Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

A DISTRIBUTED SERVICE PLATFORM FOR MANAGING STREAMING DATA IN SMART CITIES

A Citizen-Centric Approach to Improve Urban Participation

Doctoral Dissertation
submitted at Universität Hamburg

Philipp Kisters

Faculty of Mathematics, Informatics and Natural Sciences
Department of Informatics
Distributed Operating Systems

Hamburg, September 2024

REVIEWER:

Prof. Dr. Winfried Lamersdorf

Prof. Dr. Lars Braubach

DISPUTATION DATE: 16.12.2024

Theorie und Praxis sind eins wie Seele und Leib,
und wie Seele und Leib liegen sie großenteils miteinander in Streit.

— Marie von Ebner-Eschenbach

ZUSAMMENFASSUNG

Smart Cities zielen darauf ab, das städtische Leben durch verbesserte Effizienz, Nachhaltigkeit und Sicherheit zu optimieren. Allerdings haben die derzeitigen, vorwiegend technologiegetriebenen, Smart-City-Modelle Schwierigkeiten, signifikante Verbesserungen der angestrebten Ziele, im Besonderen in Bezug auf Nachhaltigkeit, zu demonstrieren. Gleichzeitig wächst die öffentliche Skepsis in Bezug auf Datenschutz und Überwachung. Diese Arbeit schlägt einen Wechsel vor, von einem technologiezentrierten hin zu einem bevölkerungszentrierten Ansatz für Smart Cities, der auf bestehenden, von der Bevölkerung betriebenen Sensoren, basiert. Dieser Ansatz stärkt die Mitbestimmungsrechte der Bevölkerung und fördert deren aktive Einbindung in urbane Räume, indem er den Bedarf an neuen Sensoren minimiert und ihnen ermöglicht, ihre Daten selbst zu verwalten.

Diese Dissertation befasst sich mit technischen und sozialen Herausforderungen bei der Umsetzung eines solchen dezentralen Datenraums. Zu den wichtigsten technischen Herausforderungen gehören die Sicherstellung der Interoperabilität zwischen heterogenen Datenquellen, das Finden relevanter Sensoren für stadtweite Dienste und die Bewertung der Datenqualität in einem dezentralen System. Soziale Herausforderungen konzentrieren sich auf die Wahrung der Datensouveränität, den Vertrauensaufbau durch Transparenz und die Einbeziehung technisch weniger versierter Personen.

Zur Bewältigung dieser Herausforderungen, liefert diese Arbeit drei wesentliche Beiträge. Erstens wird SkABNet, ein attributbasiertes Overlay-Netzwerk, eingeführt, das effiziente semantische Suchen ohne zentrale Instanz ermöglicht und den Suchaufwand um bis zu 90% reduziert. Zweitens wird ein Framework für datensouveränitätswahrende verteilte Vorverarbeitung entwickelt, das erlaubt, die Verarbeitung der eigenen Daten zu kontrollieren, bevor diese mit entfernten Diensten geteilt werden. Eine im Rahmen dieser Arbeit durchgeführte Anwendungsstudie zeigt, dass dieser Ansatz die Entscheidungsfindung für technisch weniger versierte Personen unterstützt und ihnen hilft, die Verwendung ihrer Daten besser zu verstehen. Schließlich wird eine Methode zur Kategorisierung von Datenströmen vorgeschlagen, die einerseits ermöglicht, gemeinsame Merkmale individuell platzierter Sensoren zu identifizieren, um Diensten bei der Bewertung der Datenqualität zu helfen. Andererseits können Mikroklimaereignisse identifiziert werden, die durch aktuelle Qualitätskontrollmechanismen entfernt werden. Diese Beiträge fördern und konkretisieren die Vision einer nachhaltigeren, inklusiveren und bevölkerungszentrierten Smart City.

ABSTRACT

Smart Cities aim to enhance urban living by improving efficiency, sustainability, and citizen engagement. However, current Smart City models, primarily driven by technology, have struggled to demonstrate substantial improvements in efficiency and sustainability while facing public skepticism regarding data privacy and surveillance. This thesis proposes a shift from technology-driven towards a citizen-centered approach for Smart Cities, leveraging existing citizen-operated sensors to create a more sustainable and inclusive urban environment. This approach enhances participatory rights and fosters active engagement with urban spaces by minimizing the need for new sensor installations and empowering citizens to manage their data.

