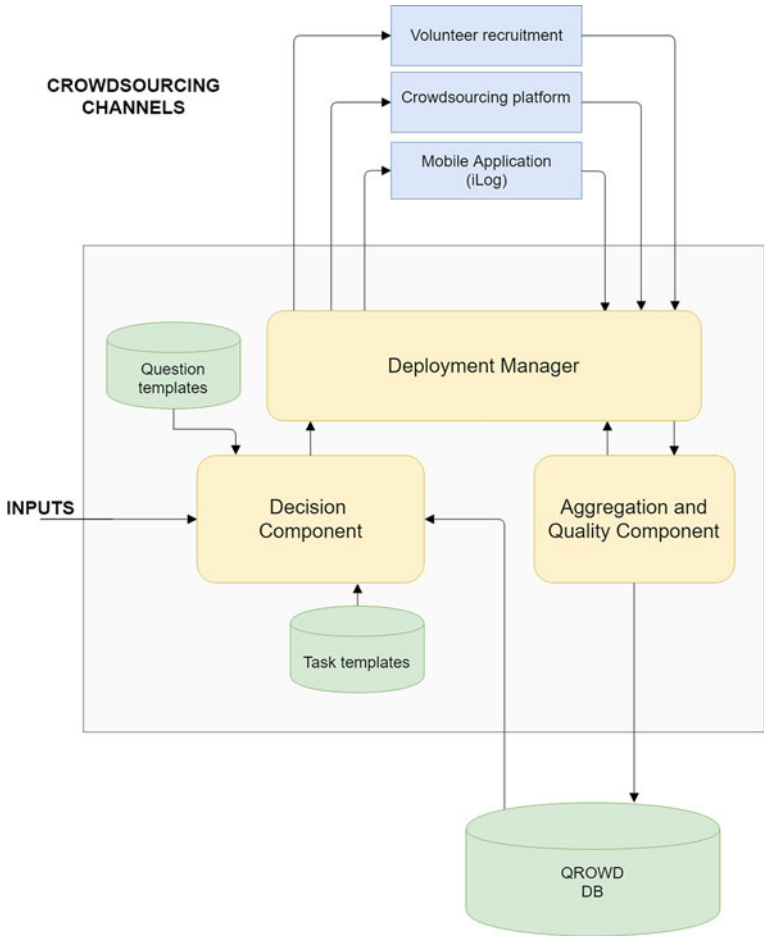

Fig. 3 Architecture of a crowdsourcing service within QROWD

The instantiated task template output by the assignment function is passed to a **Deployment Manager** that handles the deployment of the task on the relevant crowdsourcing channel and collects the results. Results are passed to an **Aggregation and Quality Component**, that based on the logic provided in the task template, executes the relevant quality checks and aggregations. If results are of insufficient quality, the decision component may decide to redeploy the task for further iterations. The final results are written either back to the context broker, or to the QROWDDB.

5 Data Acquisition and Generation

5.1 Pre-existing Data Sources

Cities often have pre-existing data sources that they would like to integrate to perform analytics or to connect with crowdsourcing services. The QROWD platform supports the acquisition of pre-existing data sources for both *static* sources that have a low update rate (also called *data at rest*); and *dynamic* sources coming from streams and other sensors (also called *data in motion*). Static data acquisition is organised around the concept of a *dataset* that is responsibility of an *organisation*, was produced by a particular *source*, has a number of different *formats* and is annotated with *provenance* information. Consider the example of a map with the coordinates and types of bike racks in the city. A possible source of this information is a city expert that has collected them for a certain area, a second possible source is a Volunteered Geographical Information system, e.g., Open Street Maps, or a bike-enthusiast association.

The procedure to add a dataset is as follows:

1. The originator of the dataset is added as a CKAN organisation
2. The visibility of the dataset is set (public or private)
3. The name of the dataset is constructed by concatenating the following input:
   - Name of the dataset, e.g., *Bike Racks*
   - Version, one of *lastVersion* or *Historical*
   - Type of the dataset *Source* (indicating that an organisation produced the dataset), *Fusion* (indicating the dataset is the result of a fusion via a QROWD automated or crowdsourced process).

When a dataset is updated, the platform manages versions automatically by backing up the contents of the current dataset in a new archive dataset tagged with the timestamp of archival.

QROWD provides three acquisition process templates according to the need or not for executing a data transformation on the acquired dataset:

- Upload/Update a dataset: takes a dataset available in a remote URL path and creates a new dataset if the name combination does not exist in CKAN, and updates (including versioning) of a dataset if it already exists.
- JSON transformations: Implements a number of configurable JSON transformations
- Custom transformations: After fetching the dataset, apply a custom transformation to a target format and add the output as a format of the dataset.

For dynamic data acquisition, a single process receives the data and transforms the original schema into FIWARE data model entities (cf. Sect. 6.1), and uploads/updates entities into Orion for their querying by other processes. The procedure is divided in the following steps:

1. *Evaluate JSON path*, processor in charge of getting the id of the entity.
2. *Invoke HTTP*, processor in charge of checking if an entity with this id already exist in the Context Broker.
3. If it exists, the entity is posted to the context broker.
4. If it does not exist, a pre-processing step is carried out to add the FIWARE entity type (if missing), followed by the posting of the entity in the Context Broker.

5.2 Data from Citizens Devices

Citizens are the principal human agents of a Smart City ecosystem. As such, a fundamental component of a hybrid human-machine platform is one that enables data collection and interaction with them. The general idea is to leverage the power of devices owned by the citizens, while at the same time balancing the level of intrusiveness of the solutions, to ensure a high rate of response and not hurting the relationship between citizens and Smart Cities.

The QROWD platform includes the i-Log mobile application [29], which collects data from the user in an unobtrusive, data protection compliant and efficient way. The application can be used to generate two very diverse types of data, namely (1) streams of value-pairs generated by the devices's internal sensors, while (2) it can also collect the user input in different formats, from text to visual. The latter capability can be used to use i-Log as a channel for pushing crowdsourcing services, as seen in Fig. 3.

A simplified version of i-Log's architecture is presented in Fig. 4. The system is composed of a set of modular, logically isolated components, each one enabling a subset of the overall functionalities of the application. The modularity of the architecture allows to personalize the application and adapt it to different contexts and projects, with the need to modify only the involved components. This architecture gives i-Log a significant advantage in terms of adaptability and extensibility of its features. The four main components are:

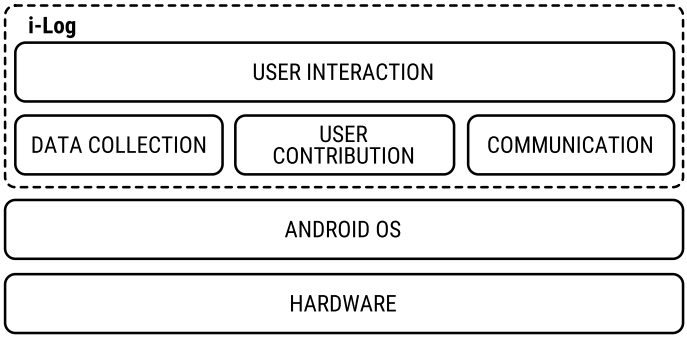

Fig. 4 i-Log mobile application architecture

- **Data collection module**: it is responsible for efficiently collecting and storing the data from the smartphone's internal sensors. The data collection has been designed to be remotely configurable in terms of (i) which sensors to use and (ii) at which frequency to collect data from them. Enabled/disabled sensors can be configured per individual tasks, within the same infrastructure. Once collected, the data are temporarily stored in compressed and encrypted logs file on the device and synchronized over WiFi whenever a connection is available.
- **User contribution module**: is responsible for collecting the user's knowledge in terms of answers to simple questions (a contribution). The knowledge can be of different types, from text, to images and to other objects that are use-case dependent, i.e., coordinates on a map. The questions are sent by a remote server as JSON objects that are then visualized on the smartphone and made available to the user.
- **Communication module**: is responsible for all the outbound and inbound connections. In more detail, it allows to contact the backend infrastructure of the application to perform operations such as registering/logging in users, to synchronize the generated logs of data and save them in a database. At the same time, it allows to receive the questions that the user has to reply to provide her own knowledge and keep her in the loop.
- **User interface module**: i-Log's main functionality is to collect data about the user while running in background on the phone. The reason for this is that the collection process must be as unobtrusive as possible. For this reason the user interface is very limited: it consists on a notification system that is always present in the notification area of the smartphone while the data collection is active. This is a mandatory requirement from a data protection point of view since the user must always been informed when someone is dealing with her data. A second notification is present whenever the user is asked to provide his knowledge. From these two notifications the user can access the actual views of the application, two menus, Settings and Contributions that allow respectively to setup the application and to have access to all the contributions.

i-Log allows to assign points to users depending on the quantity of data they have provided. These points can be used to implement the economic, reputational and hedonic values of the *Why - motivation* dimension, and by extension, the *Monetary* and prize values of the '*Why - reward*' dimension of the guidelines in Sect. 4. Points may be assigned depending on:

- The time users spent using the application throughout the day and consequently contributed with their sensors data.
- The day and time of contribution, *e.g.*, to incentivise contributions in the date and time that are more needed.
- Sensors enabled while i-Log is running. This aspect involves the combination of different factors: (1) each smartphone has a different set of sensors and this information had to be taken into account during the point assignment phase to not penalize users who don't have a sensor in their personal device, bur rather penalize those who have it and decided to turn it off. A second aspect is that (2)

not all the sensors could be disabled, *i.e.*, the accelerometer cannot while the GPS can. Finally, we should also consider that (3) not all the sensors have the same importance for all use cases, for example, accelerometer could be more valuable than position for certain type of analytics.

5.3 Citizen Challenges

Section 5.2 described how the QROWD platform supports *passive* data contributions from citizens, where the only action they need to do is to install and run the i-Log app on the background of their phones. Other types of applications require humans to take a more *active* role, like taking pictures, or answering questions about a Point of Interest, that is, the crowdsensing dimension discussed in Sect. 2. To support these use cases, the QROWD platform includes a *citizen challenges* Crowdsourcing service that could be run on top of the i-Log app. Challenges receive the following input parameters:

1. An area of interest on the challenge will take place (defined as a geospatial polygon)
2. Optionally, a set of coordinates within the area of interest where challenge participants should go to perform actions. If this set is empty, it is assumed that the purpose of the challenge is to locate something within the area of interest, that is, creating a map instead of validating it.
3. An HTML form that allows data input by the citizen, *e.g.*, coordinates using the phone capabilites, upload photos or answer questions.

Data contributions from challenges can be associated with *points* that may be redeemed for rewards, in the same way as described for passive data contributions in Sect. 5.2.

5.4 Annotations from Street-Level Imagery

The third data acquisition crowdsourcing service included with the QROWD platform is the Virtual City Explorer (VCE). It allows contributors to explore cities through street-level imagery services and provide annotations (e.g. coordinates or state) of point of interest in the map. The VCE accepts the same parameters than a citizen challenge: the area that needs to be explored by the contributors; the type of item contributors should locate; an HTML form with questions about points of interest found; and the number of contributors to assign to a given area; The VCE can be regarded as a virtual alternative over citizen challenges that is not limited to contributors physically present in the city. In turn, the VCE depends on the existence and up-to-dateness of street-level imagery.

Fig. 5 Virtual city explorer interface

Figure 5 shows a screenshot of the VCE interface from a contributor's perspective. Before exploration starts, the contributor reads the task instructions that explain its general functioning works and which are the types of objects required to locate. The contributor then starts their exploration from a random point within the area of interest. When a contributor discovers a candidate item, she is required to take three photos of it, from three different angles. In the background, the VCE triangulates the vectors of the different angles to determinate the coordinates of the item and stores them in a database. After submitting a pre-established number of items, the task ends. In case a crowdworker was the one completing the task, she is redirected to the crowdsourcing recruiting platform to receive her payment. An extensive evaluation of the VCE with paid crowdworkers is available on [16].

6 Data Models and Storage

Once data from citizens and pre-existing sources has been acquired, the next step is to have an appropriate storage in a unified data model that allows further analysis. To this end, the QROWD Platform provides: (i) An extension of the FIWARE data models for transportation to include *Citizens*, *Visitors*, and *Trips* as first-class citizens. (ii) A Big Data Storage tailored towards data collection on personal devices that facilitate compliance with data protection regulations.

6.1 Data Models

To ensure data portability for different applications including, but not limited, to Smart Cities, the QROWD Platform reuses data models developed by the FIWARE framework. We extended the FIWARE transportation data model[5] to include citizens and their data and processing contributions with three classes (*Citizen*, *Visitor*, and *Trip*) . The *Citizen* class is defined as an agent that lives or commutes in a city using the transport infrastructure. *Citizen* has two mandatory properties:

- *citizenId*: A UUID assigned to the citizen
- *citizenType*: The type of citizen according to its mobility: Resident or Commuter.

Second, the *Visitor* class as an agent that does not reside in the city. *Visitor* has two mandatory properties

- *visitorId*: A UUID assigned to the visitor
- *visitorType*: The type of visitor: Business or Tourist.

Both classes can be extended with further properties according to application needs. We describe an example of such an extension in Sect. 8.

When considering mobility data, an important concept is the one a trip a citizen makes within the city. Trips can be used to power a number of services, for example, estimate the transport mode usage in a city, understand the demand at certain times of the day, or suggest a group of citizens an alternative transportation mode. With an unified data model, different transport operators or providers can then add data to a common data shared space where analytics can be conducted.

As such, we included in our data model extension a *Trip* class, defined as follows:

- *Mode*: List of transport modes used by the trip
- *Purpose*: purpose of the trip. The concept restriction is work, school, accompanying, errands, free time, working trip, return
- *initDate*: timestamp of the start of the trip
- *endDate*: timestamp of the stop of the trip
- *startCoordinate*: coordinate of the start of the trip
- *stopCoordinate*: coordinate of the end of the trip
- *Path*: polyline representing the path of the trip
- *Multitrip* (Boolean): If the trip has multiple subtrips
- *Subtrips*: List of trip identifiers that conform a trip. Subtrips are subject to the restriction that their paths must be a subset of the path of the parent trip, that their initDate and endDate must be in the range formed by the initDate and endDate of the parent trip, and that their have a single Mode.

We also added four super-classes to facilitate further extension to related scenarios: (i) Event: events represent occurrences that can have temporal or spatial parts. We use it as superclass of *Trip* (ii) Location: represents spatial parts. We use it as superclass

[5] https://github.com/smart-data-models/dataModel.Transportation/tree/1278849c096d8ea0ceaa3e3d8d7b30d6940ab474.

of transportation areas like (iii) Structure: Physical objects representing structural entities. We use is as a superclass of the various mobility infrastructure classes in the FIWARE data models. (iv) Person: we use it as a superclass of Citizen.

