

# Implementing big data lake for heterogeneous data sources

Hassan Mehmood\*, Ekaterina Gilman\*, Marta Cortes\*, Panos Kostakos\*, Andrew Byrne\*, Katerina Valta\*\*, Stavros Tekes\*\*, Jukka Riekki\*

\*University of Oulu, Finland \*Dell EMC, Ireland \*\*Draxis Environmental S.A, Greece  
{firstname.lastname}@oulu.fi \*{firstname.lastname}@dell.com \*\*katvalta, stavros@draxis.

**Abstract**— Modern connected cities are more and more leveraging advances in ICT to improve their services and the quality of life of their inhabitants. The data generated from different sources, such as environmental sensors, social networking platforms, traffic counters, are harnessed to achieve these end goals. However, collecting, integrating, and analyzing all the heterogeneous data sources available from the cities is a challenge. This article suggests a data lake approach built on Big Data technologies, to gather all the data together for further analysis. The platform, described here, enables data collection, storage, integration, and further analysis and visualization of the results. This solution is the first attempt to integrate a diverse set of data sources from four pilot cities as part of the CUTLER project (Coastal urban development through the lenses of resiliency). The design and implementation details, as well as usage scenarios are presented in this paper.

*Keywords*- data lake; Big Data; Smart City; data analysis

## I. INTRODUCTION

Rapid and unstructured urbanization poses a major burden to the well-being, cohesion, and safety of modern cities, creating an urgent need for evidence-based decision making to drive sustainable development goals [1]. Because digital technologies and smart city solutions hold a great promise in assisting with some of these challenges, recent years have seen a booming business interest in the IoT ecosystem throughout Europe, Asia and America. According to the latest forecasts, the global market for solutions and services created by smart cities is expected to grow to approximately USD \$2 Trillion by 2025 [2].

As a result, the use of smart city platforms in improving public administration and everyday life is gaining attention in both research and policy circles [3]. Current trends in IoT adoption and the ubiquity of connectivity and mobile computing provides tremendous opportunities to provide insights and develop utility services [4]. Evidently, the proliferation of data is one of the key aspects that can enable a shift from intuition-based to evidence-based decision making that can improve the resilience of modern cities. Within this highly dynamic environment, it is estimated that by 2020 Europe will have the largest number of smart city project investments globally [2].

CUTLER [5] aims to investigate how data can help in creating the evidence for urban decision-making. Particularly, the project aims to generate and provide evidence-based solutions for policy decision making related to the coastal urban development by using Big Data solutions.

CUTLER builds the research with four real pilots conducted in Cork, Antwerp, Antalya, and Thessaloniki. These pilots have their own targets, however, all of them rely on data available from city sensing infrastructures, statistical information, and user contributed content. Analytical solutions will be implemented to measure the economic activity, evaluate the environmental impact and social consequences of implemented policies. This constitutes the evidence to inform, monitor, evaluate, and revise the decisions made by policy planners. Respectively, CUTLER relies on heterogeneous data, coming from diverse sources. Hence, solutions are required to collect, store, integrate, and analyze such data.

Therefore, there are a number of challenges to be solved. The first one is how to manage the process of identifying and curating the various data sources? Then, how to organize the process of acquiring, storing, integrating, analyzing, and managing of the data coming from those identified data sources? Finally, how to address structural and semantic changes in data and availability of data sources with the time? This article proposes an approach to tackle some of these challenges. Namely, we suggest a process to identify and manage various data sources. In addition, we present the first version of the platform enabling data acquisition, storage, and analysis. Since CUTLER utilizes various kinds of data sources, the approach presented here is suitable for other urban projects relying on diverse data. This first iteration of the platform is used to start collecting the data, to analyze the technical challenges, and to build a roadmap for new platform features and functionality. More details on CUTLER architecture can be found from [6], [7].

Hence, the contribution of this article includes: 1) process proposal for managing the collection of data sources from diverse stakeholders; 2) real-world implementation of data acquisition and storage for heterogeneous data sources; 3) analysis of implemented solution with respect to design and technological solutions.

The rest of the paper is structured as follows: Section II reviews the related work. Section III focuses on data sources management from heterogeneous data stakeholders. Section IV provides requirements and data collection, storage, and processing platform proposal. Section V describes its implementation. Section VI discusses usage scenarios. We analyze proposed solution in Section VII and offer conclusions in Section VIII.

## II. RELATED WORK

### A. Data lakes

The global digitalization trends produce large amounts of data. Current estimations show that around 2.5 quintillion bytes of data are produced and consumed globally, out of which 90% of data are unstructured [8]. Such availability of data poses the challenges for data ingestion processes, data monitoring, and data analysis. Nowadays, the concept of “Data Lake” is popular for accumulating data from heterogeneous sources. Data lakes are used for storing large-scale raw data as a single big data repository, providing ingestion, exploration, and monitoring functionality [11]. The concept of data lake has its roots also in Big Data wave and it is typically closely tethered with the Apache Hadoop ecosystem [9], [12].