This dissertation addresses the technical and social challenges associated with implementing such a decentralized data space. Key technical challenges include ensuring interoperability between heterogeneous data sources, discovering relevant sensor data, and assessing data quality in a decentralized system. Social challenges focus on maintaining citizen data sovereignty, building trust through transparency, and ensuring that non-technically trained citizens are included.

To address these challenges, this work consists of three key contributions. First, it introduces SkABNet, an attribute-based overlay network enabling efficient semantic search of distributed data streams without a central authority, reducing search overhead by up to 90%. Second, a data sovereignty-respecting framework for distributed preprocessing is developed, allowing citizens to control the processing of their collected data before sharing it with remote services. A user study demonstrates that this approach supports decision-making for non-technical users, helping them understand the usage of their provided data. Finally, a data stream categorization method is proposed, which, on the one hand, enables the identification of shared characteristics from individually placed sensors to help services rate the data quality. On the other hand, microclimate events can be identified to tackle local anomalies that current quality control mechanisms might remove. These contributions collectively advance the vision of a more sustainable, inclusive, and citizen-centered Smart City.

Alle Systeme sind falsch, die auf Beständigkeit der menschlichen Natur, nicht auf seiner Wandlungs- und Entwicklungsfähigkeit beruhen.

— Oscar Wilde

DANKSAGUNG

Diese Arbeit wäre ohne die Unterstützung und das Engagement vieler Menschen nicht denkbar gewesen. Ihnen allen möchte ich an dieser Stelle meinen aufrichtigen Dank aussprechen.

Mein besonderer Dank gilt meinem Doktorvater, Prof. Dr. Winfried Lamersdorf, der mir die Chance gegeben hat, viele spannende Themen als meine neuen Interessensfelder zu entdecken. Ich schätze die Freiheit, die er mir während der Forschung und auch der Lehre gab, durch die ich lernen konnte, wie viel Freude mir auch das Weitergeben von Wissen bereitet. Sein Feedback, seine Geduld und seine ermutigenden Worte haben diese Arbeit maßgeblich geprägt.

Ebenso möchte ich mich bei meinem Zweitgutachter, Prof. Dr. Lars Braubach, bedanken, der durch seine Bereitschaft, meine Arbeit zu begutachten, einen wichtigen Beitrag zur wissenschaftlichen Qualitätssicherung dieser Dissertation geleistet hat. Zudem bin ich dankbar, dass er sich auch kurzfristig immer Zeit für mich genommen hat.

Ein weiterer Dank gilt Prof. Dr. Janick Edinger, mich unter seine Fittiche genommen hat und der durch sein persönliches Engagement eine inspirierende Arbeitsumgebung geschaffen hat. Ich bin sehr dankbar für seine Unterstützung und für jedes Gespräch, dass wir noch nach 21 Uhr hatten, wenn alle Kinder endlich schliefen und wir in Ruhe forschen oder die Lehre planen konnten.

Mein Dank geht auch an meine (ehemaligen) Kolleginnen und Kollegen der Arbeitsgruppe, insbesondere Heiko, der mich heil durch die Heimarbeit in der Coronazeit gebracht hat. Für den Zusammenhalt bei kurzfristigen Deadlines, die Zusammenarbeit und die freundschaftliche Atmosphäre. Ihr alle habt diese Zeit bereichert und den manchmal steinigen Weg der Forschung durch Humor und Unterstützung deutlich erleichtert.

Ganz besonders möchte ich den wichtigsten Menschen in meinem Leben, meiner Familie, danken. Ohne Marikas bedingungslose Unterstützung, ihre Geduld und ihren Glauben an mich, hätte ich diesen Weg nicht erfolgreich gehen können. Mein Dank gilt auch den Großeltern meiner Kinder, die uns immer wieder den Rücken freigehalten haben und sich liebevoll mit um unsere Kinder gekümmert haben, wenn gerade wieder Land unter war.

