Challenges and opportunities in designing data structures for a smart city

Costel CIUCHI,
PhD, Assoc. Prof., Faculty of Electronics, Telecommunications and Information Technology, National University of Science and Technology POLITEHNICA Bucharest, Romania
costel.ciuchi@upb.ro

Alin BONCIU,
Faculty of Automatic Control and Computer Science, National University of Science and Technology POLITEHNICA Bucharest, Romania
alin.bonciu@stud.acs.upb.ro

Ana-Maria DINCĂ,
Faculty of Electronics, Telecommunications and Information Technology, National University of Science and Technology POLITEHNICA Bucharest, Romania
ana_maria.dinca@stud.etti.upb.ro

Mihai STOIAN,
Faculty of Automatic Control and Computer Science, National University of Science and Technology POLITEHNICA Bucharest, Romania
mihai.stoian1609@stud.acs.upb.ro

Abstract
The Internet of Things generates a vast amount of data in smart cities. However, with the increasing popularity of cloud storage and the ever-improving hardware, data storage is no longer a problem. Despite this, there may still be a need to conduct research and analyze aspects such as data variety and velocity to address some technological issues. This paper explores the concepts of normalization and denormalization in SQL and NoSQL databases, highlighting their advantages and disadvantages. It emphasizes the importance of carefully designing data structures in smart city applications. It is essential to consider the best approach to organize and manage information, taking into account the various areas of interest. To identify relevant data, analytics techniques are required to discover patterns, insights, and trends from large and complex data sets. One way to do this is through AI linked with Machine Learning. This approach can provide a better understanding of policy formulation, resource allocation, infrastructure planning, and service optimization. Additionally, we present a case study of a smart city application called SmartSantander, which illustrates how the use of NoSQL databases and denormalization can improve query performance. This could be a solution for addressing the data velocity problem. The article also emphasizes the importance of interoperability and cybersecurity in smart city data structure design. It highlights the use of JSON as a data interchange format and the need for secure data warehousing to protect sensitive information. Finally, our goal is to outline a solution that a smart city could implement to manage data effectively and securely. We propose using a hybrid approach that satisfies all its needs.

Keywords: database principles, big data, JSON, NoSQL.

1. Introduction
The Internet of Things (IoT) represents the solution for something that our society has been looking forward to since the late twentieth century: the concept of “Smart City”. In the beginning, to process data structures, the US Community Analysis Bureau started using databases, so the term “Smart City” was developed [1]. Nowadays, smart city architectures work around the same principle: data collection. Let’s see how the technology changed and aligned its lifespan until our days. The majority of databases followed the relational model.

However, in the following years, there has been a notable increase in the popularity of NoSQL databases, marking significant changes in the landscape of database technologies. We are living in a time of transition, in which questions such as: "*Which is better*?" or "*Why don't we use only NoSQL Databases?*" emerge.

From 1970 (implementation of the first relational DB models using the SEQUEL language) until 2006 (development of the first table structured on NoSQL principles – Google Big Table), SQL databases have been the benchmark of data storage and retrieval, offering well-defined schemas and ACID (Atomicity, Consistency, Isolation, Durability) compliance. However, the rigid structure of SQL databases poses limitations when confronted with the diverse and unstructured data characteristic of IoT-generated datasets. On the other hand, NoSQL databases, designed to handle vast amounts of unstructured data, provide a more flexible and scalable alternative. The choice between these two paradigms becomes particularly crucial in the context of smart cities, where the diversity of data types, the velocity of data streams, and the need for real-time analytics are key factors in determining which is the best choice.

2. SQL / NoSQL or both?

For the longest time, designing a persistence mechanism for a software application was limited to selecting one of the relational databases existent on the market, that could only be differentiated by performance criteria, the complexity of the query language and the costs. Over the years, software platforms and their associated business flows have evolved significantly, becoming increasingly sophisticated with each passing decade (see Figure 1). This evolution has led to a growing demand for additional resources, more powerful technologies, faster query executions, and the introduction of entirely new persistence features [2]. As businesses have expanded and their requirements have become more complex, software platforms have had to adapt to meet these changing needs. This has resulted in the development of more robust and resource-intensive systems that can handle larger workloads and process data more efficiently.