Data lakes, in contrast to data warehouses, are databases containing data from different sources in structured, unstructured and semi-structured formats, along with capabilities of handling batch and real-time streams. Whereas, data warehouses are built with extract, transfer and load (ETL) approach which is further accessed by alternative database environment [12]. The integration of data warehouses with existing information systems requires costly add-ons [9], [11]. The data warehouses generally are designed for single data source with aim to facilitate specific community [12].

In comparison to data warehouses, data lakes are not domain specific, rather they can store and query different types of data from the same repository [12]. The main capabilities of data lakes are as follows:

- No extra add-ons are required for integration, access is provided by programs written by data lake developer;
- Both batch and real-time data ingestion are possible in data lakes;
- Data lakes can be scaled for continuous growth of data;
- Data lakes support storage of structured, unstructured and semi-structured data;
- Massive volumes of data can be stored with low cost;
- Easy reuse and re-purposing at low cost;
- Multiple users can access the data repository at the same time for monitoring, exploration and analysis.

Data lakes do have some limitations as well. For example, implementing a data lake requires a lot of technical effort, regardless of availability of different frameworks addressing general requirements. Integration of various data sources poses requirements for metadata management. Therefore, the lack of metadata for available raw data in the repository can make the querying and integration processes particularly challenging [10]. To solve such problems, a data lake (Constance) by Rihan et al. [10] was developed to intelligently manage metadata of heterogeneous sources for

their smooth integration within data lakes. In addition, artificial intelligence-based schema evolution approaches can be used for harmonizing data from different sources [12]. Similarly, CLAMS system [13] is focused on data management challenges of data lakes, like integrity constraints of data.

Data lakes are mostly associated with Hadoop, which allows to store data with replication at low cost and to process the data with MapReduce. Therefore, data lakes are useful mostly for batch processing [9]. Hence, in many proposed data lakes real-time notion is often not addressed. This brings up the need of implementing data lakes with hybrid functionality of both batch and stream processing.

### B. Big data city platforms

Since its first appearance in the scientific discourse in the late 1990’s, the *Smart City* concept has been gaining increased traction among researchers, policy-makers and urban planners. While a clear definition of the concept remains elusive, the popularity of the smart city builds on the promise of proliferating new technologies (IoT, sensors, smartphones, distributed/cloud/fog computing) to bring positive change to the quality of life in cities. The rapidly expanding innovation ecosystem for sensors and IoT, coupled with the data produced from disparate sources within a city, have led many urban-environment stakeholders to take an interest in big data architectures and solutions for the smart city.

One recent systematic review found that scientific advances of big data analytics in smart city research, as measured by the volume of adjacent peer-reviewed publications, has been growing rapidly between 2013-2015 and steadily from 2015 onwards [14].

Moreover, the innovation ecosystem for big data solutions in smart cities is attracting increased attention in academia [14]. Additionally, industry’s interest has been also increasing with major industry leaders developing solutions for this emerging market [15]. The high level of technical maturity is also evident when surveying the tools being used to build smart city platforms. According to reviews from [16], 23 smart city projects and found that four enabling technologies stand out in the smart city research: i) Cloud Computing, ii) IoT, iii) Big Data, and iv) Cyber-Physical Systems (CPS).

Evidently, most smart city systems and platforms use more than one of these enabling technologies. For example, architectures deployed in the SmartSantander [17] and Padova Smart City [18] projects use a multitude of IoT sensors to sense environmental and traffic conditions and make these data available for city applications through a server tier. An important limitation of these architectures is the lack of functional requirements for data pre-processing, harmonization and interoperability [16]. To overcome these limitations, projects like Scallop4SC [19] and CiDAP [20] combine Cloud Computing with Big Data technologies and use Hadoop MapReduce and Spark to process data.

### III. DATA CATALOGUE

CUTLER builds its research on four pilot cities. Moreover, diverse data is aimed to be collected: economic, social, and environmental. Therefore, it was decided to create a Data catalogue, which will contain the detailed description about data sources used for each pilot. Such description includes information like data source provider, data ownership, data acquisition mode, data access and data structure details. Furthermore, a coding system was established to characterize each individual dataset e.g. *pilotname\_tag\_data\_provider\_frequency*. For complete list of information collected about data source, refer to CUTLER deliverable D3.1 [21].

Given the scale of the project, it was decided to put this Data catalogue in the project wiki for easy access for all project partners. Moreover, Data catalogue population process was developed, to ensure that high quality data descriptions are written to the data catalogue. In short, procedure includes identification of the social, economic, and environmental data to be used in the pilots with close collaboration between city pilots, and corresponding project partners responsible for social, economic, and environmental data analysis, as well as partners responsible for technical realization of the pilots. When the data sources are identified, their documentation and registration in the Data catalogue are conducted [21].

Maintaining the Data catalogue is a continuous process. Challenges already encountered include sources becoming obsolete, appearance of new sources and variations in existing sources, which can be related to the origin (e.g. variation in number of sensors or parameters measured), to the data access methods (e.g. from parsing HTML to using REST API) or to the data structure (e.g. different representation in HTML). Therefore, constant Data catalogue monitoring is required to make sure that it contains relevant information.