Abschließend möchte ich allen Freunden danken, die mir während dieser Zeit zur Seite standen und immer für Spieleabende Zeit hatten und so für Ablenkungen außerhalb der Arbeit sorgten.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

ANN	Artificial Neural Networks
API	Application Programming Interface
CDN	Content Delivery Networks
DHT	Distributed Hash Table
DP	Density Peaks
DPC	Density Peak Clustering
DTC	Deep Temporal Clustering
DTW	Dynamic Time Warping
ECS	EU-Citizen.Science
ETL	Extraction, Transformation, and Loading
FCD	Floating Car Data
IoT	Internet of Things
IP	Internet Protocol
KP	Knowledge Processors
LSTM	Long-Short Term Memory
OWL	Web Ontology Language
PCA	Principal Component Analysis
P2P	Peer-to-Peer
PHT	Prefix Hash Tree
QaaS	Quality as a Service
RDF	Resource Description Framework
SIB	Semantic Information Broker
SkABNet	Attributed-Based SkipNet
SSN	Semantic Sensor Network
SVM	Support Vector Machine
UHI	Urban Heat Island
WSN	Wireless Sensor Network
W3C	Word Wide Web Consortium

1 INTRODUCTION

Smart Cities are increasingly important in today's urban development [232]. They promise to make cities more efficient, sustainable, and livable [115, 135]. Previous approaches often aimed at installing various sensors that collect, e.g., environmental, noise, and traffic data in public places. With these new insights into a city's resources, resource consumption, and the local environment, city officials advance technology-driven solutions. However, these Smart City concepts have yet to convincingly demonstrate their efficiency improvements and sustainability [260, 276, 277]. At the same time, public perception and skepticism towards technological surveillance are growing [195]. This is partly due to concerns that citizens are being degraded to passive data sources without control over their own data [84].

Due to these social aspects, a gradual shift from Smart City 1.0 (technology-driven) to Smart City 2.0 (citizen-centric) could be witnessed over the last decade. This development reflects the growing recognition that citizens should be actively involved in designing and managing their urban environment [110].

Simultaneously, the number of citizen-operated sensors is increasing due to various technological advancements in the smart home sector [235]. Additionally, more and more citizens carry sensors on their bodies or in their vehicles. These developments open up new possibilities for a citizen-centric Smart City, making it more sustainable by using these preexisting sensors and inclusive for its citizens by enabling them to participate actively.

For this reason, the following thesis proposes a citizen-centric approach that builds on citizens' existing sensors. This concept aims to create a more sustainable Smart City model. Resources are conserved by reducing the need to install new sensors and actively involving citizens from the beginning, strengthening citizens' participatory rights.

The engagement of citizens with their data, environment, and, ultimately, their city should be increased in two ways: First, citizens should be able to manage their own data and regain control over the use of their provided data. Second, they can provide their own so-called services, which are software applications that use the available data sources to offer added value to themselves and others. A data space, built upon citizens' sensors, is intended to create a community that exchanges information and insights about data collection, processing, and their use in different domains. Citizens actively participate in the co-creation of data collected citywide, allowing them to discover existing data sources and explore new use cases for available data that not only benefits them as an individual but possibly a whole community.

Additionally, it is crucial to include non-technically trained citizens in the process. This requires user-friendly interfaces and clear explanations to lower participation barriers and ensure all citizens benefit from digital technologies.

These factors form the basis for this dissertation's research questions and objectives, detailed in the following sections.

1.1 PROBLEM STATEMENT

Implementing a citizen-centric approach for Smart Cities presents a complex challenge encompassing technical and social dimensions. The following section details the central problem areas that must be overcome to successfully realize a distributed, citizen-operated data space.

1.1.1 *Technical Challenges*

TC1 - Interoperability: Despite the availability of open standards, interoperability between different systems and data formats remains a significant challenge. Integrating existing heterogeneous data sources is essential; these must be unified through various methodologies to enable efficient data exchange and broader data utilization [232]. This challenge requires the development of robust protocols and middleware solutions to ensure the compatibility and cooperation of various technologies.
TC2 - Discovery of Sensor Data: A central technical issue is the efficient discovery of relevant sensor data in a distributed data space. Services must quickly and accurately identify relevant data sources and request their data [212]. This requires the implementation of advanced search algorithms and an appropriate citywide data model to describe the existing data sources and sensor data.
TC3 - Data Quality: In a distributed system where uniform deployment of data sources is not guaranteed, mechanisms must be developed to assess the quality of data sources without violating the data sovereignty of the citizens [16]. Ensuring data quality is crucial to guarantee the reliability and usefulness of collected data.