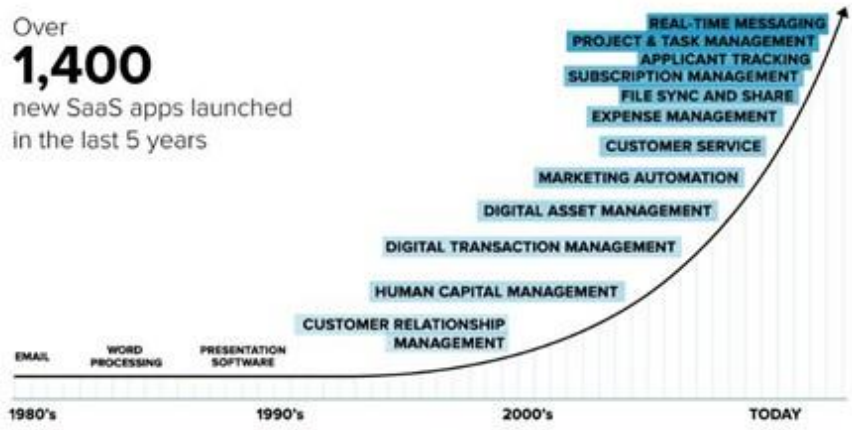

Fig. 1. Evolution of software platform capabilities and operations.
Source: [2]

To keep up with the demands of modern business operations, software platforms have incorporated new technologies that can deliver faster query executions and quick data retrieval and processing, enabling businesses to make informed decisions promptly. Overall, software platforms and workflows must offer enhanced collaboration, mobility, user-friendliness, and productivity, making it a requirement to operate more efficiently in an increasingly digital world.

One of the more difficult demands to tackle was the storage of non-strictly structured data and big objects, as they did not fit in the data models persisted by the relational database management systems (RDBMS). The exponential evolution of the volume of data leads to significant operational and maintenance challenges and pressures. Traditional relational databases have limitations when it comes to efficient scalability because they store all data in a single box, which can pose challenges for users.

Therefore, a new type of database emerged in 2006, the Google Bigtable, the first non-relational database (NoSQL) [3] .This new approach was quickly adopted for business development and exploration, and in 2007 Amazon researchers published a paper on the technical details of Dynamo, which laid the prototype of a non-relational database [4].

This innovative approach was developed using a robust set of distributed system principles, leading to the creation of a database system that is both highly scalable and exceptionally reliable. Nowadays, non-relational databases have evolved into 4 separate types, depending on how the data is modelled and saved: document databases, key-value databases, wide-column store databases, and graph databases [5]. The objects used to structure NoSQL and their corresponding usage are outlined in Table 1.

Table 1. NoSQL Database Types

NoSQL database types	Structure
document databases	consists of sets of key-value pairs stored in a document. These documents are basic units of data which you can also group into collections (databases) based on their functionality.
key-value databases	dictionary data structure and consists of a set of objects that represent fields of data. Each object is assigned a unique key. To retrieve data stored in a particular object, you need to use a specific key. In turn, you get the value (i.e. data) assigned to the key.
wide-column store databases	data is stored and grouped into separately stored columns instead of rows. Such databases organize information into columns that function similarly to tables in relational databases.
graph databases	Graph databases use flexible graphical representation to manage data. These graphs consist of two elements: Nodes (for storing data entities) and Edges (for storing the relationship between entities)

Source: Authors' own work

The persistence mechanism is a vital component of any software platform, as it directly influences the performance metrics. Consequently, it is compulsory to analyze the business requirements, in the interest of reaching a deep understanding of the features that are to be implemented, and to design an efficient and reliable database architecture. Traditional relational databases are primarily designed to operate on single servers, which can result in performance challenges when dealing with substantial data volumes. On the other hand,

NoSQL databases are specifically engineered to scale horizontally across multiple servers. This horizontal scalability enables them to efficiently manage and process extensive amounts of data without encountering performance issues.

