Data Obsolescence Detection Within Connected Environments.

Jean Raphael Richa, Hamza Noueihed, Richard Chbeir

Data Obsolescence Detection Within Connected Environments

Jean Raphael Richa
University Pau & Pays Adour
E2S UPPA, LIUPPA
64600, Anglet, France
jean-raphael.richa@etud.univ-pau.fr

Hamza Noueihed
University Pau & Pays Adour
E2S UPPA, LIUPPA
64600, Anglet, France
hamza.noueihed@etud.univ-pau.fr

Richard Chbeir
University Pau & Pays Adour
E2S UPPA, LIUPPA
64600, Anglet, France
richard.chbeir@univ-pau.fr

***Abstract*—Connected environments collect data from millions of devices every second presenting valuable information to the community. However, data integrity faces critical research gaps especially when it comes to data obsolescence and its detection. Thinking of data obsolescence as an independent terminology, we define it as the state wherein data is no longer significant or effective with respect to the device, parent zone and/or the environment. Accordingly, this paper provides connected environments with a dimensional architecture that : (1) assess predefined quality metrics of each sensed data, (2) discovers interrelation between deployed devices using clustering techniques, and (3) identifies data obsolescence through three main layers; namely, the device, hosting zone, and/or environment level.**

***Index Terms*—+ Connected environments, Data obsolescence detection, Data quality metrics, Dimensional architecture.**

I. Introduction

"Smart and more connected" is the current trend within the emerging metaverse. Smart cities deployed to collect millions and millions of data are helping in better understanding human activities, improving quality of life, and easing paths for sustainability. For instance, cities may boost their energy efficiency by 30% if their data is well utilized [1]. It is also the case in other sectors such as healthcare, finance, transportation, and retail to mention a few. However, this is possible when having a good information and communication technology (ICT) infrastructure [2]. Connected environments represent one important sector reinforcing data collection, analysis, and processing [3]. Quantifying real data from a connected environment can aid in advanced advisory and warning information across various maneuvers (such as traffic interactions and driving decisions which showed no evidence of negative impact [4]). This pushes, for instance, regional agglomeration's collective intelligence forward, supporting key urban flows; namely, forecast and management [5]. Accordingly, this all-connected paradigm is producing more and more valuable data for country, society, community, or even individual.

While connected environments are providing us with valuable data, it is notable that not all data is of benefit for the topic of interest. While many efforts have focused on data speed and security, less emphasis has been dedicated towards data quality and more particularly data significance. "Data Obsolescence", a key pillar of data integrity, is the representative behind this research gap. Knowing to which extent a data is useful and relevant to the question in hand will increase productivity and efficiency while reducing the technological lags that various sectors face. It is to be noted that the obsolescence term was initially introduced as electronic part obsolescence when military specifications were required by some stringent governmental products; however, these specifications were hard to attain causing the unfinished product to become insignificant [6]. Similarly, a data is thought of as a product with a level of importance impacting the efficiency of the whole data set. On the other hand, a data's significance towards the data set it belongs to experiences minimal research especially in the information technology sector. Connected environments are one of these sectors, especially that they are purposed to collect, store and analyze huge amount of data. In most existing approaches, obsolescence is represented globally as another terminology such as contradiction or relevance. Although these terminologies are key factors in data quality measurements, they should act as indicators helping in data obsolescence and not represent the latter itself. For this purpose, in our previous work [7], we aimed in officially defining data obsolescence as an independent terminology providing it with quality metrics essential for its computation and detection. Accordingly, in this paper, based on the metrics we provided in [7], we propose an efficient dimensional architecture for detecting obsolete data within connected environment infrastructures. Detecting data obsolescence is only the first step inside a bigger approach. Data processing is our future goal as it addresses the sources that constantly generate such obsolescence. Being able to detect such issues will help in providing processing solutions and a possibility to return the data's usefulness or even finding a new fitting scenario for the data. Take note, solving the obsolescence source does not necessarily bring improvements at the level of data storage solely; other benefits might originate in terms of location, deployment, and data querying of the device. For instance, why would a device stay in its location or zone if one knows that it is not producing useful data (such as a monitoring camera that is partly or fully blocked by a wall or any other object)? Detecting obsolescence can uncover such mishaps improving device deployment for a more comprehensive collection. The latter is one of many scenarios on how understanding data is a direct way of understanding productivity methodologies.

The remainder of the paper is organized as follows: Section II illustrates a motivating scenario. Section III presents the state of the art. Section IV describes our proposed data obsolescence detection architecture. Finally, Section V summarizes the paper and provides future research directions.

II. Motivating Scenario

Here we consider a scenario of a smart connected conference venue to clearly highlight and describe connected environments and data obsolescence motivation and concerns. It is important to note that we are presenting the setup and needs of connected environments in a simplified manner, and it does not necessarily fully incorporate all data obsolescence issues. Figure 1 depicts this connected environment which includes five zones (i.e., geographical areas) whereby four zones designate conferences rooms sharing a common hallway zone. The zones host different categories of devices: (1) static devices (e.g, temperature (T1, T2, T3, T4, and T5), fixed camera (C4)), and (2) mobile devices (e.g., rotating camera (C1, and C2), mobile phone (P1, and P2)). Each device has a well defined location and senses different environmental parameters (e.g., temperature, presence, motion) within particular coverage areas (ca) represented by hashed shapes (i.e., hashed rectangle, circle, or cone). A device's coverage area is not necessarily limited by the respective hosting zone. In other words, a device can measure data from adjacent zones depending on the data and border nature (e.g., since zone2 and zone3 are separated by a glass wall, C2 located in zone2 might capture data from zone3).