### IV. DATA MANAGEMENT PLATFORM

#### A. Data characteristics

CUTLER data is diverse in content: economic, social, environmental data. Moreover, data have various representation formats such as XML (including HTML), JSON or CSV, and it comes from different sources including specialized databases, raw files (e.g. in EXCEL, ODS), web pages and APIs. Finally, data become available with disparate frequencies: from data batches that must be loaded just once, to data that become available with predefined intervals (like hourly or daily) to streaming data. A summary of the different data source origins identified so far, and their corresponding representation formats are presented in TABLE 1.

#### B. Technical requirements for data management platform

Data characteristics dictate the following technical requirements:

**Availability of diverse collection mechanisms supporting different file formats.** It should be possible to use different approaches to ingest the data into platform, e.g. by implementing scripts to pull the data from third party services or to enable one-time data load.

TABLE I. SUMMARY OF DATA ORIGINS

| Pilot                        | Environmental  | Economic                          | Social                     | Total |
|------------------------------|--|-----------------------------------|----------------------------|-------|
| Municipality of Thessaloniki | 1 Excel file from URL                                    | 2 HTML from URL<br>8 XML from API | 1 NDJSON from local folder | 12    |
| Municipality of Antalya      | 1 HTML from URL  | 8 XML from API                    |                            | 9     |
| City of Antwerp              | 3 Excel files from local folder                          | 8 XML from API                    |                            | 11    |
| Cork County                  | 14 HTML from URL<br>1 GeoJSON from API<br>1 CSV from URL | 8 XML from API                    |                            | 24    |
| Applied for all              |  |                                   | 1 JSON from API            | 1     |
| Total                        | 21   | 34                                | 2                          | 57    |

This also means, that platform should allow both collection of stream and less dynamic and historical data regardless of their format like CSV, TSV, JSON, and others.

**Reliability and fault-tolerance.** When data comes from various sources with different sampling frequencies, there is a possibility of losing the data, e.g. due to connectivity issues. Therefore, data management platform should be reliable and fault-tolerant. In this regard, choosing right ingestion frameworks is necessary, e.g. Apache Flume supports durable channels backed by the local file system to persist the unprocessed events in case of failure [22]. Other examples of big data ingestion tools include Apache Kafka [23] and Apache NiFi [24].

**Maintainability and scalability.** The number of data sources in CUTLER is continuously growing. In this regard, the new data will be produced, and older data may become obsolete. Hence scalability of storage resources is expected.

**Processing and analysis toolset.** The veracity of real-world data cannot be guaranteed. Data can be inaccurate, incomplete, inconsistent or redundant. To enable data pre-processing, the data management platform should support tools for pre-processing, as well as development of custom scripts. Examples of big data technology tools used to process and perform various kinds of analysis on large-scale data include Apache Solr [25], Apache Storm [26], and Apache Spark [27].

**Support for real-time analysis.** The traditional data lakes are more batch processing oriented due to their close association with Hadoop [9]. Often, they are primarily designed to make data available in platform for processing and analysis. Therefore, often such platforms are limited in terms of processing and analyzing real-time data streams.

CUTLER requires also that the platform supports stream data processing capabilities.

**Metadata management.** Metadata is defined as the information pertaining to gathered data itself and contains information regarding structure and semantics of data. Metadata management in data lakes is an important constraint, especially in a longer run when data is used for long period of time. If metadata is not managed, the stored information is nothing more than a dispersed pool of data.

Furthermore, over time more data will be integrated, use cases will evolve, and data will be manipulated in different ways. As a result, the original context and intended use of data may erode. Therefore, metadata management should be supported by the platform. The metadata should include also information about data preprocessing, like restructuring, transformation, conversion of data to agreed standard, reconciling and utilizing semantic methods for matching different data types. Although these techniques are not easily supported by data lakes [11], adding such functionality allows the user to maintain data quality and provides better data governance.

### C. Architecture

Guided by the requirements described in Section IV.B, we propose a data lake architecture to allow data collection, storage, and processing of diverse data. The proposed data management platform strongly relies on Hadoop Ecosystem. The current version of the platform is a cluster, containing four virtual machines, each having 2 virtual cores, 16 GB RAM, 100 GB disk space with Cloudera Distribution including Hadoop (CDH) distribution installed. The current version of the platform allows to collect, store, and process, analyze, and visualize data by using different big data technology tools. As can be seen from Fig. 1, we followed data pipeline approach to construct our architecture.

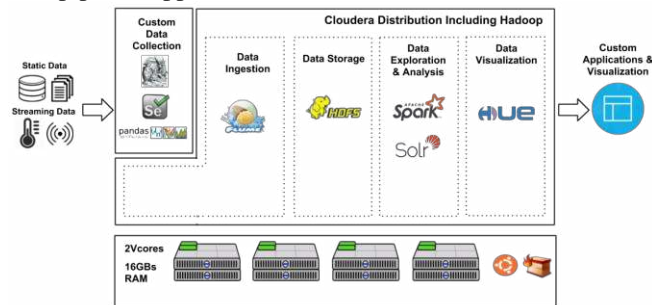


Figure 1. The proposed architecture for collecting, storing and analysing the data, redrawn from [28].