In the context of smart cities, the information is modelled in various and different data structures, the data is continuously generated and sent for processing, and therefore, there have been identified business flows that are both built on flexible data models and handle massive amounts of data and should be managed using a NoSQL database [6]. A NoSQL document database such as MongoDB would fit flawlessly in a scenario where the data is sent through JSON files, and it stores its data in BSON documents, which are the binary representation of JSON files [7], and the information persistence operations should require little to no additional processing. While traditional relational database management systems cannot store Big Data efficiently and should be avoided for such applications, it is unlikely that a software platform that manages a smart city would implement only features that require the use of non-relational databases. Normally, the generated data is stored, and processed and the refined data has to be stored in a more organized data structure, which is accommodated better by a light RDBMS, such as PostgreSQL.

The research has underlined that to build an efficient and reliable database system for a platform that handles and manages Big Data for a Smart City, the architecture should include both, a non-relational database, such as MongoDB, OrientDB, RedisGraph, or Neo4j, for the entities that have a dynamic structure, and a lightweight traditional relational database, such as MariaDB, PostgreSQL, for refined data, or other information that is described by a rigid data model, such as a user profile.

3. The 3Vs of big data (Volume, Velocity, and Variety)

It is crucial for a smart city to have a significant amount of data to be analysed by city administrators to identify patterns, trends, and areas for improvement, leading to more sustainable and livable urban environments. Additionally, the availability of data enables the development of innovative technologies and services that can further enhance the quality of life for residents and promote economic growth. But the data is diverse and dynamic and this includes data related to traffic flow, energy consumption, and public transportation, among others. The essential attributes for data in a smart city context are the 3Vs of big data, which are *Volume*, *Velocity*, and *Variety*.

3.1. First V (Volume)

The astonishing amount of data that could be generated by IoT devices in 2025 is predicted to be 79.4 zettabytes. To put this number into perspective, let's think that an average computer has a storage capacity of around 512GB. Considering this amount of data as being a single drop of water, 79.4 zettabytes would be equivalent to around 3 Olympic pools. As mentioned before the data quantity will soon no longer be a problem because of the technological developments. Looking below (Figure 2) we will see a graph from a popular ML article written by Microsoft researchers [8], in which they worked on natural language disambiguation problems (*homophones = each of two or more words having the same pronunciation but different meanings, origins, or spelling, for example, new and knew).

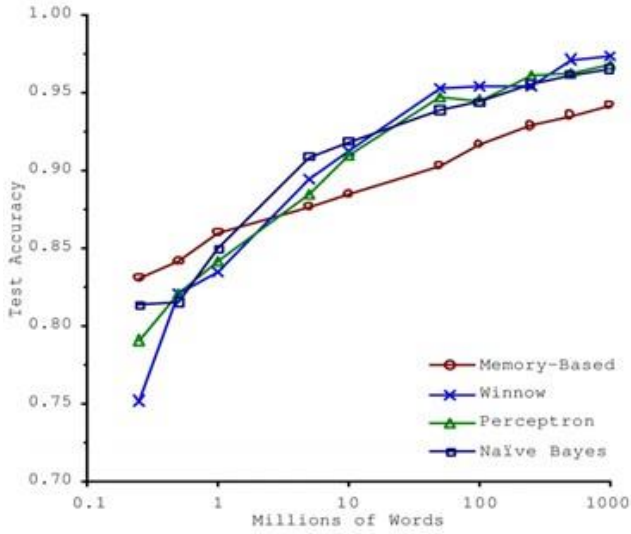

Fig. 2. Machine learning: Learning Curves for Confusion Set Disambiguation.
Source: [8]

In this graph we can see different Machine learning (ML) algorithms and that the larger the dataset, the precision increases. Should we just collect more data, in order to make the models more accurate in a smart city? We can generate our own data that is also relevant for our model. For example, there are Machine learning models that can do image classification, in this scenario we can just rotate the image that is fed to the model to obtain new data.