One can interact with the environment through queries, by requesting data from any device. In Figure 1 queries are illustrated by dotted shapes. There are two types of queries: (1) Projection: requesting all data collected by a device, (2) Selection: querying certain data gathered by a device through specifying particular conditions. Each query is assigned to one or multiple devices, and each device keeps record of the requests it receives. For instance, q1, q2 and q3 are assigned to device C1 while q4 and q5 are assigned to C4, both devices keeping track of when data was queried.

This scenario highlights the dynamicity of a connected environment as it exposes its evolution over time. This dynamic nature is due to the zones capability of merging or splitting (e.g., zone4 and zone5 are combined into zone4bis resulting in a larger conference room), and mobile devices' capability of moving. For instance, mobile devices can change location within the same zone (e.g., device P1 moved within zone1), change zones (e.g., P2 moved from zone1 to zone2), and/or rotate to sense data from different angles (e.g., rotating cameras C1, C2, and C3).

One or more sources might be behind the generation of data obsolescence. Here are some illustrative cases:

- Case 1: A 360° rotating camera (C1) covering the entire zone every minute while only data every hour are required (i.e., temporal source).
- Case 2: A 360° rotating camera (C1) periodically collecting data from the whole zone when only data belonging within 120° portion are needed (i.e., spatial source).
- Case 3: A mobile device (P1) capturing several types of data (e.g., temperature, presence) while only temperature data are required (i.e., contextual source).

Now that the general view was described, the next step revolves around listing the challenges we will be addressing:

- Challenge 1: How to assess data as obsolete ?
- Challenge 2: How to evaluate obsolete data with respect to the connected environments and their dynamicity ?
- Challenge 3: What dimensions to consider for obsolescence source interpretation ?

III. State of the Art

In 2015, McKinsey Group estimated by 2025 a potential loss of $3.9-11.1 trillion per year for the IoT sector; i.e., worth 11% of the world's economy [17]. Lack of data processing being one of the latter's main causes, increasing data cohesiveness is deemed necessary. Heterogeneous solutions are required with the evolving smart cities concept supporting higher urbanism qualities [18]. In [19], a generic IoT architecture is built of: (1) a perception layer: devices and sensors, (2) a network layer: data transmission, and (3) an application layer: user view point. Here, the authors explain how the perception layer continuously faces major challenges including storage supervision and device durability. Similar cases for the network layer, interference and communication range are in crucial

Fig. 1: Motivating scenario

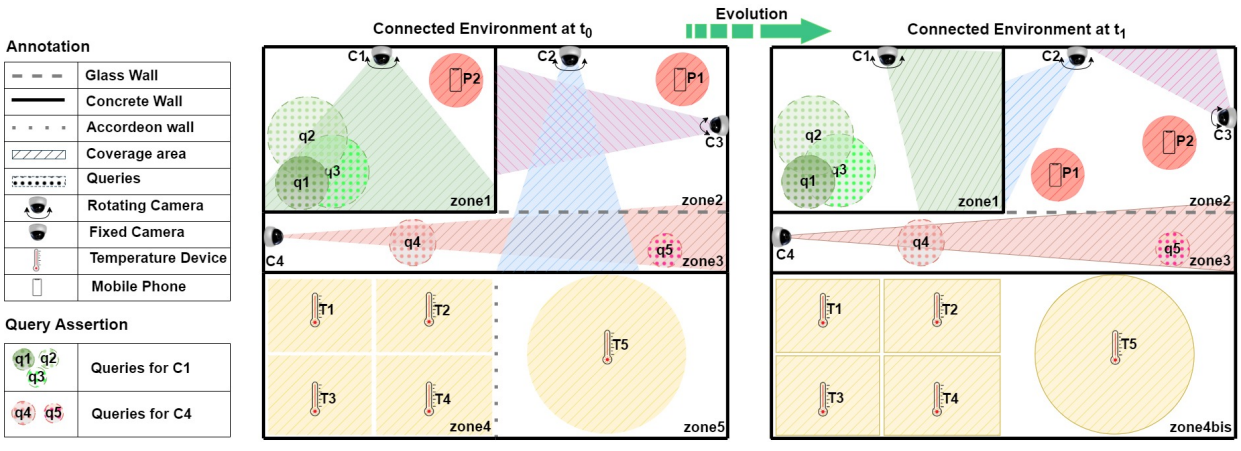


TABLE I: Data Obsolescence Detection Comparison

Approaches	Data Quality Metrics										CE and Dynamicity	Obsolescence Grounds		
	ac	pl	rl	tl	a	av	cr	ct	cn	fc		Temporal	Spatial	Contextual
[8]				✓	✓							✓		
[9]		✓		✓								✓		
[10]				✓			✓					✓	✓	
[11]			✓	✓								✓	✓	
[12]				✓	✓		✓					✓	✓	
[13]				✓				✓				✓		
[14]	✓	✓												
[15]				✓		✓						✓		
[16]			✓	✓								✓		
Our contribution	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

need for solutions. These obstacles still need processing due to the aforementioned issue; lack of emphasis on the data itself and how significant it is. This reflects the impact that data obsolescence will have once it is detected and dealt with. Connected environments, using this generic IoT architecture, are in need of a detection methodology required by this domain's research breach.