**Custom Data Collection.** The custom data collection enables data retrieval from data sources requiring custom scripts, e.g. if the data is embedded into HTML pages, APIs, or when files containing data are provided manually in CSV, TSV and PDF formats. Here, a set of three standard scripting libraries for scraping data are used, such as BeautifulSoup4 (BS4) [29], Selenium [30] and Pandas [31]. A combination of these tools is used to extract and pre-process data from sources which are available as webpages

and require human-like interaction to access the data. Whereas for data sources containing web APIs, Python-based Pandas is used for receiving the response and proceeding with its pre-processing. In addition, scripting libraries are used for restructuring and cleaning the available data.

**Data Ingestion.** Data ingestion is described as the process of importing data (batch or real-time) from external sources to the platform for the storage and analysis. In proposed platform, process of data ingestion is carried out mainly by Apache Flume. Apache Flume is a reliable solution for ingesting data to HDFS from external sources. It has three main drivers i.e. source, sink and channel. The source fetches the data from data provider and passes the information to the sink for insertion to HDFS through the channel.

**Data Storage.** To support the storage of diverse data formats, it was decided to store the data directly in HDFS with dedicated folder structure. More details on this is presented in Section V.

**Data Exploration & Analysis.** There are various tools available for processing and analyzing large-scale data. The data exploration and analysis in this study have been performed with Apache Solr and Apache Spark.

Apache Solr is a search-engine for querying the stored data. It has been developed on top of Apache Lucene [25] libraries making it capable of supporting different operations like wildcards – text matching, joins, grouping, and facet-based filtering along with others [32].

Apache Spark is used for large-scale data analysis due to its support for cluster-based computing [33], [34]. The support for general programming languages (R, Python, Java and Scala) allows the developer to take full advantage of its APIs [35]. Spark’s machine learning library, MLlib, provides common machine learning algorithms and allows analyzing large-scale data in fast manner [35]. In addition, Spark Streaming enables development of scalable stream processing applications.

**Data Visualization.** Data visualization has proven to be a powerful tool for drawing insights from data. Initial data visualization could be achieved with Hue Web interface and python-based library called Matplotlib. Hue is a user-friendly web interface developed for Hadoop, which allows the user to navigate files in HDFS, and allows to create dynamic dashboards, which include bar and pie charts, facet-based grouping, time series graphs and others. Matplotlib allows data scientists to draw a compelling story in the form of graphs and figures to simulate the extracted and analyzed data. It is recommended to use Matplotlib with IPython, which has a pylab mode that by default detects the settings for drawing interactive plots and figures [36].

## V. DATA PIPELINE IMPLEMENTATION

### A. Data acquisition

Data acquisition phase involves retrieval of data from the different identified data origins described in Section

IV.A. For data retrieval, multiple data crawlers are developed using the different python-based frameworks described in Section IV.C. Based on the data availability, manual and automated data ingestion approaches are currently supported.

**Manual data ingestion.** The manual data ingestion approach is used for the data which is provided once only or updated rarely, regardless of any defined schedule e.g. annual air quality reports. This approach is mainly followed for data supplied by the data provider itself, which is directly inserted to the system using HDFS commands or Hue web interface.

**Automated data ingestion.** Here, scripts are written to load the data to the system automatically.

*One-time data acquisition* means that the data is retrieved only once or very rarely. This is similar to manual data ingestion, but with particular script.

*Scheduled data acquisition* is used for data sources with defined schedules i.e. every hour, daily or monthly. In this case, cron scheduler executes the written crawler for extracting the information and stores it in specified spool directory – local directory in system. As soon as the file appears in spool directory, a Flume agent reads the file and injects it to corresponding HDFS directory, see Fig. 2.

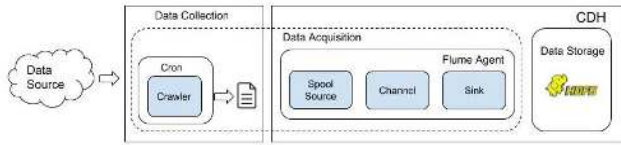


Figure 2. Scheduled data acquisition approach, redrawn from [28].

*Stream data acquisition* is used for data sources which are updated in near real-time (as soon as data is generated), e.g. Twitter and sensor readings. A push-based mechanism is used for injecting data to HDFS using a custom source in the Flume agent configuration. To ensure reliability, the file channel is used, which stores the cached data in local directory. With this configuration data can be recovered in case of failure.