6.2 Big Data Storage

Including citizens in data analytics of Smart Cities enables the leveraging of their personal mobile devices to contribute with data or to solve tasks. As such, is crucial for Smart Cities to have the technical means to manage huge amounts of data from potentially thousands of citizen's devices. Furthermore, inline with the recent entry into force of data protection laws like the European General Data Protection Regulation (GDPR), data controllers, *i.e.*, organizations that collect personal data are responsible of ensuring that any sharing of data with other organisations for further processing is consented, or that appropriate anonymisation or pseudonymisation measures have been taken.

The QROWD Platform provides a Big Data Storage component based on Apache Cassandra.[6] Cassandra offers robust support for clusters spanning multiple data centers, with asynchronous masterless replication allowing low latency operations for all clients. Linear scalability and proven fault-tolerance on commodity hardware or cloud infrastructure make it the perfect platform for mission-critical data.

To provide pseudonymisation, we included the following policies:

1. A Cassandra keyspace is associated with the data of a single user. This allows to have different consistency strategies for different users and, most importantly, will enable to isolate the data for privacy concerns. If every user's data is saved in a separate keyspace it is easier to deal with data protection requests, e.g., delete them if the user wants to uninstall the application. The anonymization is granted at this level since the name of the keyspace is a 160bit salt string generated randomly using the Secure Hash Algorithm10 (SHA-1). All data processing by other components of the platform in an hybrid workflow uses this anonymized identifier. Thus, the users' personal data is never used in this regard. Both data, the salt and any property of the citizen considered personal are stored in a disambiguation table that is accessible only by designated data controllers.
2. There is one table per query we need to reply per sensor. Since we are dealing with time series, we chose to allow querying the data by time and in some limited cases also by value. In time series most of the time a client application needs to have the values in a time interval, e.g., the accelerometer data to understand if the user is moving from 08:00 AM to 10:00 AM. In less common situations, we would like to query by value, e.g., to understand is the user previously visited a specific location.

[6] https://cassandra.apache.org/.

7 Data Integration

Data integration is a fundamental task in most value-adding data processing workflows. The general problem statement is to obtain a single coherent dataset from a set of hetergoeneous data sources. Data integration consists of data normalization, data interlinking and data fusion. The QROWD Platform supports data through the leveraging of two tools: Limes,[7] a link discovery framework for the Web of Data with time-efficient approaches for large-scale link discovery based on the characteristics of metric spaces, provides the interlinking capabilities; and Sparql-Integrate,[8] for normalization and fusion of data.

The first step of data integration is normalization, that is, represent data in a uniform way. Foremost, this involves data models (e.g. graph-based, hierarchical or tabular), and schemata (e.g. the domain of bicycle parking). However, it also affects units, lexical representations (e.g. date formats) and encodings. Traditionally, a distinction between schema and instance data is made, as there are different data integration problems and solutions related to them, for example, the approaches for aligning class hierarchies in general differs from fusing attributes of instance data. Once data has been normalized, interlinking can be applied to both schema and instance level in order to find candidate matches. These matches serve as the base for fusion. In general, the set of matches may suffer from data quality problems related to ambiguity (multiple candidates exist where only one is expected), faultiness and incompleteness. While ambiguity is resolved using conflict resolution strategies, these may itself introduce additional errors. For this reason, it makes sense to decouple the dataset of annotated candidate matches (e.g. confidence scores and provenance) as a valuable asset by itself—i.e. in isolation from the remaining fusion process. For example, a search for *Trento* on OpenStreetMap yields the city in Italy as well as a *Paseo del Trento* in Mexico.

Data fusion refers to the merge of data records of a given set of datasets for the sake of completing information and enabling resolution of conflicts. A prerequisite to fusion is schema integration such that the relevant properties of data records from multiple sources are represented uniformly. Interlinking can be applied to provide additional input to fusion processes in order to establish candidate equivalence relations between entities. Going back to the *Trento* search example above, while the remaining fusion process simply adds the geo-coordinates of the match marked as correct to the final dataset, the dataset of candidate matches allows for quick verification and revision.

The QROWD platform makes use of Semantic Web technologies RDF and SPARQL. With RDF, we can represent schema and instance data uniformly in a graph-like structure often referred to as a *knowledge graph*, which enables retrieval and manipulation of data stored in with SPARQL queries. By relying on the linked data principles this workflow keeps minimizes the necessary groundwork. All data transformations, which achieve the data integration tasks are defined by SPARQL

[7] http://aksw.org/Projects/LIMES.html.

[8] https://github.com/QROWD/SparqlIntegrate.

queries and/or RDF config files. This leads to two main benefits: the user only has to be fluent in these two technologies and this workflow can be integrated in any other dataflow where SPARQL processors can be added. Traditionally, in the relational database world, fusion processes are specified using sequences of SQL statements which implement domain specific rules. Although the SPARQL standard does not provide a feature set as rich as that of SQL (dialects), the basic principles can be applied to SPARQL nonetheless. For example, DBpedia recently introduced a very similar workflow for data fusion in [7], where the set of annotated candidate matches is referred to as the *PreFuse* dataset.

Following the same pattern described in Sect. 4 for crowdsourcing services, the integration of Limes and Sparql-Integrate into the QROWD Platform was achieved by writing corresponding Apache NiFi processors.[9] Recall from Sect. 3 that the QROWD Platform is based on Apache NiFi, which is designed to automate the flow of data between software systems. A NiFi dataflow is defined by a network of *processors*, where *flow-files* are used to pass data along connections, and may have multiple ingoing and outgoing connections.

Sparql-Integrate is a tool developed in QROWD which leverages SPARQL for the integration of heterogeneous data. The SPARQL specification itself allows for extension functions but also notes the risk of limited interoperability.[10] For QROWD, we chose the Apache Jena Semantic Web framework as the basis for our own SPARQL extensions. Our extensions exploit this framework's plugin system, making it possible to easily integrate them into any other Jena-based software project. The SPARQL-Integrate project is comprised of two libraries and two interfaces:

- A standalone SPARQL extension library for Jena with support for data formats (XML, JSON, CSV), HTTP requests, and file system access. Naturally, some of these extensions are meant only for internal processing and should not be exposed in e.g. public SPARQL endpoints due to their potential for abuse.
- A small standalone core library with additional functionality, especially a parser for documents holding multiple SPARQL statements, and a corresponding processor that gives control over the output.
- A command line interface for processing files of RDF data and SPARQL queries. Additionally it supports launching an embedded SPARQL endpoint with HTML frontend and the provided extensions.
- An Apache NiFi processor integration.

We wrapped Sparql-Integrate into a processor which can take either data or its configuration as content of a input flow-file. With this the Sparql-Integrate processor is able to access files over HTTP, locally on the file system or as the content of a flow-file. With the most common file formats CSV, XML and JSON supported it is possible to create an ontology aligned graph out of these sources with one ore more SPARQL queries.

[9] https://github.com/QROWD/nifi-sparql-integrate-bundle.

[10] https://www.w3.org/TR/sparql11-query/#extensionFunctions.

LIMES is a link discovery frameworks for the Web of Data that identify similar entities as well as duplicates in Web datasets. LIMES can execute so-called link specifications, which contain heuristics for the similarities of entities in datasets. Those specifications can be either created manually or via machine-learning techniques. Human feedback is required to assess and maximize the precision and recall of these link specifications as well as resultant output. We also wrapped Limes into a Apache NiFi processor, the processor accepts a configuration file as input and returns a list of found links between entities. The configuration specifies the location of the datasets, which entities based on which specific properties should be linked and what kind of metric should be used as a distance measure.