The aim of our study is to provide a solution to detect data obsolescence within a connected environment. Although little (to no) research is done in this field, yet there exists some studies related to obsolescence in general. Life cycle curve forecasting is the main obsolescence detect method used in [20]. The authors depend on data mining; however, it is worth mentioning that this obsolescence is specific to physical electronic parts (e.g., integrated circuits). In other words, the authors deal with real vendor parts while connected environments generally focus on cloud data provided by the embedded devices (e.g., real and virtual sensors). A similar topic, which is the product life cycle, is the focus in [21], especially when it comes to data centers and infrastructure. The authors suggest: (1) reactive approaches: solving obsolescence after it occurs, and (2) proactive approaches: preventive measures and obsolescence predictability, or the formers combined. While they appeal representative of data obsolescence, they are described theoretically without an actual detection and computation strategy; the gap this paper aims to fill in. In [22], Rojo et al. extend on electronic part obsolescence agreeing how a component's life cycle is usually shorter than that of the system; i.e., insignificance at the technology level. Logic wise, the aim is to prove that a similar process occurs at the data level of evolving technologies. At one point, a piece of data is very important to a system or user while at another point, it might become no longer up to date and not representative of the real world. Similar to how electronic parts have detection methods, this paper's mission revolves around filling in this gap for data obsolescence via providing a layered detection infrastructure.

Storage data obsolescence, specifically temporal obsolescence, is the main topics [10] discusses. Data repositories aim to save the world's dynamic states; however, if they change faster than our capability to save, they become obsolete. In other words, it's temporal obsolescence representing data going out of date whereby each data has a credibility parameter which changes over time. For that purpose, the authors agree that their analytical methodology is a difficult approach as the decay in credibility requires experts in the domain at hand. Similarly in [8], old data is considered obsolete with respect to time solely; i.e., the authors mainly focus on the timeliness factor for classifying obsolescence reiterating the fact how it is regularly confused as obsolescence itself. In this paper, we aim to reveal that data obsolescence does not necessarily have to be solely a temporal phenomena as it can also be spatial and/or contextual (i.e., providing an architecture capable of detecting and identifying the type of obsolescence). In [9], obsolescence has been also addressed in databases, namely relational databases. Each table undergoing obsolescence cycles goes through at least three checks each with a specified time limit. A data record is then considered obsolete when not accessed after a certain number of cycles 'M'. In this context, the provided approach focuses more on data popularity whereby the number of access is the primary pillar. This refers back to the idea of confusing data obsolescence with another existing quality indicator. Accordingly, popularity is a crucial parameter that works alongside other indicators towards a comprehensive obsolescence detection.

Few authors (e.g., authors of [23]–[25]) categorize obsolescence as either: (1) technological obsolescence: new technologies replaces older ones, (2) functional obsolescence: reduction of a system's performance, (3) logistical obsolescence: shortages in manufacturing and materials, (4) ecological obsolescence: high environmental negative impacts, or (5) economic obsolescence: discarding a product due to high maintenance. In [26], the authors provide mathematical detection/prediction methods for obsolescence based on the category of interest. Gaussian distribution is used for data obsolescence at the end-of-product level. For general technological obsolescence, the authors suggest logistic regression, binary assessment such as in or out of production, in order to predict the obsolescence. In [26], other insights are put forward tending to lean towards electronic part obsolescence and product life cycles. Accordingly, this reinforces the lack of detection and prediction methods at the level of data. In other words, technological obsolescence is not necessary that of a physical electronic part, rather a much wider scope. For that purpose, our study here extends our previous research [7] where we provided several definitions (about how the connected environment is built with all its zones and sensors, sensor measurements called individual data, the sensor's data set called data item,

the various parameters provided by the sensor's documentation and environment, etc.). Specific indicators are computed at the level of individuals data (e.g., timeliness) and data item (e.g., accuracy) to provide a comprehensives detection pipeline. Table 1 re-emphasises, in a comparative manner, how existing obsolescence detection approaches fail in covering the following three criteria that address the challenges previously listed in Section II:

- Data quality metrics: Stating the data quality metrics taken into account by the approach (cf. Challenge 1).
- Connected environment and dynamicity: Denoting whether the approach is compatible with the connected environments and its changing nature (cf. Challenge 2).
- Obsolescence grounds: Specifying the obsolescence types supported by the approach; namely, temporal, spatial, and/or contextual (cf. Challenge 3).

While few related works (cf. Table I) manage to discuss some quality indicators (such as popularity and relevance), none entirely fulfils all our pre-proposed criteria (i.e., accessibility ac, popularity pl, relevancy rl, timeliness tl, accuracy a, availability av, credibility cr, contradiction ct, correctness cn, format consistency fc). Worth mentioning, these approaches delve into the indicators from a general-scope perspective without necessarily putting forward a calculation methodology that can be applied. As noticed in Table I, most papers cover data quality from the time perspective with less focus on other obsolescence possibilities such as spatial and contextual aspects. Opposite these approaches, our contribution is comprehensive using various data quality metrics especially that it considers the dynamic nature of the connected environments. Focusing on connected environments and with the pre-identified algorithms, we discuss in the next section how quality metrics, being the key pillars for analysis, contribute to our layered architecture.

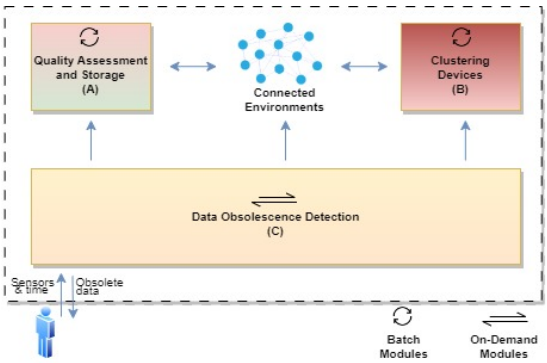
IV. Proposal

Our previous work [7] selects data quality metrics necessary for data obsolescence, categorizes them into individual data and data item quality metrics, and adapts their definitions to deal with connected environments. In this context, we provide data obsolescence with the following definition: "Data obsolescence in connected environments describes the state wherein data is no longer significant or effective with respect to the device, parent zone and/or the environment". In this section, based on the provided definition, we propose an approach for detecting data obsolescence (cf. Fig. 2). This approach consists of three main modules:

A) Quality Assessment and Storage: Pre-detection process responsible for collecting data and enriching it with efficient metadata.
B) Device Clustering: Clustering process allowing to detect dynamically the device neighborhood.
C) Data Obsolescence Detection: Detection process employed for identifying data obsolescence with respect to the device, parent zone and environment.