### B. Data cleaning

Data cleaning [37] involves the detection and removal of errors and inconsistencies from data to improve its quality. According to [21], [38], four types of data errors within the CUTLER platform were identified: outliers, duplicates, rule-violation and pattern violation. Outliers are unusual observations that are distant from the rest of the observations. Duplicates refer to the repetition of the exact same observation. Rule-violations are values that violate any kind of integrity constraint. Finally, pattern violations refer to errors due to misspelling, formatting, or violation of semantic constraints. Some of these errors is possible to address already at the data ingestion stage whereas others may require attention during analysis.

After careful inspection of the available data sources, few cases where duplicate values could occur were detected. In order to avoid such cases, each record is programmatically checked before it is inserted, and discarded in case it is already registered in the system.

### C. Data Integration

The main purpose of data integration is to combine the data from diverse sources as a single entity for routine data processing [39]. Data integration is itself a complex task which may involve development of data models, transformation of unstructured data for further usage and avoiding inconsistency in collected data [39].

For data alignment, the first version of the data management platform relies on data models. These data models provide unified vocabulary among data sources, aligning syntactic and semantic differences. For example, data providing information about air temperature, coming from different sensors and Web services, may have different naming conventions (e.g. “temperature”, “air\_temperature”) and different measurement units (e.g. Celsius or Fahrenheit). Having unified vocabulary enables easier data integration and further data analysis.

The first version of the platform includes the data models, retrieved with individual variables’ analysis and the help of standards and accepted practices, like ISO 8601, ISO80000-3 [28]. Corresponding partners have defined the data models for environmental, social, and economic data [21], [28]. In addition, data models for temporal and spatial information were identified, see TABLE 2 for examples.

TABLE II. EXAMPLE OF DATA INTEGRATION MODELS

| Parameter Representation in Data Sources |   | Parameter Representation in Data Models |                      |
|--|---|---|----------------------|
| Name                                     | Units/representation  | Name                                    | Units/representation |
| datetime, Date, Tarih                    | YYYY-MM-DD<br>HH:mm:ss+HH,<br>YYYY-MM-DD,<br>DD.MM.YY<br>HH:mm            | DateTime                                | ISO 8601             |
| geohash, location, address               | string of letters and digits, degrees north and east, Address as a string | latitude, longitude                     | longitude, latitude  |
| WindSpeed<br>Ruzgar Hizi                 | kn (knots)<br>m/s (metre per second)                                      | wind_speed                              | knots                |

### D. Data storage

The data collected from diverse sources is stored into HDFS, allowing to store large-scale data in raw format. The files containing raw data sets are placed in defined directories for accessibility. The following directory structure is used [39]:

- <Pilot name>
- <Type of data> (environment, economic, social)
- <Data source code>
  - <Data\_code\_number>

- File1
- File2

### E. Data analysis

The current implementation of the platform supports Apache Solr [25] and Apache Spark [27] for data exploration and batch and real-time analysis. Here, we provide Twitter sentiment analysis (distinguishing between positive and negative Tweets) example built with Spark.

The combination of Logistic regression model and TF-IDF feature vectorization was used for performing the sentiment analysis. Logistic regression is used to determine the relationship of independent variable(s) with a binary response – dependent variable [40], e.g. analyzing whether the tweet in provided document shows positive sentiment or not. Term Frequency-Inverse Document Frequency (TF-IDF) is statistics-based numerical method utilized for determining the relevance of a word in a specified document [42], e.g. using the TF – frequency of word in a tweet and IDF – frequency of that word in all tweets to calculate its importance.

The training of the model was performed using a 1.6 million tweets Sentiment140 dataset [41]. The dataset contained pre-cleaned tweets. However, while exploring the data, we found few records with special characters and URLs, requiring removal. After this further cleaning, the dataset was split into two different sets: training set (80%) and test set (20%). Then, the features were extracted from training set based on TF-IDF. The extracted features were passed to the model for training. The total time required by the platform for computing TF-IDF and training of the model is as follows:

*CPU times: user 36 ms, sys: 4 ms, total: 40 ms*

*Wall time: 19.9 s (TF-IDF computations)*

*CPU times: user 28 ms, sys: 4 ms, total: 32 ms*

*Wall time: 51.1 s (Model Training)*

Model results are evaluated with the ROC curve. Training set AUC=0.886 and test set AUC=0.857.

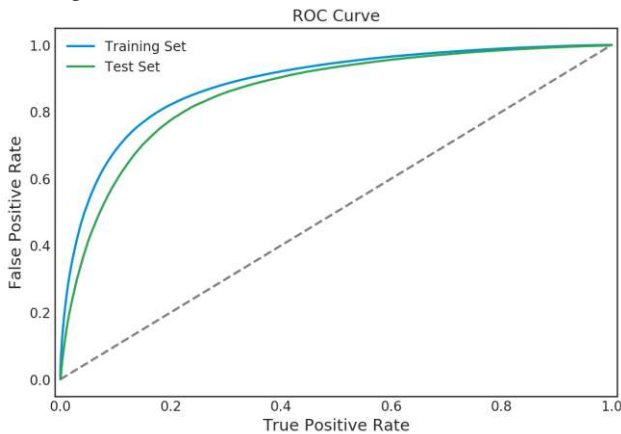


Figure 3. ROC Curve of training set and test set.