1.1.2 *Social Challenges*

SC1 - Ensuring Data Sovereignty and Privacy of Citizens: A central social challenge is the retainment of citizen's control over their collected data and the protection of their privacy [110]. This requires transparent data management practices and clear guidelines that respect and promote citizens' rights. Implementing such practices must

ensure that citizens are informed about their data usage and understand the benefits of providing their data.
SC2 - Trust and Transparency: Building and maintaining trust in technological solutions are fundamental for acceptance and use by citizens [64]. This can be achieved through transparent processes, open communication, and active involvement of citizens in the development and management of the data space. Creating a trustworthy environment is essential to encourage citizens' willingness to participate.
SC3 - Inclusion of Non-Technically Trained Citizens: Another ethical issue is the inclusion of citizens who do not have a technical background. It is crucial that all citizens, regardless of their technical knowledge, can participate in the use and management of the distributed data space [110]. This requires the development of user-friendly interfaces and clear, understandable instructions to lower participation barriers and ensure that the benefits of digital technologies are accessible to everyone.

These technical and ethical issues must be addressed to ensure the successful implementation of a citizen-centric approach in Smart Cities. The following section details this dissertation's specific research questions and objectives.

1.2 RESEARCH QUESTIONS

The central research of this dissertation is guided by two overarching research questions, focusing on the sustainability of modern Smart Cities and the utilization of a distributed, citizen-operated data space by services. Specific sub-questions further detail these overarching questions to illuminate various aspects of the research project.

RQ1: How can modern Smart Cities actively integrate their citizens to become more sustainable?
This question aims to investigate the integration of citizens to improve the acceptance and sustainability of Smart Cities. The following specific sub-questions arise from this:

- **RQ1.1:** How can data from existing sensor hardware be adequately and effectively integrated into a citywide data space?
- **RQ1.2:** What mechanisms are required to ensure the data sovereignty of citizens and to create an understanding of the use of their data?
- **RQ1.3:** How can non-technically trained citizens be effectively involved in using and managing the citywide data space?

RQ2: How can services access a citywide data space and utilize the available data while considering data sovereignty?
This question examines how services can efficiently and securely access and utilize the data available in the distributed data space while

maintaining the data sovereignty of citizens. The following specific sub-questions arise from this:

- **RQ2.1:** How can services efficiently find relevant data sources in the distributed data space and utilize their data?
- **RQ2.2:** What mechanisms are necessary to evaluate data quality in a distributed system without violating the data sovereignty of citizens?

1.3 CONTRIBUTIONS

After establishing the research questions, the following section briefly summarizes the three contributions of this work. These contributions can be applied independently but are integrated into a comprehensive system through the corresponding overarching architecture presented in this thesis. Each component was prototypically implemented and evaluated either with real-world data or through simulations.

CONTRIBUTION 1: DATA STREAM DISCOVERY IN DISTRIBUTED DATA SPACES This contribution enables services to search for and request existing data streams in a distributed data space. For this purpose, an attribute-based overlay network named SkABNet was developed. SkABnet is a distributed overlay architecture based on Skipnets introduced by Harvey et al. [106]. SkABNet enhances the SkipNet identifiers by attribute-value pairs, hence the name attribute-based SkipNet (SkABNet). SkABNet provides a new search algorithm that uses these attribute-value pairs and performs semantic searches within a network of nodes without a central instance. A service can find individual and entire groups of data streams with specified semantic properties through this semantic search. The search is intelligently disseminated within the overlay network so that each node receives the message once at most, and each desired node is found in $O(\log(n))$ message hops.

In the Smart City use case, the nodes represent the individual data sources of participating citizens. Semantic properties include, for example, the type of data collected, the sensor's location description, and characteristics such as the temporal resolution of the data stream.

To evaluate SkABNet, a simulation containing up to 80,000 sensors was created. Within these networks, 14 different search queries with increasing complexity were evaluated, proving that SkABNet can reduce the total search message overhead by up to 90% compared to SkipNets search.

CONTRIBUTION 2: DATA SOVEREIGNTY RESPECTING DISTRIBUTED PREPROCESSING To adhere to citizens' privacy concerns and data sovereignty, the second contribution consists of developing a framework that allows services to define a preprocessing pipeline executed locally on the data sources. By providing various privacy settings and a visual representation of data processing, citizens are empowered to maintain their privacy and data sovereignty. Services also benefit from decentralized preprocessing of raw data, allowing them to distribute the load and ensure that all retrieved data is available in a standardized format.