The modules (A) and (B) process differently than the module (C). More specifically, the first two modules are always running and listening to the environment (i.e., Batch modules), while module (C) only runs on requests (i.e., On-Demand module). Figure 2 represents the workflow followed by the modules. As we can see, one invokes the module (C) whenever he needs to detect the obsolescence of specific devices during a given time. During this detection process, the module (C) interacts with the batch modules (A) and (B) in order to fetch real-time information about data quality and device neighborhood respectively.

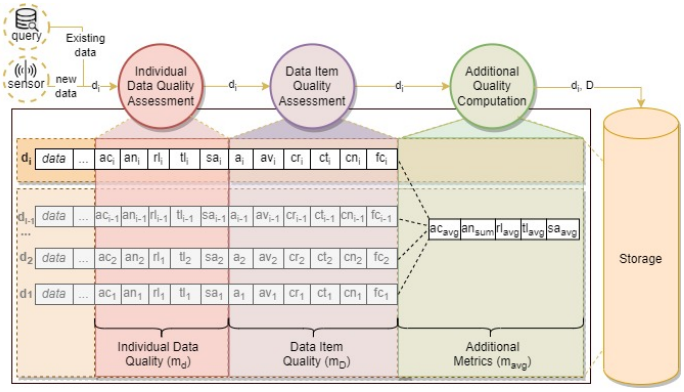
Fig. 2: Overall obsolescence detection architecture



In what follows, we highlight and detail separately each module.

A. Quality Assessment and Storage

Fig. 3: Data quality assessment and storage



This module calculates and saves the quality metrics necessary for data obsolescence detection pipeline (cf. Fig. 3). It maintains and provides the most up-to-date data quality metrics. The metrics computation is triggered for every new sensed data and/or every data interacting with any query. In other words, whenever d_i gets collected via a sensor or becomes the point of interest of a query, its quality metrics (i.e. m_d, m_D) gets created or modified respectively. Subsequently, two actors can execute this process: sensors that generate the data and the queries that request the existing data. For example, each time a query retrieves d_i, its ac increases with respect to the incremented access number. In essence, whether a data is getting gathered or engaged by a query, it goes through the three following phases pursing its storage along with its corresponding quality metrics:

1) Individual Data Quality Assessment: computes the quality metrics of the new individual data d_i represented by $m_d :\prec ac, pl, rl, tl, sa \succ$. Precisely, it is composed of accessibility ac, popularity pl, relevancy rl, timeliness tl, and spatial accuracy sa. The storage of d_i and m_d is illustrated in Fig. 3.

When it comes to popularity (pl), it is based on the number of requests an individual data d_i receives out of the overall requests to the respective data item set D_i. In other words, every time d_i is accessed, its popularity increases; however, all other individual's popularity would also require update to accommodate the new total number of requests towards D_i. The latter is exhaustive; continuously updating all individuals' popularity percentage per access is inefficient. Consequently, we save pl as the number of access; instead of saving the popularity percentage, saving the number of access is sufficient whereby pl is computed when required. This approach leads to updating only the individual requested and incrementing the total access of the data set; a less bulky process providing the same outcome.

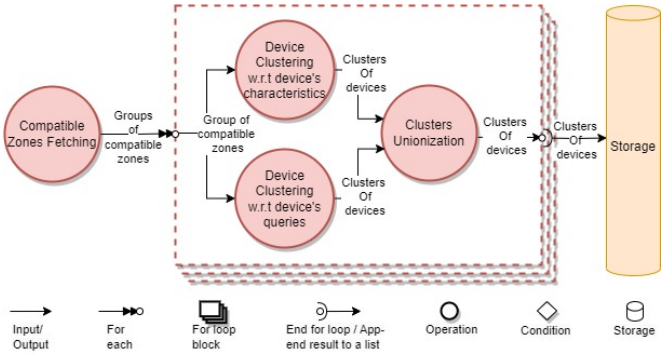
2) Data Item Quality Assessment: assesses the data item's new quality metrics computed after d_i gets collected. It is designated by $m_D :\prec a, av, cr, ct, cn, fc \succ$, and consists of accuracy a, availability av, credibility cr, contradiction ct, correctness cn, and format consistency fc. It is saved in the tuple of d_i (cf. Fig. 3), allowing to track how the quality of the data item is getting altered over time as more and more data is generated by the sensor. Subsequently, it enables the preservation of m_D history. It is worth mentioning that the current data item quality metrics are saved in the tuple of the last collected data d_i.