The trained model was then saved into the HDFS directory and could be used for analyzing the sentiment. For

example, with custom Flume source we collected weather-related tweets for two weeks from Cork County, Ireland. The collected tweets were cleaned with regular expressions, like removing special characters, URLs, emoticons, HTML characters, and split attached words. The cleaned tweets were supplied to the saved model for prediction. As a result, 76% of Tweets were computed to be positive about recent Cork weather and 24% were negative.

### F. Results delivery

The process of drawing decisions from data often becomes complicated especially when the data has many dimensions, is multi-sourced, and contains time-variant information. Processed data is of interest for variety of applications and stakeholders. Therefore, platform should provide a range of mechanisms for results delivery that meet a variety of potential use cases.

Visualization instruments were already discussed in Section IV. In addition, processed data and models can be used by third party applications. In this case, the platform should enable push/pull mechanisms to deliver these intermediate results to the end applications. For example, it is possible to put the data into dedicated HDFS folders or data stores for quick look up and queries for searchers (e.g. HBase) and pull these data from third party applications via APIs and drivers. Alternatively, tools like Apache Sqoop can be used to transfer the data to dedicated data stores or scripts could be written to push the data to ftp locations. Depending on the requirements, a data service as a REST endpoint could be implemented for enabling data access by third parties and hide the internal details.

## VI. USAGE SCENARIOS

The data management platform, presented in this article, enables also more demanding scenarios, like the proposed one:

Due to urbanization, the number of vehicles in use have significantly increased, especially where residential and commercial regions are geographically spread across wide areas causing traffic flow management challenges. Similarly, facilitating parking spaces for a mobile population becomes problematic. According to current estimations, the cause of about 30 percent of traffic congestion is due to vehicle owners searching for vacant parking slots [43]. Besides, it is determined that vehicle owners spend approximately 3.5 to 14 minutes in searching for available parking spaces [43]. The typical factors contributing to this spent time are distance to the parking garages, purpose of travel, knowledge about respective geographical area, and weather conditions [44].

The struggle for finding the available parking slots can be reduced by performing predictive analysis on factors that influence traffic flows and parking spaces. This can help in improving traffic management policies and notifying the vehicle owners about traffic and parking situation beforehand.

We propose that combination of big data and advanced machine learning methods can help in producing accurate predictions about parking availability from heterogeneous data sources. In this regard, the implemented data lake can help in collecting and analyzing the produced data from traffic counters, parking sensors, geographical information and weather conditions to address the arising parking problems. The data from these sources is currently being collected to the platform. After fusing the data, we will experiment different machine learning techniques for performing a predictive analysis.

## VII. DISCUSSION

### A. Technological alternatives

At this stage, the proposed solution is a basic data lake, accumulating data from diverse data sources and facilitating further analysis and results delivery phases. Alternative data management and processing architectures could be experimented whether they provide better support for the project, e.g. lambda-architecture. Data ingestion process is currently performed using Apache Flume. As we use Flume Spooling Directory Source to ingest the data to the lake, extra disk space for storing the queried files is needed. One approach to overcome the issue is to use custom Flume sources. In addition, alternative frameworks could be tested for data injection, e.g. Apache Kafka or Apache NiFi. The used visualization tools are general in nature. Integrating more advanced solutions like Kibana and Grafana can provide richer user experience. This is left for future work.

### B. Data sources and meta-data management

Data harmonization and interoperability appear to be the most challenging phases at this stage. The current implementation relies on developed data models, which are recommended for partners to follow. However, better approach is required, e.g. enabling verification upon the data models. Semantic Web technologies provide solutions for resolving interoperability issues, e.g. with the use of well-known ontologies. Experimenting with such solutions is left for future work.

Metadata management is very important aspect for any data lake. At this stage we apply two mechanisms: 1) embedding metadata information in file paths and names (see Section V.D); 2) complementing data with metadata description file. Such an approach is flexible enough, however, it places the metadata management responsibilities on data lake developers. Given the fact that some data sources become obsolete and new data sources become available, better procedures for metadata management are required. In the future, we will experiment with alternative instruments to manage metadata, e.g. Apache Atlas or Kite SDK.

### C. Limitations

In this study, we have implemented only the main components for data lake to accumulate data from disparate

sources. However, the current state of the platform doesn't address fully the issues related to data security and data quality. Controlling the quality of data requires, in addition to data pre-processing, advanced metadata management techniques e.g. verification against data models. Current version of the platform provides limited support for that.

The second main limitation in our platform is maintaining its security and ensuring that integrity of data remains intact. During the implementation we faced different challenges, like keeping the platform invulnerable for any kinds of attacks. The access of the platform is currently limited to only whitelisted IPs. In our opinion, this is still not sufficient and requires further research for adopting advance security protocols. For the final deployment of the CUTLER platform, an emphasis on security drawing from the expertise of cyber-security specialists will provide further assurances over the platform security.