Nifi processors that wrap Limes and Sparql-Integrate can be used within the QROWD Platform with a simple "drag and drop". While it is possible to engineer a data integration workflow with the QROWD Platform only, we also developed a complementary offline workflow. To deal with the increasing complexity of Sparql-Integrate queries as tasks become more complex, the complementary workflow[11] treats those files as source code, which enables syntax checking, autoformat, completion and version control. When finished, these files can be used to configure the QROWD platform processors.

With the addition of Sparql-Integrate and Limes processors, it is possible to create arbitrary data integration workflows within the QROWD Platform. In Sect. 8 we will see the processors in action for solving the urban auditing problem.

8 Use Cases

In this section we show how we used the QROWD platform to develop two hybrid human-machine data flows: one to generate and manage mobility infrastructure data, and another to estimate modal split. We report on the piloting and evaluation of both applications in the city of Trento, Italy.

8.1 Generating and Managing Mobility Infrastructure Data

Accurate information of current mobility infrastructure is crucial for the implementation of mobility policies. However, records may be incomplete due to certain items being installed and owned by private parties, or due to digitization errors. Sending municipality employees to scout an area or regularly check known infrastructure does not scale in area and is expensive. A smarter alternative would be to involve citizens to help with the task. In this section, we describe a hybrid human machine workflow we deployed on a live setting in the city of Trento, Italy, for generating and curating a map of bike rack locations for the Limited Traffic Zone of Trento. In

[11] https://github.com/QROWD/link-discovery-and-data-fusion.

Table 3 Design guideline applied to bike rack map collection using the VCE

What	**Who**	**Why—motivation**	**Why—reward**
Collection	Crowdworkers	Economic	
How	**Required skill**	**Required device**	**Device constraint**
Collection	Visual recognition	PC	None
Interaction limit	**Question delay**	**Resolution delay**	
5	Medium	Medium	

the following, we describe each of the steps we follow together with the Smart City managers in Trento.

Acquisition

The municipality of Trento had an initial dataset of 39 bike racks in the area of interest, each one including the type of bike rack (single sided or double sided), the name of the street is located and the capacity. The Smart City managers wanted to know id the dataset was complete in the sense that all bike racks in the area were included, and accurate, in the sense that all properties of each bike rack in the dataset were correct. Using the tools described in Sect. 5.1, we acquired this dataset into the CKAN repository.

A second bike rack dataset was openly available from OpenStreetMaps. Two volunteers had contributed the locations and properties of 59 bike racks in the same area of interest than the Municipality dataset. However, 36 bike racks were missing at least one of the type or capacity properties. We acquired the dataset to the CKAN repository, but before further analysis, we decided to generate a new dataset taking advantage of the availability of recent street-view level imagery in Trento.

Generation We used the Virtual City Explorer tool described in Sect. 4 to create a crowdsourcing task to collect bike racks from the Google Maps street-view imagery of the area of interest defined by the municipality. Table 3 shows the application of the design guidelines to this task. We deployed the task on a crowdsourcing platform and recruited 25 crowdworkers that mapped 44 bike racks.

Interlinking and Fusion

Using the machine components described in Sect. 7 we ran an interlinking process between the three acquired datasets based on the bike rack's geographical coordinates. The output is a fused dataset where bike racks from different datasets judged to be the same are merged into a new *representative* entity that aggregates all properties from its parents in a set.

Figure 6 shows a visualisation of the fused dataset. Bike racks from different datasets considered to be the same are shown as groups of circles with the same color. White circles represent bike racks that were not linked with any other. Shapes within the circles encode from which dataset the bike rack comes from: squares represent bike racks from the data generated with the VCE; triangles from OpenStreetMaps, and crosses bike racks from the Municipality dataset. Datasets for each

Fig. 6 Map of bike rack clusters

source (OpenStreetMaps, Municipality and VCE) and the fused dataset are openly available in the Zenodo repository.[12]

Curation

To validate the location and properties of the bike racks in the interlinked map, a human needs to verify them. We modeled this as a *Citizen Challenge* (Sect. 5.3) using as input the area of interest and the interlinked map. Table 4 shows the instantiation of the crowdsourcing guidelines given in Sect. 4 for this task.

Citizens were asked to go to one of the locations in the interlinked map and confirm (Fig. 7 (center)) if the bike rack is there using a form with the following input fields:

1. Their location, taken from the device's GPS by the i-Log app. This step is needed to confirm that the citizen is on the point featured in the map. Location was only considered valid if the measured GPS accuracy provided by the app was below $10.0m$.

[12] https://doi.org/10.5281/zenodo.3574485.

Table 4 Design guideline applied to bike rack verification challenges

What	**Who**	**Why—motivation**	**Why—reward**
Verification and validation	Citizens	Economic, hedonic	Prizes
How	**Required skill**	**Required device**	**Device constraint**
Collection	Visual recognition, Physical	Mobile phone	Bandwidth
Interaction limit	**Question delay**	**Resolution delay**	
None	Challenge duration	Challenge duration	

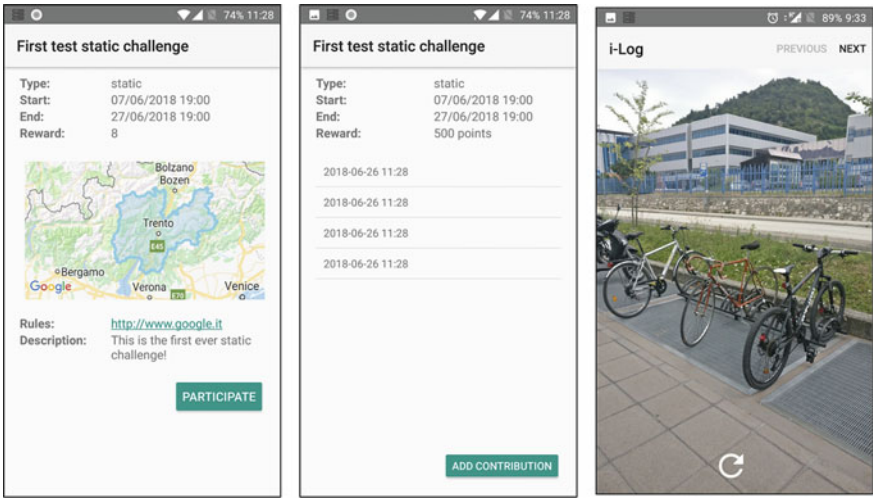

Fig. 7 Three i-Log interfaces, for (left) a user to decide if accept to participate a challenge, (middle) a user contribute with a new item detection, and (right) a user taking a picture of a new item discovered

2. Reply Yes/No to the question *Is the bike rack still here?* If the selected answer is No, then the contribution is submitted.
3. If the answer to the previous question was Yes, provide a photo of the bike rack (Fig. 7 (right)).
4. Answer the question *What kind of bike rack do you see?* The answer is picked from example pictures of three different types of bike racks.
5. Answer the question: *What is the capacity of the bike rack?*
6. Answer the question: *How many available spots does the bike rack have?*

12 citizens of age 18–25, students of the University of Trento, accepted to participate in the challenge over a one week period. For each verified bike rack, participants received 5 points, for each 20 points accumulated, the participant had the right to a 5 euros voucher to exchange for phone credit. All participants validated at least 4 bike racks, and all but one of the bike racks on the input map were verified as as existent