3.2. Second V (Velocity)

A Machine learning model after it is trained, is a set of parameters that define behavior. In some cases, it remains the same, but sometimes in the context of a smart city, it is not. Because of its dynamic nature, we have to consider that some of the models need to be retrained. Now computational complexity needs to be taken into consideration.

In a smart city, data comes at a faster pace than it can be processed. We can store the data, then process it later and also speed up the procedure with parallel processing. Data cleaning is essentially correcting errors, inconsistencies and inaccuracies by standardizing and normalizing data. We still need to clean data, although we would store it in a NoSQL database because providing uncleaned data to a ML model will affect its performance and reliability. Also, dimensionality reduction is a key improvement, consisting of transforming the original set of characteristics into a new set, which has reduced dimensions (for example, Principal Component Analysis by using linear combinations of the original characteristics). This leads to reduced computational complexity but also affects the performance of the model, as in the graph [9] shown in Figure 3.

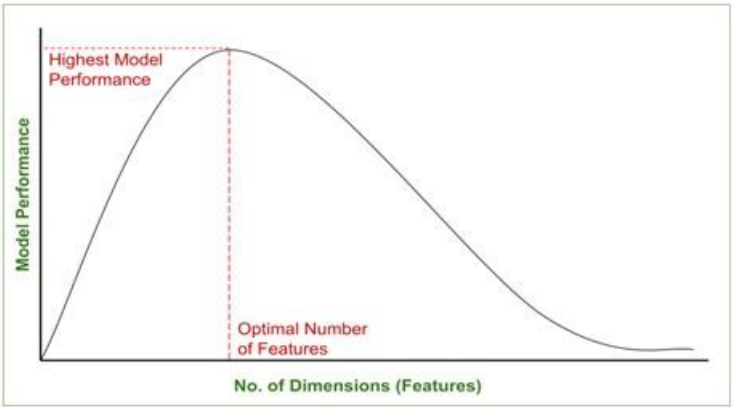

Fig. 3. Model Performance vs No. of Dimensions (Features) – Curse of Dimensionality.
Source: [9]

3.3. Third V (Variety)

The variety of the data comes from the fact that it can be structured, unstructured and semi-structured data. Structured data is often found in SQL databases as it fits neatly into tables, the semi-structured data consists of video files, images, and text documents. So unstructured data needs to undergo processing to extract the important features. The unstructured data consists of files such as JSON, XML, and BJSON and it is most often used in NoSQL databases.

From another perspective, the data could also be categorized as temporal data, geospatial data, multimedia data, numeric data and text data. The sources vary and the opportunities are endless. Having access to abundant data provides a better view of insights, giving the opportunity for a clearer understanding of the complex urban ecosystem that a smart city implies. Machine learning can determine correlations between the data that the human eye cannot perceive, but it is our duty to determine how to use this to our advantage to get closer to the ideal of a smart city.

Before moving forward, the key points of this would be that all the challenges that come when working with big data have a solution and overcoming them is taking our society a step closer to the concept of a smart city. The interconnection between big data, SQL, and NoSQL databases will continue to shape the future of smart cities. As we anticipate further improvements in technology and a growing influx of data sources, the importance of scalable database solutions cannot be overlooked.

4. Case study (some real example/s, e.g., Smart Santander)

In this section, we delve into practical, real-world applications of NoSQL databases in the realm of smart cities, with a focus on the SmartSantander project. SmartSantander proposes an experimental research facility in support of typical applications and services for a smart city. The project envisions the deployment of 20,000 sensors in Belgrade, Guildford, Lübeck and Santander, exploiting a large variety of technologies [10]. This case study exemplifies how NoSQL technology is being leveraged to address the complexities and challenges inherent in managing the voluminous and diverse data generated in a smart

urban environment. Through the SmartSantander project, we aim to highlight the practicalities, benefits, and innovations brought forth by the integration of NoSQL databases in large-scale, data-intensive smart city applications.

The SmartSantander project, a large, open and innovative city-scale experimental research initiative, exemplifies the vital role of NoSQL databases in managing the complex data ecosystem of a smart city. This unique venture aims to transform the city of Santander, Spain, into a smart city by deploying around 12,000 sensors across diverse urban environments. These sensors, utilizing various technologies, support a wide array of smart city applications, highlighting the project's scalability and heterogeneity [11].