## VIII. CONCLUSIONS AND FUTURE WORK

This work presents the very first implemented solution for CUTLER to collect, store, and process heterogeneous data from four pilot cities. The article has described the process of collecting data from diverse stakeholders; platform design, supported by requirements coming from diverse data; platform implementation; analyzed the challenges and limitations; and proposed solutions to explore in the future. So far, in CUTLER project, data from 57 origins were crawled, processed and stored in the platform, supporting pilot needs.

Future work will continue to advance the platform and implement the scenario presented in Section VI. The current implementation of the platform will be evaluated as part of the future work. Moreover, as it is described in [6] and [7], the CUTLER architecture will also explore the use of Elastic stack within the architecture.

### ACKNOWLEDGMENT

This research has been financially supported by EU Horizon 2020 project CUTLER: Coastal Urban development through the Lenses of Resiliency, under contract no. 770469 (<http://www.cutler-h2020.eu/>), ERDF project A71720, 'Big Data for 5G', governed by the HILLA program ([www.hilla.center](http://www.hilla.center)), and by Academy of Finland 6Genesis Flagship (grant 318927).

### REFERENCES

- [1] United Nations, The Sustainable Development Goals Report, <https://unstats.un.org/sdgs/files/report/2018/TheSustainableDevelopmentGoalsReport2018-EN.pdf>, 2018.
- [2] Frost and Sullivan, "Smart City Adoption Timeline," *Global Information, Inc.*, 24-Jan-2018. [Online]. Available: <https://www.giiresearch.com/report/fs604427-smart-city-adoption-timeline.html>
- [3] R. Kitchin, "The real-time city? Big data and smart urbanism," *GeoJournal*, vol. 79, no. 1, pp. 1–14, 2014.