3) Additional Quality Computation: aggregates the m_d of all individual data into averages. It is represented by $m_{avg} :\prec ac_{avg}, an_{avg}, rl_{avg}, tl_{avg}, sa_{avg} \succ$. Specifically, five main additional metrics relative to all the individuals' m_d are computed; namely,:

- ac_{avg}: average of all individuals' ac
- an_{sum}: total accesses/requests to D
- rl_{avg}: average of all individuals' rl
- tl_{avg}: average of all individuals' tl
- sa_{avg}: average of all individuals' sa

B. Device Clustering

Fig. 4: Device clustering



This module is responsible for dynamically computing and storing device neighborhood; an essential requirement for data obsolescence detection pipeline (cf. Fig. 4). This is achieved by identifying clusters of neighbors; group of devices that are thought to impact on another. There are two types of neighbors: (1) devices sensing same physical properties, sharing similar spatial characteristics (i.e., located nearby), and having compatible coverage areas (i.e., intersects, contains, equals), and (2) devices receiving queries asking for same physical characteristics, and are compatible (i.e., intersects, contains, or equals).

To pinpoint the aforementioned neighbors, the clustering module goes through multiple steps (cf. Fig. 4):

1) Compatible Zones Fetching: The initial step consists of retrieving and grouping all compatible zones; zones that are directly adjacent and have physical barriers that do not block sensing. Namely, the latter are selected according to the adaptability of the physical barriers, type of detection and devices' coverage areas. Accordingly, this step helps determine which zones should be considered when locating devices' neighbors. Identifying neighbors from several zones is crucial since devices are capable of measuring data outside their hosting zone (i.e., device c2 cf. Fig. 1).

2) Device Clustering: The aim of this step is to identify all device's neighbors within compatible zones. To achieve this, two clustering algorithms are applied in parallel for every group of compatible zones.

a) Characteristics Clustering of Devices: The first clustering is related to the physical characteristics of the devices themselves. Devices get grouped based on what their sensors are capable of measuring (e.g., temperature, humidity, wind) alongside their location, and accordingly coverage area.

b) Query Clustering of Devices: The second clustering is according to the queries of the devices. Devices get grouped according to the intersection of queries they receive for the same physical characteristics.

3) Cluster Unionization: Each clustering algorithm, previously mentioned, results in several groups of devices. In this step, these two results are merged into a single one.

As a result, this model identifies all device's neighbors. However, as aforementioned in Section II, the connected environment's dynamicity leads to unstable variations in a device's neighborhood. For this purpose, the clusters are updated whenever the environment evolves (e.g., new devices appear, devices change zones).

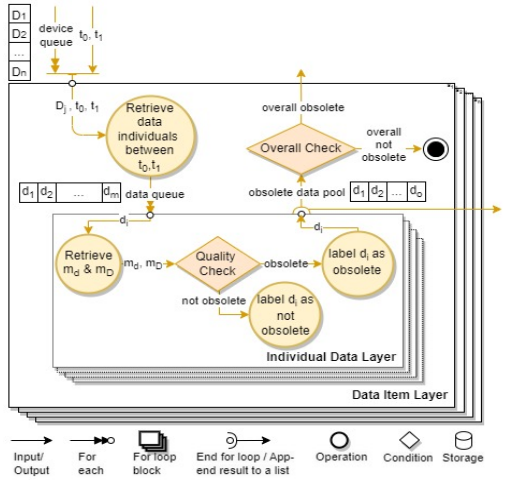
C. Data Obsolescence Detection

This module is implemented for detecting obsolescence of data collected by particular devices. The detection is applied on a selection of devices based on the system's default choice, expert's recommendation, or user's customization (e.g., obsolescence detection every day at 3:00 for the data generated/accessed within the last 24 hours by certain devices). Consequently, two main inputs are taken into consideration: (1) a queue of devices to check obsolescence for, and (2) a specific time span $[t_0, t_1]$ of interest that the data was generated within. As an end result, the framework identifies the obsolescence result and type for each data individual.

Data generated by a single device is concerned relative to this device, the related zone, and the environment itself. In other words, to be able to check data obsolescence, it is mandatory to check it with respect to all entities that might be in need of it. For that purpose, in order to have a cohesive detection pipeline, our architecture is based on three main levels: (1) Device level, (2) Parent zone level, and (3) Environment level, each to be later on delved into.

1) Device Level: As depicted in Figure 5, this level examines sequentially a queue of devices through retrieving, analyzing, and verifying each device's data individuals and qualities. The latter produces a group of the obsolete individuals per device. Each device then undergoes a further evaluation, once all its data have been assessed, to determine its overall quality. In essence, for every device concerned, this level evaluates each piece of data from two different perspectives: (a) Individual data layers: per data for a more comprehensive detection, and (b) Data item layers: against all the data collected by the same device.

Fig. 5: Obsolescence detection: Device level



a) Individual data layer: This layer is responsible for checking the obsolescence of one individual data (i.e., d_i). The first step is retrieving the quality metrics (i.e., m_d, m_D, and m_{avg}) of d_i (cf. Fig. 3). Following this, the algorithm "Individual Data Quality Check" (algorithm 1) is applied for detecting whether the individual data d_i is obsolete or not. It computes two main quality scores:

- m_{d_score}: reflecting a weighted average of the quality metrics of d_i (cf. line 6 in algorithm 1).
- $m_{D\_score(d_i)}$: representing the variation of the data source's quality metrics resulting from the collection of d_i (cf. line 7-8 in algorithm 1).

Based on the thresholds assigned to each of the two scores, d_i is acknowledged as either of low quality or not (cf. line 9-10 in algorithm 1). Subsequently, each d_i is labeled as potentially obsolete or not whereby the obsolete ones get appended to an obsolescence data pool. Once all the data of a single device have been evaluated, the respective pool undergoes an overall check at the following data item layer.