The architecture of SmartSantander is designed to be both scalable and trustable, meeting the primary requirements for a real-world IoT experimental platform. This encompasses incorporating essential building blocks necessary for the IoT framework. SmartSantander focuses on validating approaches to the IoT architectural model, evaluating key components of IoT architecture, and assessing the social acceptance of IoT technologies and services. This comprehensive approach underscores the importance of flexible and scalable data management systems, like NoSQL databases, in smart city contexts.

In this context, the massive data sets generated by the extensive sensor network underline the need for databases capable of efficiently handling large and diverse data sets, a hallmark of NoSQL databases. Their inherent design to accommodate a variety of data types and formats makes them particularly suitable for the varied data generated in smart cities. Additionally, the real-time data analysis required for services such as traffic management and emergency response necessitates the speed and efficiency provided by NoSQL databases in data processing and querying. The platform’s architecture is delineated into a three-tiered structure (see Figure 4), encompassing IoT devices, gateways, and server tiers, each serving distinct functions within the smart city infrastructure [12].

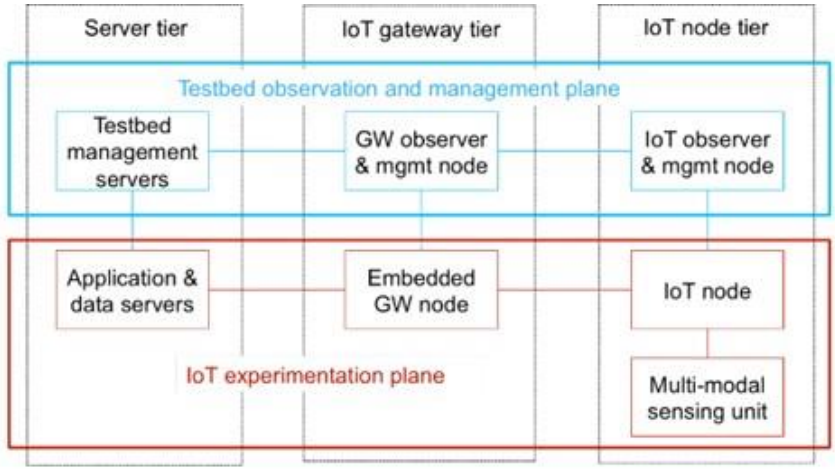

Fig. 4. Logical separation of 3-tier node architecture into a testbed observation and management and an experimentation plane.
Source: [12]

The platform's architecture is designed with a three-tiered structure, consisting of IoT devices, gateways, and server tiers. Each tier has its own specific role and function within the smart city infrastructure:

- *IoT Device Tier*: represents the primary experimental base, includes IoT devices and IoT nodes witch and are responsible for collecting data from various sources within the city. To ensure reliable operation in outdoor conditions, these devices are strategically placed in inaccessible locations and equipped with dual power supplies and communication interfaces, alongside robust management procedures for rapid malfunction detection.
- *IoT Gateway Tier*: serving as an intermediary between the IoT devices and the server tier, this tier links edge IoT devices to the core network infrastructure. Gateways aggregate and preprocess the data collected from the devices before transmitting it to the server tier. They also provide security and connectivity functionalities, ensuring reliable and secure data transmission.
- *Server Tier*: is responsible for processing and analyzing the collected data, it comprises server devices connected directly to the core network. Hosting IoT data repositories and application servers, this tier includes powerful servers and cloud infrastructure that can handle large volumes of data and perform complex analytics ensuring high reliability and availability. The server tier enables data storage, real-time analysis, and the generation of actionable insights that can be used for decision-making and optimizing city operations.