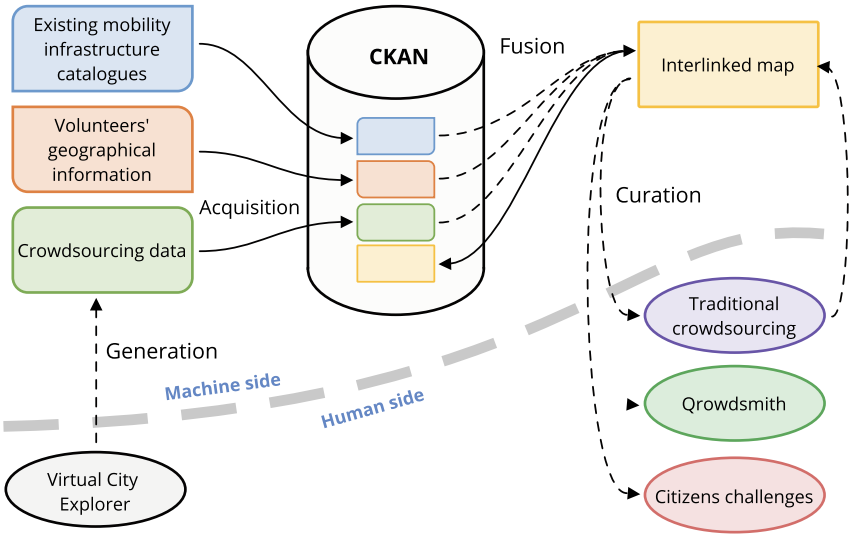

Fig. 8 High level description of the hybrid human workflow for completing mobility infrastructure

or not. The one bike rack missed by the challenge was checked by a municipality employee and found to be within a private ground, therefore, removed from the final result.

On the light of the good results with bike racks, we decided to invite the same participants to a second challenge to collect locations, types and pictures of special parking spots (disabled, taxi ranks, freight load/unload). These type of parking spots are challenging to collect with the VCE due to the fact that available pictures on street-view level imagery may have a vehicle on them, impeding their identification. The input form was comprised of the following three fields:

1. Take picture of parking spot
2. Share location using phone capabilities
3. Answer the question *What type of parking spot is this, disabled, taxi rank or freight?* Examples of each type were provided in the interface for clarity.

The new challenge ran for one week and allowed the collection of a dataset containing 401 special parking spots (Fig. 8).

8.2 Modal Split Surveys

Modal split is a fundamental indicator for understanding how citizens use various means of transport. It is defined as the percentage of citizens using a particular

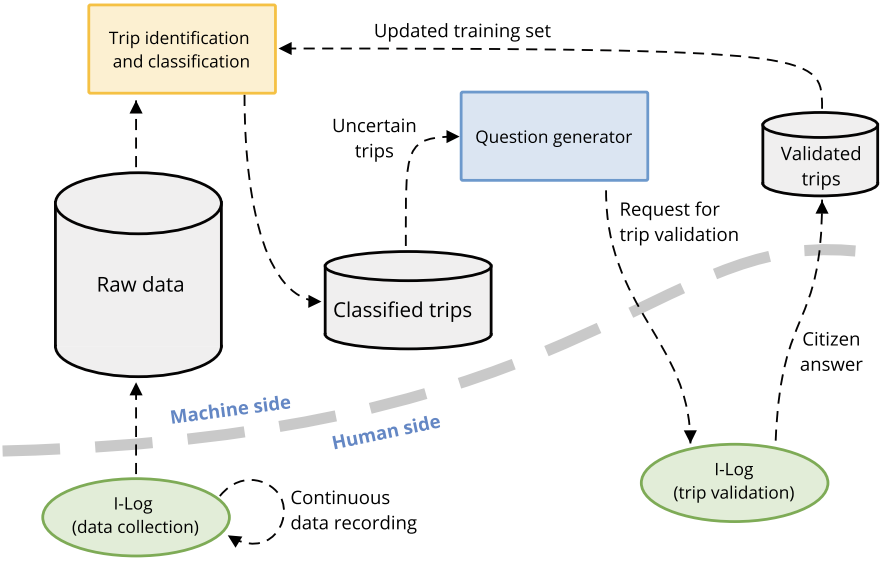

Fig. 9 High level description of the hybrid human workflow for trip collection from Citizen's devices

mode of transportation for their travel in a specified time period (e.g., 30% car, 30% bus, 20% bike, 20% walking). It is also an important input for designing and evaluating mobility policies. For example, if a large number of car trips are detected towards a certain district, the municipality can then devise a policy to encourage other transportation modes. Furthermore, the same measurement can be made again focused on trips to that district to evaluate the effectiveness of the policy.

Traditionally, modal split is estimated through travel surveys, where citizens either fill a paper form, or provide answers by telephone to an operator, with details of their trips during a certain period of time. This is quite expensive and time consuming, greatly limiting the number of times the modal split can be measured. An interesting approach is to use citizen's mobile phones to automate the application of the survey, and use data analytics on collected data to "fill the form" as automatically as possible, asking the user only to confirm trips where we are not confident enough about the result provided by the machine. Figure 9 shows a high level overview of the data flow for collecting trips from citizens. We describe below how we implemented it to satisfy the particular needs of the Municipality of Trento (MT) with the help of the QROWD platform.

First, we extended the Citizen data model described in Sect. 6.1 with the demographic properties required by MT to filter and aggregate modal split. Table 5 describe the added attributes. The data model was loaded into QROWDDB, and CRUD operations on Citizens and Trip were configured into the QROWDDB (Sect. 6)

For connected vehicles and public transport, it is relatively easy to generate trip data that is complete, with accurate start, stop points and a correct transport mode

Table 5 Attributes added as an extension to Citizen data model for modal split survey application

Attribute	Description
Occupation	Principal work activity
numberCohabitants	Number of people that live in the same house
numberVehicles	Total number of vehicles available to all cohabitants
preferredMode	Preferred transportation mode
WorkSector	Sector where citizen works
HomeSector	Sector where citizen lives
Age	Age of citizen
Gender	Gender of citizen
Email	e-mail address
streetAddress	Address
drivingLicense	Type of driving license owned, if any

label. However, contributions from mobile devices simply push data upstream and do not have the capabilities to convert raw data into trips. Within the QROWD Platform we developed a Transport Mode Detection component, that uses Machine Learning models to processs GPS and accelerometer time series to (i) Separate a GPS trace into (multimodal) trips by detecting start and stop points. (ii) For each trip, infer the transport mode of each leg.

However, data from personal devices is often noisy and/or sparse, and training data for specific transport modes in the specific topological and traffic conditions of a city may not be available, leading to inaccurate trip classification. To solve this, we put citizens in-the-loop by providing a *Trip Update Interface* as a Crowdsourcing Service (cf. Sect. 4) to allow the confirmation and amendment of the trips inferred by the machine.

The overall data and workflow is shown in Fig. 10. We assume the sensor data captured by the citizens' device is available in the QROWDDB component. For each citizen, we analyse whole-day GPS trajectories. First we preprocess them to extract the actual traveling segments. After removing outliers, unsupervised machine learning techniques like space-time clustering are applied to find stop points, e.g. when a citizen only moved inside a building where many captured GPS positions are in the near vicinity, thus building a point cluster, as shown in Fig. 11. The actual traveling segments, or *'trips'*, are then the movements between clusters.