Algorithm 1: Individual Data Quality Check

Inputs : d_i, // current individual data
d_{i-1}, // individual data at a previous time step
th_1, th_2, // thresholds
k_1, k_2, k_3, k_4 // weights

Output : $is\_obsolete$ // indicates whether the d_i is obsolete or not

Variables: $m_d\_score, m_D\_score(d_i), m_D\_list, x$

// Variables initialisation
1 $m_d\_score = 0$; // score of d_i quality metrics
2 $m_D\_score(d_i) = 0$; // score of the impact of d_i on D
3 $m_D\_list = [a, av, cr, ct, cn, fc]$; // list of the m_D metrics
4 $is\_obsolete = False$; /* Boolean variable that specify if data is obsolete or not */

// Calculate the popularity
5 $d_i.m_d.pl = \frac{d_i.an}{an_{sum}}$
/* m_d_score is calculated using weighted average */
6 $m_d\_score = k_1 \times d_i.m_d.ac + k_2 \times d_i.m_d.pl + k_3 \times d_i.m_d.r + k_4 \times d_i.m_d.t$;
7 **for** x ***in*** m_D_list **do**
// Score of m_D evolution between two consecutive individuals
8 $m_D\_score(d_i) += \frac{d_i.m_D.x - d_{i-1}.m_D.x}{d_{i-1}.m_D.x} \times 100$;

// Checking if scores are exceeding the thresholds
9 **if** $m_d\_score < th1$ ***Or*** $m_D\_score(d_i) < th2$ **then**
10 $is\_obsolete = True$;
11 **return** $is\_obsolete$;

b) Data item layer: This layer takes as input the time settings $[t_0, t_1]$ and a set of devices $\{d_1, d_2 \ldots d_n\}$ as explained earlier. In this context, for each device, we first retrieve the targeted set of data $[d_j, d_i]$ from the storage; sensor's data collected between t_0 and t_1 inclusive. Next, all d in $[d_j, d_i]$ goes iteratively through the data individual level which results in an obsolete data pool. This pool is necessary for applying the algorithm "Data Item Overall Check" (algorithm 2). It analyzes the device's overall obsolescence; a step that aims to confirm whether only the individual data is of low quality, or the data item it belongs to (cf. line 2-4 in algorithm 2).

Algorithm 2: Data Item Overall Check

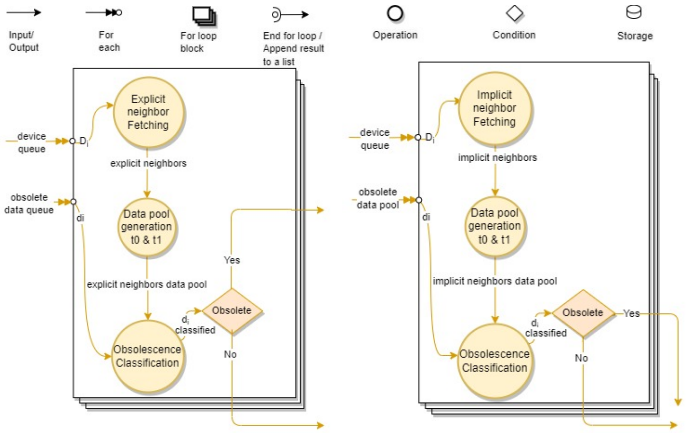
Input : D, // data item
OD, // obsolete data pool
th // threshold

Output : $is\_overall\_obsolete$// indicates whether the overall data item is obsolete or not

Variables: $percentage\_obsolete$// % of obsolete data contained in D

// Variables initialisation
1 $percentage\_obsolete = 0$;
2 $is\_overall\_obsolete = False$;
// Calculating the percentage of obsolete data in the data item
$percentage\_obsolete = \frac{size(OD)}{size(D)} \times 100$;
3 **if** $percentage\_obsolete > th$ **then**
/* Checking if the percentage of data obsolescence in the data item exceeds the threshold */
4 $is\_overall\_obsolete = True$;
5 **return** $is\_overall\_obsolete$;

2) Parent Zone Level: If the device level shows that d_i is of low quality, this is not sufficient to approve its obsolescence. Accordingly, this level checks the validity of the obsolescence data pool, generated at the device level, with respect to the device's hosting zone. To do so, each d_i belonging to the obsolete data pool gets checked against its devices' neighborhood within the same zone (i.e., explicit neighbors). It is necessary to perform the latter due to the possibility of an individual being obsolete to the data item it belongs to, but rather



(a) Parent zone level (b) Environment level

Fig. 6: Zone and environment detection levels

complementary and reasonable to the devices' neighborhood. This is accomplished via the parent zone dimension that passes every device containing data of low quality through several phases (cf. Fig. 6a) as detailed in the following.

a) Explicit neighbors fetching: As aforementioned, an obsolete individual data is to be checked with other related devices within the same hosting zone to verify the validity of the previously processed obsolescence check. Accordingly, the first step within the parent layer is to retrieve the explicit neighbors (i.e., interrelated devices located in the same zone) of the device D_i undergoing the obsolescence check. Based on the device clustering results, devices from the same zone belonging to D_i's cluster group are fetched.