By organizing the architecture into these three tiers, the platform can efficiently manage and process the data collected from IoT devices, ensuring seamless communication, data integrity, and scalability within the smart city infrastructure. Also, this layer provides valuable insights for decision-making and resource optimization. The architecture's design is communication-agnostic, permitting diverse technological implementations. It's bifurcated logically into a *Testbed observation and management plane* and an *IoT experimentation plane*. The former manages dynamic, automated fault management and configuration, while the latter is used for configuring and controlling experiments through APIs. This logical separation does not imply physical segregation, allowing functionalities of both planes to coexist within a single device or to be physically separated to avoid performance impairment on constrained devices. This framework is pivotal in minimizing human intervention and enhancing the manageability of large-scale IoT infrastructures.

Transitioning from the broader perspective of the SmartSantander project's architecture, we delve into a more specific and concrete framework, focusing on the City Data and Analytics Platform (CiDAP) [13]. CiDAP is ingeniously designed as a middle layer interfacing between various data sources and the applications that serve a smart city. Its architecture consists of several critical components explained in Table 2.

Table 2. Base components in City Data and Analytics Platform (CiDAP)

IoT-Broker and IoT-Agents	These elements are pivotal for data collection. The IoT-broker harmonizes communication with IoT-agents, which function as intermediaries for different data sources, managing the intricate details of numerous sensor nodes.

Big Data Repository	This is the core where data storage is meticulously managed. The sensor data, harvested through IoT-agents, are securely stored as JSON documents within a NoSQL database. CouchDB has been selected for its robustness and suitability for handling the voluminous data characteristic of smart cities.
Big Data Processing	To handle the intensive data processing and analytics, this module leverages the power of a Spark cluster. This setup provides the necessary scalable and flexible computation resources to process and analyze large data sets effectively.
City Model Server	This server acts as the communication hub with external applications. It utilizes predefined City Model APIs to manage queries and subscriptions, facilitating a seamless flow of information between the platform and its users.

Source: Authors' own work

The data collection in CiDAP is diverse, encompassing semi-structured data like sensor information in JSON format and unstructured data such as social media texts and surveillance images/videos. This assortment of data is efficiently managed through the IoT-broker and IoT-agents. Data processing in CiDAP is categorized into two distinct types: internal and external. Internal processing is confined within the CouchDB database and mainly involves creating indexed views through map and reduce functions. In contrast, external processing, which encompasses more complex data analytics, is executed outside CouchDB, utilizing the extensive capabilities of the Spark cluster.

The CityModel API plays a crucial role in interfacing external applications with the CiDAP platform. It supports various types of queries, from simple ones that provide a snapshot of the latest sensor status to complex queries that delve into historical data. Additionally, it offers subscription services for continuous data updates. The IoT-broker, derived from NEC's open-source project Aeron, along with the CityModel server and platform management portal developed in JavaScript, based on the NodeJS platform, demonstrate the platform's versatility. The deployment of anomaly detection for traffic sensors and the integration of a comprehensive dashboard service highlight the platform's practical applicability in a real-world smart city scenario. Incorporated below is a diagram that visually represents the CiDAP architecture, illustrating the interconnectedness of its various components and the flow of data between them.

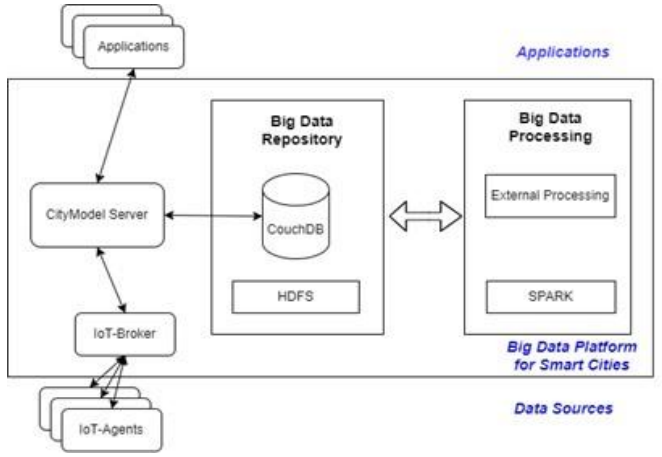

Fig. 5. CiDAP Architecture Diagram.
Source: [13]

This is achieved by including a broad set of applications that demonstrate the platform's utility and its impact on citizens, emphasizing the diversity, dynamics, and scale required for advanced protocol solutions. Based on this architecture, SmartSantander offers an Experimentation Facility among the scientific community, end users, and service providers including a set of applications that follow the development of measurements for utility and its impact on citizens, emphasizing the diversity, dynamics, and scale.