To detect transportation modes, we apply supervised machine learning techniques trained on *labeled data*, *i.e.*, example trips where we knew the correct transportation modes. For training, the same preprocessing was performed. We use supervised machine learning approaches that can be grouped into two categories, *'numeric'* and *'symbolic'*. The numeric machine learning approaches work on features derived from the accelerometer data streams we captured on the users' smart phones. Here we use several classification algorithms from the Scikit-Learn [20] machine learning

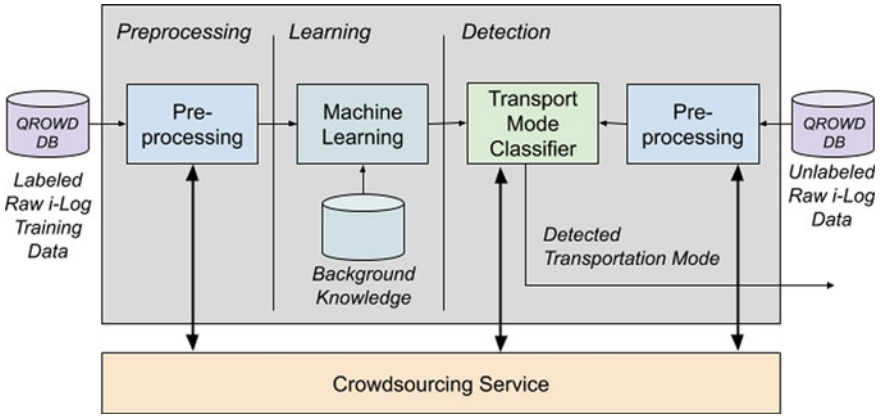

Fig. 10 Data and workflow of the transport mode detection component

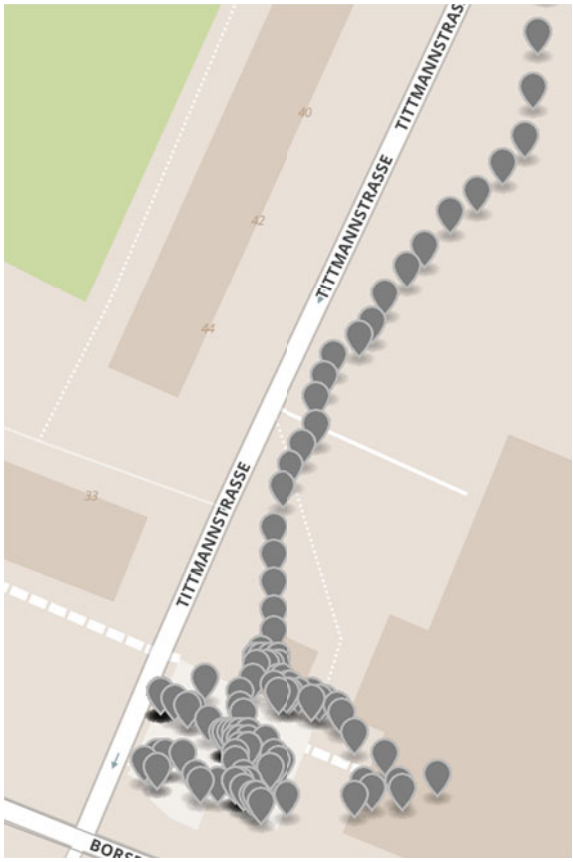

Fig. 11 Part of an example GPS trajectory showing a cluster at a stop inside a shopping mall

library.[13] The symbolic machine learning algorithms make use of the citizens' trip trajectories and symbolic background data representing the traffic infrastructure and further geographic information of the model region. This background data is represented by means of the Resource Description Framework (RDF)[14] and the Web Ontology Language (OWL).[15] As supervised machine learning software for learning OWL class expressions that describe and serve as classifiers to distinguish, e.g. bus trips from non-bus trips, we used the DL-Learner framework[16] [1]. To be able to infer class expressions that reflect distinctive spatial relations, e.g., that a trip was probably made by bus if it started and ended *near* a point of interest of type '*bus stop*' and *went along* a line feature which represents a known *bus route*, we extended the OWL reasoning components of the DL-Learner to enable 'spatial reasoning'. This spatial reasoner component is able to make implicit knowledge stored in the background knowledge base *explicit* and thus usable in OWL class expressions. We set up an RDF vocabulary of 'virtual' spatial RDF properties covering the relations from the Region Connection Calculus (RCC) [3] and further relations that seemed suitable for the task of expressing characteristic features of the different transportation modes. Those spatial properties are inferred by means of the spatial coordinates attached to spatial entities in the knowledge base. A simple example would be the *near* property, where an assertion *a near b* is inferred whenever the distance between the geographical coordinates of *a* and *b* is less than, e.g. 10 m (where the actual value can be configured). Taking into account that GPS trajectories recorded on general purpose commodity hardware like smart phones usually are not 100% accurate, the spatial reasoner also needs to handle a certain degree of fuzziness. Figure 12 exemplifies this for the *runs along* property.

Here, the spatial reasoner extension we developed returns all road segments from LinkedGeoData[17] [26] on which the given GPS trajectory runs along even though the respective trajectory segments do not exactly match the road segments.

All trained classifiers are consolidated in an overall 'meta' transportation mode classifier which may chose a classification outcome, e.g., from that classifier that could achieve the highest confidence. However, both start-stop detection and trip classification may fail, e.g. due to very sparse GPS trajectories or lack of enough training data. The citizen is the only one that knows exactly what the itinerary was, therefore, to put them in the loop, we designed a *Trip Update Interface* (TUI) as a crowdsourcing service to allow the confirmation and amendment of the trips inferred by the machine. Table 6 shows the design guidelines applied to the TUI. The motivation was set as partly altruistic (desire to collaborate with the municipality) and partly economic (winning a prize for contributing data). To avoid annoying users, the interaction limit was set to one question per trip. To avoid issues of users forgetting

[13] https://scikit-learn.org.

[14] https://www.w3.org/TR/rdf11-primer/.

[15] https://www.w3.org/TR/owl2-overview/.

[16] http://dl-learner.org/.

[17] http://linkedgeodata.org.

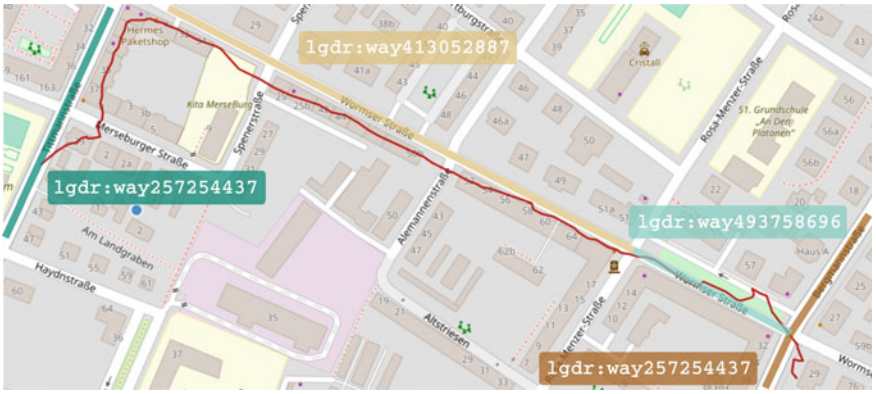

Fig. 12 Bike ride trajectory exemplifying the spatial relation *runs along*; the lgdr prefix of the results resources resolves to http://linkedgeodata.org/triplify/

Table 6 Design guideline applied to the trip verification task

What	**Who**	**Why—motivation**	**Why—reward**
Verification and validation	Trento citizens	Economic, altruistic	Prizes
How	**Required skill**	**Required device**	**Device constraint**
Collection	Visual recognition	Mobile phone	Battery—bandwidth
Interaction limit	**Question delay**	**Resolution delay**	
1 question per trip	24 h	72 h	

the details of a trip, the question delay was fixed to 24 h. Resolution delay was set to 72h.