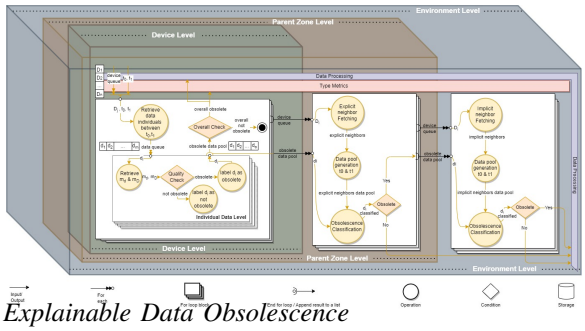
b) Data pool generation: In this step, a data pool is created and filled with the data individuals of every explicit neighbor of D_i. Namely, this pool represents a collection of explicit data sensed within the $[t_0, t_1]$ time-frame. In the next step, each obsolete individual data collected by D_i is analyzed against the pool to approve or disapprove its obsolescence.

c) Obsolescence classification: The parent zone architecture is given a queue of low quality data to check against the data pool created earlier in part b. At the device level, d_i was proved to be obsolete; however, its mandatory to check if it might be comprehensive with the neighboring devices. For instance, consider cameras C2 and C3 of zone2 (cf. Fig. 1). These two cameras might help complement one another to get a comprehensive view of the whole room specs. In other words, while C2 might miss some data due to a blind spot, C3 can come in handy if it is collecting data from such areas. This reiterates how it is possible for one piece of data to be obsolete at the device level, but not with respect to other devices. Accordingly, to validate if an obsolete data is obsolete at the parent zone level, the individual undergoes classification machine learning algorithms with respect to the data pool. This results in either: (1) Processing Data: data proves to be not obsolete and sending it to to check what might have caused its obsolescence at the device level, or (2) Going up the hierarchy (i.e., environment level): data proves to be obsolete and is sent for obsolescence check with respect to other zones.

3) Environment Level: If both device and zone levels prove d_i to be obsolete, the environment layer (cf. Fig. 6b) gets executed as the confirmation coating. Correspondingly, in this level, the obsolescence of d_i is checked with respect to the neighboring zones of the hosting zone. In other words, each d_i gets evaluated against its device's neighbors located across different but compatible zones (i.e., implicit neighbors). That is the case as when a device is deployed, the coverage area is based on, but not limited to, the zone's physical barrier. In this dynamic connected environment approach, a device can sense values from another zone if they are separated by what doesn't interrupt the sensing procedure (e.g., glass or virtual wall). Going back to zone2's C2 and C3 cameras (cf. Fig. 1), we notice camera C3 is located in zone 3 but is gathering some information from zone 2. Consequently, C3 might be collecting information proving C2's potentially obsolete data as significant or confirming its obsolescence. In essence, the same process applied at the parent level (neighbor fetching, pool generation, and obsolescence classification) is applied at the environment level. However, at the environment level, the neighbor fetching is concerned with implicit devices; devices from the different zones belonging to D_i's cluster group.

All the aforementioned levels work hand in hand providing one complete detection architecture depicted in Figure 7. In essence, each data sensed passes through device, parent zone, and/or environment level to check its obsolescence. Furthermore, it undergoes an additional service (i.e., explainable data obsolescence) that is essential before its final treatment.

Fig. 7: Obsolescence detection architecture



D. Explainable Data Obsolescence

The detection architecture presented earlier classifies data into obsolete or not. However, relying solely on this categorization is insufficient for adjusting obsolescence efficiently; additional strategies are required to address this sophisticated issue appropriately. In this context, this section introduces a solution that revolves around pinpointing the source responsible for producing obsolescence; "explainable data obsolescence". This verification process plays a crucial role in properly reducing concerns related to obsolescence. Obsolescence can fall into one or a combination of the following categories:

- Temporal obsolescence: poor quality is related to the time at which the individual data was collected.
- Spatial Obsolescence: poor quality is related to the location from which the individual data was collected.
- Contextual Obsolescence: poor quality is related to the content of the individual data.

Similar to the detection architecture, the explainable data obsolescence is processed at three dimensions (i.e., device, zone, environment) forming three split parts: (1) Part 1: type check with respect to the device it belongs to, (2) Part 2: with respect to the explicit devices within the same zone, and (3) Part 3: with respect to the implicit devices from neighboring zones. This module is established at the end of each level. The latter requires as input the classified data with their corresponding metadata (i.e., collection time, spatiality, attribute) and outputs the obsolescence grounds. The aim of this module is to check what's in common between the data, it inspects the data of every level and checks whether the obsolescence is occurring during a periodic range of time, and/or within a particular common spatial location, and/or for one specific attribute. With each shift to a higher level (e.g., device to zone level), the number of metadata increments, resulting in a higher number of features, which subsequently leads into more powerful and semantic results. For instance, after implementing this process on the camera C1, illustrated in the motivating scenario (cf. Fig. 1, Section II case 2), obsolete data was noticed to share common spatial properties; namely, they originate from the 120-degree section of the camera's rotation. This helps infer how this spatial section is the potential source of obsolescence accordingly marked for inspection and treatment.

V. Conclusion

Data integrity faces a critical research gap especially when it comes to data obsolescence - when data becomes insignificant towards its original purpose - and its detection. Thinking of data obsolescence as an independent terminology, this paper provides connected environments with a dimensional architecture to detect obsolescence using predefined data quality metrics. Namely, sensed data initially passes through a quality assessment phase after which it is checked against device, zone, and environment layers for obsolescence. With the detection architecture examined, the next step involves providing the experimental results on real deployed devices. The detection architecture this paper proposes relies on two main settings: (1) time span $[t0, t_1]$, and (2) a queue of sensors to check obsolescence for. Accordingly, as future work, it is required to provide a methodology through which the inputs are automatically chosen and triggered to schedule obsolescence detection. In other words, the current hierarchy needs to be topped with a recommendation system for a more productive selection, detection, and treatment. Speaking of treatment, this layer is to be provided with its approaches of handling and solving obsolete data. The current architecture productively measures obsolescence within data and requires two other systems to provide its input and handle its output of obsolescence.