SmartSantander has become a model for smart cities worldwide for several key applications workflows and management that need to be implemented in all smart city frameworks [10]:

Table 3. SmartSantander applications

Environmental Monitoring	SmartSantander introduces a paradigm shift by deploying a vast network of low-cost IoT sensors, providing data on air quality, noise levels, and luminosity across extensive areas. While these sensors may not match the precision of traditional equipment, their widespread distribution allows for intelligent data processing to achieve sufficiently accurate environmental assessments.
Outdoor Parking Management and Driver Guidance	SmartSantander involves deploying ferromagnetic wireless sensors under the asphalt of parking bays in the city center. These sensors monitor parking space occupancy and connect to the internet through repeaters, enabling real-time dissemination of parking availability to drivers and traffic control centres. This system also aids municipal authorities in analyzing parking patterns and making informed decisions about parking provisions.
Parks and Gardens Precision Irrigation	SmartSantander has augmented existing automated irrigation systems in parks and gardens with IoT devices. These devices collect data on environmental conditions like air and soil temperature, humidity, leaf wetness, and rainfall. This precision approach replaces the traditional schedule-based systems, significantly enhancing irrigation efficiency and reducing water wastage.
Augmented Reality	This application involves enhancing the cityscape with IoT endpoints to provide context-sensitive information and services. Visitors interacting with these tags through their mobile phones receive information and services relevant to their location. For instance, tourists can enjoy an augmented reality experience with descriptions of monuments in their preferred language.
Participatory Sensing	Citizens' mobile phones double as sensors, contributing data like GPS coordinates, environmental conditions (noise, temperature), and direction to the SmartSantander platform. Users can subscribe to services like alerts for specific city events and contribute to the data pool by reporting occurrences, enhancing community engagement and responsiveness.

Source: Authors' own work

These real applications of the SmartSantander project showcase the transformative potential of IoT in urban environments. By integrating technology into everyday city operations and citizen engagement, SmartSantander leads the way in smart city development, enhancing urban living through technology-driven solutions. In conclusion, SmartSantander serves as a compelling case study for the application of NoSQL databases

in smart cities. Its scale, diversity, and technological sophistication underscore the advantages of using NoSQL databases in managing the complex and voluminous data inherent in smart city environments.

5. Conclusions

The evolution of devices and sensors in a city has led to a significant growth in data. As the volume, velocity, and variety of data continue to increase, there is a growing need for large, flexible, and open system databases. These databases play a crucial role in managing and accessing complex and large volumes of data, enabling efficient storage, retrieval, and analysis. By utilizing such databases, cities can effectively handle the diverse and dynamic nature of data generated by devices and sensors. The integration of both SQL and NoSQL database systems plays a crucial role in the advancement of data management within smart cities, contributing to the evolution of urban landscapes. Given the dynamic nature of smart cities, careful consideration is required when selecting a database architecture. SQL databases excel in managing structured data, providing robustness and reliability. On the other hand, NoSQL databases offer flexibility in handling unstructured and real-time data, accommodating the diverse and rapidly changing data generated in smart city environments. By leveraging the strengths of both SQL and NoSQL databases, smart cities can effectively manage and analyze data, enabling informed decision-making and facilitating the development of innovative solutions for urban challenges.

The SmartSantander project, particularly through the CiDAP framework, provides a vivid illustration of the practical applications of NoSQL databases in smart city environments. The strengths of NoSQL databases are clearly evident in this context: their ability to handle large volumes of diverse data efficiently, their scalability to accommodate the rapid expansion of urban IoT networks, and their flexibility in data modelling, which is crucial for processing the varied data generated by a smart city. These attributes make NoSQL databases particularly suitable for the dynamic and heterogeneous nature of smart city data. A lesson learned within the SmartSantander project was that the initial version of the database system was based on SQL relational databases (e.g. used to store resource descriptions) but to meet the new requirements, it was replaced with a MongoDB (NoSQL) to ensure flexible handling of various resource description documents and better performance [14] [15].