- [4] S. Pirttikangas *et al.*, “Experiences with smart city traffic pilot,” in *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*, 2016, pp. 1346–1352.
- [5] CUTLER website, <https://www.cutler-h2020.eu/>
- [6] M.S. Yümlü *et al.*, CUTLER deliverable D8.1 “Integration Protocol and Technical Verification”, 2018
- [7] A. Barnett and A. Byrne, CUTLER deliverable D2.2 “Threat analysis for policysupporting hybrid cloud infrastructure”, 2018
- [8] U. Sivarajah, M. M. Kamal, Z. Irani, and V. Weerakkody, “Critical analysis of Big Data challenges and analytical methods,” *J. Bus. Res.*, vol. 70, pp. 263–286, 2017.
- [9] C. Madera and A. Laurent, “The next information architecture evolution: The data lake wave,” in *8th International Conference on Management of Digital EcoSystems, MEDES 2016*, 2016, pp. 174–180.
- [10] R. Hai, S. Geisler, and C. Quix, “Constance: An Intelligent Data Lake System,” in *Proceedings of the 2016 International Conference on Management of Data - SIGMOD '16*, 2016, pp. 2097–2100.
- [11] H. Fang, “Managing data lakes in big data era: What’s a data lake and why has it become popular in data management ecosystem,” in *2015 IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, IEEE-CYBER 2015*, 2015, pp. 820–824.
- [12] D. E. O’Leary, “Embedding AI and Crowdsourcing in the Big Data Lake,” in *IEEE Intelligent Systems*, vol. 29, no. 5, pp. 70–73, Sept.–Oct.2014. doi: 10.1109/MIS.2014.82
- [13] M. Farid, A. Roatis, I. F. Ilyas, H.-F. Hoffmann, and X. Chu, “CLAMS: Bringing Quality to Data Lakes,” *Proceedings of the 2016 International Conference on Management of Data - SIGMOD 16*, 2016.
- [14] S. N. Brohi, M. Bamiah, and M. N. Brohi, “Big Data in Smart Cities: a Systematic Mapping Review,” *J. Eng. Sci. Technol.*, vol. 13, no. 7, pp. 2246–2270, 2018.
- [15] A. Spadafora, “Dell Boomi: Bringing smart cities to life,” *TechRadar*, 16-Nov-2018. [Online]. Available: <https://www.techradar.com/news/dell-boomi-bringing-smart-cities-to-life>. [Accessed: 22-Jan-2019].
- [16] E. F. Z. Santana, A. P. Chaves, M. A. Gerosa, F. Kon, and D. S. Milojevic, “Software Platforms for Smart Cities,” in *ACM Computing Surveys*, 2017, vol. 50, no. 6, pp. 1–37.
- [17] L. Sanchez *et al.*, “SmartSantander: IoT experimentation over a smart city testbed,” *Comput. Networks*, vol. 61, pp. 217–238, 2014.
- [18] A. Cenedese, A. Zanella, L. Vangelista, and M. Zorzi, “Padova smart City: An urban Internet of Things experimentation,” *Proceeding IEEE Int. Symp. a World*
- [19] S. Yamamoto, S. Matsumoto, S. Saiki, and M. Nakamura, “Using materialized view as a service of scallop4sc for smart city application services,” in *Advances in Intelligent Systems and Computing*, 2014, vol. 271, pp. 51–60.
- [20] B. Cheng, S. Longo, F. Cirillo, M. Bauer, and E. Kovacs, “Building a Big Data Platform for Smart Cities: Experience and Lessons from Santander,” in *Proceedings - 2015 IEEE International Congress on Big Data, BigData Congress 2015*, 2015, pp. 592–599.
- [21] F. Tsalakanidou *et al.* CUTLER D3.1 “Requirements for data crawling, integration and anonymization”, 2018.
- [22] “Welcome to Apache Flume,” *Flume 1.8.0 User Guide - Apache Flume*. [Online]. Available: <https://flume.apache.org/>. [Accessed: 22-Jan-2019]
- [23] *Apache Kafka*. [Online]. Available: <https://kafka.apache.org/>. [Accessed: 22-Jan-2019].
- [24] *Apache NiFi*. [Online]. Available: <https://nifi.apache.org/>. [Accessed: 22-Jan-2019].
- [25] *Apache Solr*. [Online]. Available: <http://lucene.apache.org/solr/>. [Accessed: 22-Jan-2019].
- [26] *Apache Storm*. [Online]. Available: <http://storm.apache.org/>. [Accessed: 22-Jan-2019].
- [27] *Apache Spark™*. [Online]. Available: <https://spark.apache.org/>. [Accessed: 22-Jan-2019].
- [28] E. Gilman *et al.* CUTLER D3.2 “First version of the framework for the collection, cleaning, integration & anonymization of big data”, 2018.
- [29] “beautifulsoup4,” PyPI. [Online]. Available: <https://pypi.org/project/beautifulsoup4/>. [Accessed: 22-Jan-2019].
- [30] *Selenium Blog Posts*. [Online]. Available: <https://www.seleniumhq.org/>. [Accessed: 22-Jan-2019].
- [31] “Python Data Analysis Library,” *pandas: powerful Python data analysis toolkit - pandas 0.23.4 documentation*. [Online]. Available: <https://pandas.pydata.org/>. [Accessed: 22-Jan-2019].
- [32] E. Lacic, D. Kowald, D. Parra, M. Kahr, and C. Trattne, “Towards a Scalable Social Recommender Engine for Online Marketplaces : The Case of Apache Solr c Categories and Subject Descriptors,” in *WWW 2014 Companion: Proceedings of the 23rd International Conference on World Wide Web*, 2014.
- [33] S. Gopalani and R. Arora, “Comparing Apache Spark and Map Reduce with Performance Analysis using K-Means,” *Int. J. Comput. Appl.*, 2015.
- [34] X. Meng *et al.*, “MLlib : Machine Learning in Apache Spark,” *J. Mach. Learn. Res.* 17, vol. 17, pp. 1–7, 2016.
- [35] S. Salloum, R. Dautov, X. Chen, P. X. Peng, and J. Z. Huang, “Big data analytics on Apache Spark,” *Int. J. Data Sci. Anal.*, vol. 1, no. 3–4, pp. 145–164, 2016.
- [36] J. D. Hunter, “Matplotlib: A 2D graphics environment,” *Comput. Sci. Eng.*, 2007.
- [37] E. Rahm and H.H. Do, “Data cleaning: Problems and current approaches,” *IEEE Data Eng. Bull.*, 23.4 (2000): 3-13.
- [38] Z. Abedjan *et al.*, “Detecting Data Errors: Where are we and what needs to be done?,” in *Proceedings of the VLDB Endowment*, 2016, vol. 9, no. 12, pp. 993–1004.
- [39] A. Kadadi, R. Agrawal, C. Nyamful, and R. Atiq, “Challenges of data integration and interoperability in big data,” in *Proceedings - 2014 IEEE International Conference on Big Data, IEEE Big Data 2014*, 2014.
- [40] A. C. Leon, “3.12 – Descriptive and Inferential Statistics,” in *Comprehensive Clinical Psychology*, Pergamon, 1998, pp. 243–285.
- [41] A. Go, R. Bhayani, and L. Huang, “Twitter sentiment classification using distant supervision,” CS224N Project Report, Stanford, pp. 1–12, 2009.
- [42] P. Barnaghi, P. Ghaffari and J. G. Breslin, “Opinion Mining and Sentiment Polarity on Twitter and Correlation between Events and Sentiment,” *2016 IEEE Second International Conference on Big Data Computing Service and Applications (BigDataService)*, Oxford, 2016, pp. 52-57. doi: 10.1109/BigDataService.2016.36
- [43] A. Ionita, A. Pomp, M. Cochez, T. Meisen, and S. Decker, “Where to Park?: Predicting Free Parking Spots in Unmonitored City Areas,” in *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, 2018, p. 22:1--22:12.
- [44] C. Badii, P. Nesi, and I. Paoli, “Predicting Available Parking Slots on Critical and Regular Services by Exploiting a Range of Open Data,” *IEEE Access*, vol. 6, pp. 44059–44071, 2018.