The evaluation of this approach focuses on usability and applicability, particularly for non-technically trained citizens. By integrating all citizens and promoting their knowledge and understanding of data processing and analysis, the Smart City's social sustainability can be strengthened. Evaluation results show that a granular representation of processing steps can enhance citizens' understanding of the data processing procedures.

While this contribution can be applied in various use cases utilizing a distributed service architecture, it builds on the comprehensive architecture in this work, enabling citizens to engage actively with the use of their data.

CONTRIBUTION 3: CATEGORIZATION OF DATA STREAMS BASED ON THEIR DATA QUALITY When working with private sensors, one of the main challenges is evaluating the data quality. While previous work has focused on improving data quality by removing faulty measurements, this study presents an approach that categorizes data sources based on data anomalies and the events causing them. Services can subsequently use this categorization as an indicator of data quality. Still, it also allows citizens to improve their sensors' placement, thereby proactively ensuring more accurate measurement data. To achieve this, a data processing pipeline was implemented and evaluated. This pipeline compares data streams from various data sources and groups them based on common anomalies and events. While anomalies describe unexpected changes in the data stream, they are usually triggered by external events. All data sources affected by the same or similar events can be identified by categorizing the data streams.

In the thesis, this method is provided by a developed service that can assess the data quality of unsupervised data source placements by participating citizens. However, this method can also be applied in other use cases where similar data streams exist and external influences must be identified.

Chapter	Research Question	Contribution	Publications
1: Introduction			
2: Fundamentals			
3: Requirement Analysis			
4: Related Work			
5: Architecture	*RQ1.1* & *RQ1.2*		[31, 32]
6: Discovery	*RQ1.1* & *RQ2.1*	C1	[137]
7: Preprocessing	*RQ1.2* & *RQ1.3*	C2	[140]
8: Quality	*RQ1.1* & *RQ1.2* & *RQ2.2*	C3	[139]
9: Discussion and Conclusion			

Table 1: Overview of the chapters assigned to the contributions and research questions.

1.4 OUTLINE

The following section outlines the structure of the thesis. Additionally, Table 1 gives a short overview of each chapter's contribution, research questions, and published papers.

CHAPTER 2 provides an overview of contextual fundamentals such as Smart Cities and distributed data management.

CHAPTER 3 examines existing survey papers in the field of Smart City platforms and related areas to derive general requirements for an architecture focusing on a citizen-centric Smart City platform.

CHAPTER 4 reviews existing Smart City platforms and solutions for involving citizens in the Smart City context to identify open research gaps.

CHAPTER 5 introduces the overarching architecture of a citizen-centric Smart City data space. The previously identified research gaps and requirements are addressed here.

CHAPTER 6 presents the distributed discovery overlay SkABNet. It begins with the technical foundations and explains the individual components of the overlay. Finally, the discovery mechanisms are evaluated and discussed based on a simulated data space.

CHAPTER 7 attends to the distributed preprocessing of sensitive data. A user study was conducted to evaluate the feasibility and usability for non-technically trained citizens.

CHAPTER 8 introduces mechanisms for classifying data sources based on commonalities in their data streams. These classification mechanisms are evaluated and discussed using a real-world dataset.

CHAPTER 9 summarizes and discusses the research questions and contributions. In a final outlook, the remaining research topics are presented.

1.5 PUBLICATIONS

Some ideas and figures have appeared previously in the following publications:

[31] Heiko Bornholdt, David Jost, Philipp Kisters, Michel Rottleuthner, Dirk Bade, Winfried Lamersdorf, Thomas C Schmidt, and Mathias Fischer. "SANE: Smart networks for urban citizen participation." In: *26th International Conference on Telecommunications (ICT)*. 2019.

[32] Heiko Bornholdt, David Jost, Philipp Kisters, Michel Rottleuthner, Sehrish Shafeeq, Winfried Lamersdorf, Thomas Schmidt, and Mathias Fischer. "Smart Urban Data Space for Citizen Science." In: *Electronic Communications of the EASST* (2021).

[136] Philipp Kisters, Dirk Bade, and Julius Wulk. "Dynamic routing using precipitation data." In: *4th International Conference on Fog and Mobile Edge Computing, FMEC* (2019).