We implemented a *decision component* that for each trip detected by the TMD generated one instance of the TUI. Both the classifier and the decision component were executed every morning on data collected from the previous day, according to the value of the question delay property in the guidelines. In case no sensor data had been received, a *failsafe* question asking the reason why no data had been received (app failure or conscious decision of not submitting data) and providing the user with a blank map where they could manually input their trips if they wished to do so. As *deployment manager*, we implemented a simple connector that encapsulated the instantiated TUI into an i-Log question and called the i-Log API. As questions about a trip can only be answered by the citizen that made it, we did not implement any aggregation metric.

The TUI receives as input a trip, and generates an HTML5 + JavaScript responsive interface (for which an example is shown in Fig. 13) that enables amendment as follows:

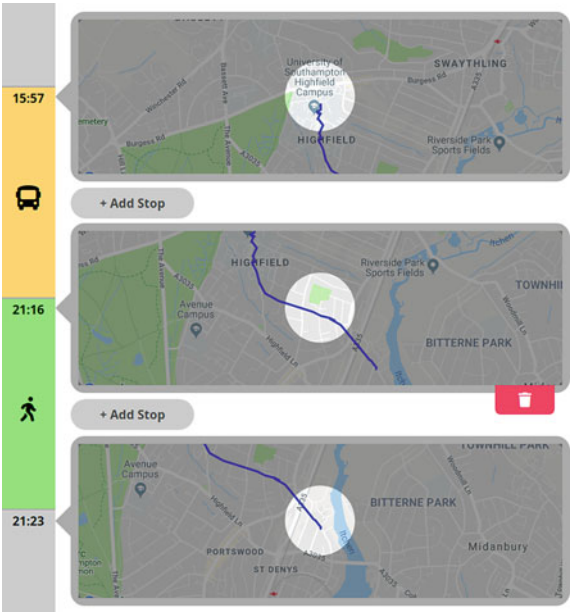

Fig. 13 The trip update interface

1. Each approximate start and stop point is shown as a highlighted circle on a section of the map. A user can drag the circle on the map to amend the location any of the points.
2. Each pair of start/stop points is linked to an icon representing the detected transport mode used between the two points in question. In the example, bus was detected between the start point and the first stop point, while walking was detected between the first and second stop points. A user can tap on the icon to select another transportation mode.
3. Finally, an user can add or remove intermediate stops using the add stop button and the rubbish bin icon.

The TUI outputs a trip with the amended start/stop points and transport modes (if any). Amended trips are considered 'ground truth'. As such, we can also use them for bootstrapping the training of the classifiers by always asking for confirmation of all the trips and periodically re-training the classifiers. Once they achieve a certain accuracy, one can only ask for amendment of those trips with a confidence level below a certain threshold. Model re-training can be configured as an offline process, or as a step of the data flow, based on a certain condition, *e.g.*, number of confirmed/amended trips collected.

To collect data from citizens, we used the i-Log application described in Sect. 5.2. Inline with the data minimisation principle of the European General Data Protection Regulation (applicable as the use case is within Europe), we configured i-Log to only collect accelerometer and GPS data, as the only streams required by TMD.

A pilot was run in the first week of October 2019 with 149 participants. 44 participants submitted either sensor phone data or a failsafe question every day of the week, 65 at least one day of the week and 40 abandoned the experiment without contributing any data. Users could provide qualitative feedback through email after the end of the pilot. 18 people chose to do so, from which we highlight the following comments.

1. In some phones, the impact of sensor data collection on the device's battery was perceived as high, prompting users to uninstall the app. Lesson learned: Further improvements in the engineering of the app would be required for going into production.
2. When the automatically inferred trips were very different from real trips, it was hard to update it to reflect reality. Lesson learned: further research needs to be conducted on the user experience of interfaces to update trips, especially in mobile devices.
3. Users that were less skilled with their phones considered the interface too complicated, leading to worries about providing wrong data. Lesson learned: the rationale of using mobile devices for modal split surveys is to take advantage of their ubiquity and the assumption that embedding questions about the data on the same devices would increase the number and quality of the answers. However, for some demographics this needs to be balanced with the user experience. A possible way forward is allowing trip update on a PC or tablet.

In terms of the experience of the Municipality, we identified as main pain point the need to run a helpdesk to support citizens with questions and to resolve issues with. This need partially offsets the savings of this approach with respect to the phone surveys that it intends to replace. Nevertheless, the i-Log approach was estimated to be 20% less expensive than an equivalent phone survey. We expect this percentage to increase with improvements on the battery usage of the app and in the user experience of the Trip Update Interface.

9 Summary and Conclusion

In this chapter, we presented the QROWD Platform, a collection of crowdsourcing enabled integrations within a FIWARE-compliant architecture to create hybrid human-machine data processing workflows. The platform provides a framework for helping Smart City managers and their IT teams with the design and implementation of human computation tasks and citizen sensing as *Crowdsourcing services* such that they can be integrated with machine processes.

We demonstrated QROWD's capabilities by describing how we use it to develop two hybrid human-machine workflows to solve two real problems in the municipality of Trento, Italy: locating mobility infrastructure, and implementing modal split surveys leveraging mobile phone sensor data and citizen's feedback. The approach

was very successful for the first use case. For the second, there is still room for improvement for their large scale implementation: better engineering of the sensor data collection application to reduce impact on phone battery, and providing more interface options to citizens for validating trip classifications provided by the machine.

As the amount of data available to Smart Cities grows, there will be a need for purposeful analytics for operational managers and decision makers, in addition to approaches that enable citizen inclusion towards more human-centric cities. QROWD paves the way towards this end, and at the same time provides tools for leveraging other types of human contributions, such as those available from crowd-working platforms. Future work will be focused on two areas: first, incorporate advanced data privacy mechanisms that allow citizens fine-grained control on what type of analysis they allow on their data; second, when integrating data from different sources from different providers, the question about how to price queries on that data in the context of an analytic process arises. This also has implications for monetary or prize rewards for citizens contributing data.

Acknowledgements Research on this paper was supported by the QROWD project, part of the Horizon 2020 programme under grant agreement 732194. We also acknowledge the Smart City managers of the Municipality of Trento.

References

1. Bühmann, L., Lehmann, J., Westphal, P.: DL-Learner—a framework for inductive learning on the semantic web. J. Web Semantics **39**, 15–24 (2016)
2. Calderoni, L., Magnani, A., Maio, D.: IoT Manager: An open-source IoT framework for smart cities. J. Syst. Architect. **98**, 413–423 (2019). https://doi.org/10.1016/j.sysarc.2019.04.003. https://www.sciencedirect.com/science/article/pii/S1383762118306520
3. Cohn, A.G., Bennett, B., Gooday, J., Gotts, M.M.: Qualitative spatial representation and reasoning with the region connection calculus. GeoInformatica **1**(3), 275–316 (1997)
4. Costa, D.G., Damasceno, A., Silva, I.: CitySpeed: a crowdsensing-based integrated platform for general-purpose monitoring of vehicular speeds in smart cities. Smart Cities **2**(1), 46–65 (2019). https://doi.org/10.3390/smartcities2010004. https://www.mdpi.com/2624-6511/2/1/4
5. Delmastro, F., Arnaboldi, V., Conti, M.: People-centric computing and communications in smart cities. IEEE Commun. Mag. **54**(7), 122–128 (2016). https://doi.org/10.1109/MCOM.2016.7509389
6. Doran, D., Gokhale, S., Dagnino, A.: Human sensing for smart cities. In: Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 1323–1330. ASONAM '13, Niagara, Ontario, Canada. ACM, New York, NY, USA (2013). https://doi.org/10.1145/2492517.2500240
7. Frey, J., Hofer, M., Obraczka, D., Lehmann, J., Hellmann, S.: DBpedia FlexiFusion the Best of Wikipedia > Wikidata > Your Data. In: Ghidini, C., Hartig, O., Maleshkova, M., Sviatek, V., Cruz, I., Hogan, A., Song, J., Lefrancois, M., Gandon, F. (eds.) The Semantic Web - ISWC 2019. pp. 96–112. Lecture Notes in Computer Science, Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-30796-7_7
8. Garcia-Retuerta, D., Chamoso, P., Hernández, G., Guzmán, A.S.R., Yigitcanlar, T., Corchado, J.M.: An efficient management platform for developing smart cities: solution for