Also, there is a need to establish a universal standard that would increase the speed of the development process. Creating a standard means that all researchers work towards the same goal and after it is tested and implemented for the first time, the only thing left is to replicate that for another city from the European Union. By creating a common foundation, we can unlock the true potential of this data-driven system that could delve more into detail about the economic and social effects of implementing a benchmark [16]. On the other hand, some aspects should be taken into consideration, like resilience, reliability and security of NoSQL database systems. Cyber security and data privacy represent a concern, through securing and encrypt communications between IoT devices, sensors, fog and cloud architectures to avoid attacks on the city infrastructure and information theft.

In summary, while NoSQL databases offer significant advantages for smart city applications, there remains a need for ongoing innovation and development in areas of standardization, data integrity, and advanced analytics to fully realize their potential in this rapidly evolving field. The establishment of a standardized database framework propels the development of smart cities forward and sets the stage for interconnected smart cities taking humanity towards an intelligent and technology-driven future.

References

[1] "The Welding Institute. [Online]. Available: https://twi-global.com/technical-knowledge/faqs/what-is-a-smart-city".

[2] "D. Politis, "The SaaS-Powered Workplace: The Present & Future of Work". [Online]. Available: https://www.bettercloud.com/ monitor/saas-powered-workplace-multi-saasmanagement/.".

[3] "F. Chang, J. Dean, S. Ghemawat, W. Hsieh, D. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. Gruber, "Bigtable: A Distributed Storage System for Structured Data," OSDI'06: Seventh Symposium on Operating System Design and Implementation, Seattle, WA,".

[4] "G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, W. Vogels, "Dynamo: Amazon's highly available key-value store," ACM SIGOPS Oper. Syst. Rev., 41(6)," pp. 205-220, 2007.

[5] "S. Simic, "NoSQL Database Types," phoenixNAP Globat IT Services, 2020. [Online]. Available: https://phoenixnap.com/kb/nosql-database-types".

[6] "What is NoSQL?". [Online]. Available: https://www.mongodb.com/.".

[7] "Is JSON a NoSQL?". [Online]. Available: https://www.mongodb.com/.".

[8] "M. Banko and E. Brill, "Scaling to Very Very Large Corpora for Natural Language Disambiguation," In Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics, Toulouse, France," pp. 26-33, 2001.

[9] "Reduce Data Dimensionality using PCA – Python". [Online]. Available: https://geeksforgeeks.org/reduce-data-dimentionality-using-pca-python/.".

[10] "SmartSantander". [Online]. Available: https://smartsantander.eu/.".

[11] "SmartSantander Project". [Online]. Available: https://cordis.europa.eu/project/id/257992.".

[12] "L. Sánchez, L. Muñoz, J. Galache, P. Sotres, J. Santana, V. Gutierrez, R. Ramdhany, A. Gluhak, S. Krco, E. Theodoridis, D. Pfisterer, SmartSantander: IoT Experimentation over a Smart City Testbed," Computer Networks, 10.1016/j.bjp.2013.12.020," 2013.

[13] "B. Cheng, S. Longo, F. Cirillo, M. Bauer, E. Kovacs, "Building a Big Data Platform for Smart Cities: Experience and Lessons from Santander," (BigData2015-5084).10.1109/ BigDataCongress.2015.91," 2015.

[14] "V. Tsiatsis, et al., "The SENSEI Real World Internet Architecture," IOS Press, Towards the Future Internet," pp. 247-256, 2010.

[15] "S. Jokić, et. al., "Evaluation of a Document Oriented Resource Directory Performance", in Proceedings of the Telecommunication Forum, November 2011, Belgrade.".

[16] "Online platforms: Economic and societal effects," STUDY Panel for the Future of Science and Technology, European Parliamentary Research Service, Scientific Foresight Unit (STOA) PE 656.336, March 2021. [Online]. Available: https://www.europarl.europ".