[137] Philipp Kisters, Heiko Bornholdt, and Janick Edinger. "SkAB-Net: A Data Structure for Efficient Discovery of Streaming Data for IoT." In: *32nd International Conference on Computer Communications and Networks (ICCCN)*. 2023.

[138] Philipp Kisters, Vinh Ngu, and Janick Edinger. "Urban Heat Island Detection Utilizing Citizen Science." In: *European Conference on Service-Oriented and Cloud Computing*. 2022.

[139] Philipp Kisters, Hanno Schreiber, and Janick Edinger. "Categorization of crowd-sensing streaming data for contextual characteristic detection." In: *Journal of Smart Cities and Society* (2023).

[140] Philipp Kisters, Leonie v. d. Veen, and Janick Edinger. "Privacy-Preserving Edge Processing in Decentralized Citizen-Centric Sensor Networks." In: *International Symposium on Intelligent and Distributed Computing*. 2023.

2 FUNDAMENTALS

This chapter provides fundamental contextual information necessary for understanding the citizen-centric approach in Smart Cities. First, the development of the concept of Smart Cities is explained, including their goals and current challenges. Then, different data management architectures are discussed, including data warehouses, data lakes, and data spaces, which are central components for managing large amounts of data in modern urban environments. Finally, the basics of sensor networks, which serve as essential data sources for Smart Cities, are covered.

Discussing these fundamentals is crucial for understanding the proposed concept of a decentralized, citizen-operated data space. These architectures and technologies form the foundation for the proposed approach and provide the necessary infrastructure and technical framework to actively involve citizens in data collection, management, and utilization.

Dedicated fundamentals sections in the following chapters provide deeper insights into the foundations required for each individual contribution.

2.1 SMART CITIES

With the rising population in urban regions, which surpassed the 55% mark in 2018 according to the United Nations [255], cities face increasing challenges. These range from heightened traffic congestion and housing shortages to escalating environmental pollution, resource scarcity, and growing vulnerability to climate change [146, 256]. To address these challenges, modern cities employ various information technologies, which, while not directly transforming them into Smart Cities [110], form a crucial foundation for tackling these multifaceted issues through informed decision-making.

The first section highlights the origins of the concept of a Smart City and how it has evolved over the years. After that, different definitions are presented and discussed. With the evolution of the term and its definitions, there also has been a shift in approach. G. Trencher categorizes this development into Smart City 1.0, which primarily focuses on the use of various technologies, and Smart City 2.0, which centers on the citizens using smart technologies to solve social problems and address individual needs [253].

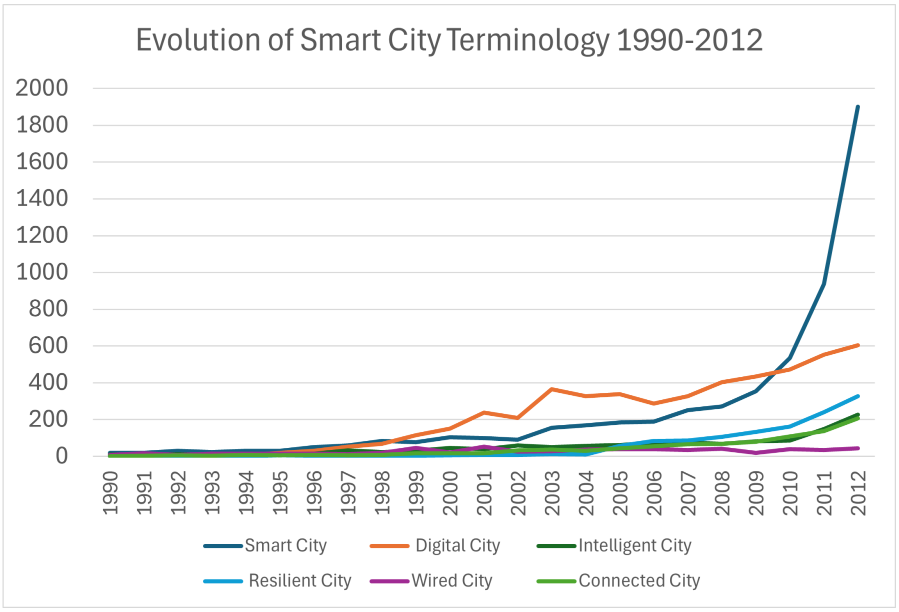


Figure 1: Development of the number of research papers published containing the given terminologies in their title or abstract.