real-time and future crowd detection. Electronics **10**(7), 765 (2021). https://doi.org/10.3390/electronics10070765. https://www.mdpi.com/2079-9292/10/7/765
9. Gooch, D., Wolff, A., Kortuem, G., Brown, R.: Reimagining the role of citizens in smart city projects. In: Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers, pp. 1587–1594. UbiComp/ISWC'15 Adjunct, Osaka, Japan. ACM, New York, NY, USA (2015). https://doi.org/10.1145/2800835.2801622
10. Gopinath, B., Kaliamoorthy, M., Ragupathy, U.S., Sudha, R., Nandini, D.U., Maheswar, R.: State-of-the-art and emerging trends in internet of things for smart cities. In: Maheswar, R., Balasaraswathi, M., Rastogi, R., Sampathkumar, A., Kanagachidambaresan, G.R. (eds.) Challenges and Solutions for Sustainable Smart City Development, pp. 263–274. EAI/Springer Innovations in Communication and Computing, Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-70183-3_12
11. Gutiérrez, V., Amaxilatis, D., Mylonas, G., Muñoz, L.: Empowering citizens toward the co-creation of sustainable cities. IEEE Internet Things J. **5**(2), 668–676 (2018). https://doi.org/10.1109/JIOT.2017.2743783
12. Kong, X., Liu, X., Jedari, B., Li, M., Wan, L., Xia, F.: Mobile crowdsourcing in smart cities: technologies, applications, and future challenges. IEEE Internet Things J. **6**(5), 8095–8113 (2019). https://doi.org/10.1109/JIOT.2019.2921879
13. Kuru, K., Ansell, D.: TCitySmartF: a comprehensive systematic framework for transforming cities into smart cities. IEEE Access **8**, 18615–18644 (2020). https://doi.org/10.1109/ACCESS.2020.2967777
14. Liu, J., Shen, H., Narman, H.S., Chung, W., Lin, Z.: A survey of mobile crowdsensing techniques: A critical component for the internet of things. ACM Trans. Cyber-Phys. Syst. **2**(3) (2018). https://doi.org/10.1145/3185504
15. Lécué, F., Tallevi-Diotallevi, S., Hayes, J., Tucker, R., Bicer, V., Sbodio, M.,Tommasi, P.: Smart trac analytics in the semantic web with STAR-CITY: Scenarios,system and lessons learned in Dublin City. J. Web Semantics 27–28, 26–33 (2014). https://doi.org/10.1016/j.websem.2014.07.002. http://www.sciencedirect.com/science/article/pii/S157082681400050X
16. Maddalena, E., Ibáñez, L.D., Simperl, E.: Mapping points of interest through street view imagery and paid crowdsourcing. ACM Trans. Intell. Syst. Technol. **11**(5), 1–28 (2020). https://doi.org/10.1145/3403931
17. Maddalena, E., Ibáñez, L.D., Simperl, E., Gomer, R., Zeni, M., Song, D., Giunchiglia, F.: Hybrid human machine workflows for mobility management. In: Companion Proceedings of The 2019 World Wide Web Conference, pp. 102–109. WWW '19. ACM, New York, NY, USA (2019). https://doi.org/10.1145/3308560.3317056
18. Malone, T., Laubacher, R., Dellarocas, C.: The collective intelligence genome. IEEE Eng. Manage. Rev. **38**(3), 38–52 (2010). https://doi.org/10.1109/EMR.2010.5559142. http://ieeexplore.ieee.org/document/5559142/
19. Olariu, S.: Vehicular crowdsourcing for congestion support in smart cities. Smart Cities **4**(2), 662–685 (2021). https://doi.org/10.3390/smartcities4020034
20. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, È.: Scikit-learn: machine learning in python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
21. Pfisterer, D., Romer, K., Bimschas, D., Kleine, O., Mietz, R., Truong, C., Hasemann, H., Kröller, A., Pagel, M., Hauswirth, M., Karnstedt, M., Leggieri, M., Passant, A., Richardson, R.: SPITFIRE: toward a semantic web of things. IEEE Commun. Mag. **49**(11), 40–48 (2011). https://doi.org/10.1109/MCOM.2011.6069708
22. Puiu, D., Barnaghi, P., Tönjes, R., Kümper, D., Ali, M.I., Mileo, A., Parreira, J.X., Fischer, M., Kolozali, S., Farajidavar, N., Gao, F., Iggena, T., Pham, T., Nechifor, C., Puschmann, D., Fernandes, J.: CityPulse: large scale data analytics framework for smart cities. IEEE Access **4**, 1086–1108 (2016). https://doi.org/10.1109/ACCESS.2016.2541999

23. Quinn, A.J., Bederson, B.B.: Human computation: a survey and taxonomy of a growing field. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1403–1412. CHI '11. ACM, New York, NY, USA (2011). https://doi.org/10.1145/1978942.1979148
24. Santos, F.A., Silva, T.H., Braun, T., Loureiro, A.A.F., Villas, L.A.: Towards a sustainable people-centric sensing. In: 2017 IEEE International Conference on Communications (ICC). pp. 1–6 (2017). https://doi.org/10.1109/ICC.2017.7997223
25. Smart, P., Simperl, E., Shadbolt, N.: A Taxonomic framework for social machines. In: Miorandi, D., Maltese, V., Rovatsos, M., Nijholt, A., Stewart, J. (eds.) Social Collective Intelligence: Combining the Powers of Humans and Machines to Build a Smarter Society, pp. 51–85. Computational Social Sciences, Springer International Publishing, Cham (2014)
26. Stadler, C., Lehmann, J., Höffner, K., Auer, S.: Linkedgeodata: a core for a web of spatial open data. Semantic Web J. **3**(4), 333–354 (2012)
27. Sánchez, L., Gutiérrez, V., Galache, J.A., Sotres, P., Santana, J.R., Casanueva, J., Muñoz, L.: SmartSantander: experimentation and service provision in the smart city. In: 2013 16th International Symposium on Wireless Personal Multimedia Communications (WPMC), pp. 1–6 (2013)
28. Vakali, A., Dematis, I., Tolikas, A.: Vol4all: A volunteering platform to drive innovation and citizens empowerment. In: Proceedings of the 26th International Conference on World Wide Web Companion. pp. 1173–1178. WWW '17, Perth, Australia. Companion, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland (2017). https://doi.org/10.1145/3041021.3054712
29. Zeni, M., Zaihrayeu, I., Giunchiglia, F.: Multi-device activity logging. In: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication, pp. 299–302. UbiComp '14 Adjunct, ACM, New York, NY, USA (2014). https://doi.org/10.1145/2638728.2638756