References

[1] V. Moustaka, A. Vakali, and L. G. Anthopoulos, "A systematic review for smart city data analytics," *ACM Computing Surveys (cSuR)*, vol. 51, no. 5, pp. 1–41, 2018.

[2] E. Al Nuaimi, H. Al Neyadi, N. Mohamed, and J. Al-Jaroodi, "Applications of big data to smart cities," *Journal of Internet Services and Applications*, vol. 6, no. 1, pp. 1–15, 2015.

[3] R. Chbeir, E. Mansour, S. Allani, T. Yeferny, J.-R. Richa, F. Yessoufou, and S. Sellami, "Opencems: An open solution for easy data management in connected environments," in *Transactions on Large-Scale Data-and Knowledge-Centered Systems LII*. Springer, 2022, pp. 35–69.

[4] Y. Ali, A. Sharma, M. M. Haque, Z. Zheng, and M. Saifuzzaman, "The impact of the connected environment on driving behavior and safety: A driving simulator study," *Accident Analysis & Prevention*, vol. 144, p. 105643, 2020.

[5] R. Novotnỳ, R. Kuchta, and J. Kadlec, "Smart city concept, applications and services," *J. Telecommun. Syst. Manag*, vol. 3, no. 1, pp. 2167–0919, 2014.

[6] P. Singh and P. Sandborn, "Obsolescence driven design refresh planning for sustainment-dominated systems," *The Engineering Economist*, vol. 51, no. 2, pp. 115–139, 2006.

[7] J.-R. Richa, "Data quality computation for obsolescence detection within connected environments," in *International Conference on Innovations in Intelligent SysTems and Applications (INISTA)*. IEEE, 2023.

[8] V. Alieksieiev, G. Ivasyk, V. Pabyrivskyi, and N. Pabyrivska, "Big data aggregation algorithm for storing obsolete data," *Industry 4.0*, vol. 3, no. 1, pp. 20–22, 2018.

[9] L. C. Giralte and J. L. M. Gomez, "Managing data obsolescence in relational databases," Mar. 31 2020, uS Patent 10,606,835.

[10] M. Finger and F. Da Silva, "Temporal data obsolescence: modelling problems," in *Proceedings. Fifth International Workshop on Temporal Representation and Reasoning (Cat. No. 98EX157)*. IEEE, 1998, pp. 45–50.

[11] A. Glushkov, S. Belyakov, and M. Belyakova, "Intellectual obsolescence detection method of spatial data using historical data," in *2017 IEEE 11th International Conference on Application of Information and Communication Technologies (AICT)*. IEEE, Sep. 2017.

[12] N. Escribano, A. H. Ariño, and D. Galicia, "Biodiversity data obsolescence and land uses changes," *PeerJ*, vol. 4, p. e2743, 2016.

[13] S. Chaieb, B. Hnich, and A. B. Mrad, "Data obsolescence detection in the light of newly acquired valid observations," *Applied Intelligence*, vol. 52, no. 14, pp. 16 532–16 554, Mar. 2022.

[14] A. Habermann, "Automatic deletion of obsolete information," *Journal of Systems and Software*, vol. 5, no. 2, pp. 145–154, May 1985.

[15] T. Rousselin, K. Guérin, and N. Saporiti, "Definition of an alarm system to assess the obsolescence of african spatial data infrastructures."

[16] O. Bashir, "Managing obsolescence in information technology," in *Working document presented in the National IT conference*, 2000.

[17] D. De Cremer, B. Nguyen, and L. Simkin, "The integrity challenge of the internet-of-things (iot): on understanding its dark side," *Journal of Marketing Management*, vol. 33, no. 1-2, pp. 145–158, 2017.

[18] R. P. Dameri *et al.*, "Searching for smart city definition: a comprehensive proposal," *International Journal of computers & technology*, vol. 11, no. 5, pp. 2544–2551, 2013.

[19] N. Chamara, M. D. Islam, G. F. Bai, Y. Shi, and Y. Ge, "Ag-iot for crop and environment monitoring: Past, present, and future," *Agricultural systems*, vol. 203, p. 103497, 2022.

[20] P. A. Sandborn, F. Mauro, and R. Knox, "A data mining based approach to electronic part obsolescence forecasting," *IEEE Transactions on Components and Packaging Technologies*, vol. 30, no. 3, pp. 397–401, 2007.

[21] F. A. Schulze, H.-K. Arndt, and H. Feuersenger, "Obsolescence as a future key challenge for data centers," in *Advances and New Trends in Environmental Informatics: Digital Twins for Sustainability*. Springer, 2021, pp. 67–78.

[22] F. Romero Rojo, R. Roy, and E. Shehab, "Obsolescence management for defence and aerospace systems: state of the art and future trends," *Int J Adv Manuf Technol*, vol. 49, no. 9-12, pp. 1235–1250, 2010.

[23] P. Sandborn, "Software obsolescence-complicating the part and technology obsolescence management problem," *IEEE Transactions on Components and Packaging Technologies*, vol. 30, no. 4, pp. 886–888, 2007.

[24] B. Bartels, U. Ermel, P. Sandborn, and M. G. Pecht, *Strategies to the prediction, mitigation and management of product obsolescence*. John Wiley & Sons, 2012.

[25] C. Déméné and A. Marchand, "L'obsolescence des produits électroniques: des responsabilités partagées," *Les ateliers de l'éthique*, vol. 10, no. 1, pp. 4–32, 2015.

[26] I. Trabelsi, M. Zolghadri, B. Zeddini, M. Barkallah, and M. Haddar, "Prediction of obsolescence degree as a function of time: A mathematical formulation," *Computers in Industry*, vol. 129, p. 103470, 2021.