2.1.1 *Origin*

The term *Smart* City first emerged in the 1990s [11]. Since then, many synonyms and definitions have been developed. Here, synonyms refer to alternative terms that precede a different adjective than *smart*, including *intelligent*, *wired*, or *digital*. Each adjective slightly shifts the focus, yet there is considerable overlap in the concepts understood under the umbrella of a modern city [222]. The use of these synonyms already indicates the complexity of the concepts of a modern city, as technology alone is not the sole driving force behind a Smart City. Figure 1 illustrates the change in the most used terms. The evolution of Smart City terminology reflects broader trends in urban development and technological integration. Initially, terms like *Digital City* focused on the technological infrastructure, emphasizing the digitization of urban services and processes. This period marked the early stages of urban digital transformation, where the primary objective was to establish the technological backbone for future smart applications.

Numbers are based on the results from searching Google Scholar with a given term.

As the concept matured, the term *Intelligent City* began to gain prominence in research, indicating a shift towards leveraging this digital infrastructure to make more data-driven and intelligent decisions in urban management. The focus expanded from merely having digital tools to applying these tools in a way that could enhance efficiency, sustainability, and the quality of urban life.

The transition to the term *Smart City* signifies a further evolution. It encompasses the technological and intelligent aspects and integrates them into a holistic approach that includes economic, social, and environmental dimensions. Smart Cities are now seen as ecosystems that use digital and communication technologies to improve municipal services, reduce costs and resource consumption, and engage more effectively with citizens.

In recent years, the emergence of terms like *Sustainable City* or *Resilient City* has been seen, which reflect a growing emphasis on environmental sustainability and the capacity to withstand and quickly recover from physical, social, and economic challenges. These terms highlight the need for cities to be designed and managed in a sustainable way for future generations and resilient in the face of emerging challenges.

While the term Sustainable City was previously used in the city planning and sociological context, only in the recent decade was this term also used within the technological context of Smart Cities.

The trend in Smart City terminology thus mirrors the evolving understanding of what it means to be a "smart" city in the 21st century. It's not just about the technology itself but how it is integrated and utilized to improve the overall fabric of urban life, making cities more livable, efficient, and sustainable for their inhabitants. The progression of these terms from Digital to Smart to Sustainable and Resilient Cities underscores a growing recognition of the multifaceted nature of urban challenges and the need for comprehensive, integrated solutions that address the economic, social, and environmental dimensions of urban life [58].

2.1.2 *Definitions*

As previously mentioned, there are various definitions of the Smart City concept [182], which not only highlight individual areas but also obscure different aspects behind the notion of *smartness* [110]. While individual definitions focus on diverse areas within the Smart City, most can be categorized into two main emphases. The first group of definitions primarily concentrates on the use of information technologies [100, 105, 272]. These align with the concept of Smart City 1.0, which views technology as the driving force behind Smart Cities.

The second group prioritizes the social aspect, positioning citizens as active participants in the Smart City [29, 37, 59, 75, 99]. This perspective emphasizes the importance of social interactions, community engagement, and citizen empowerment in the urban environment.

Other definitions segment Smart Cities into various components to address both the technological and social spheres [73, 132, 176]. These often adhere to the six characteristics of a Smart City identified by R. Giffinger [90]: Smart Economy, Smart People, Smart Governance, Smart Mobility, Smart Environment, and Smart Living. These characteristics shift the focus from technology to a broader view, including citizens and the real-world issues facing urban areas. This dual perspective reveals a dynamic and multifaceted understanding of Smart Cities. On one side, there is an emphasis on the foundational role of technology in enabling smart urban operations and services. On the other, there is a growing acknowledgment of the importance of social factors and citizen involvement in shaping the Smart City's evolution.

2.1.3 *Smart City 2.0*

The primary criticism of the initial approaches to Smart City initiatives revolves around their strong emphasis on technical means [163, 253]. Scholars in urban development and city planning have progressively articulated a collective critique of the neoliberal vision of the Smart City, questioning whether digitalization can truly foster sustainability, especially in terms of environmental protection and social justice [84, 91, 110, 111, 228, 260]. Consequently, the concept of Smart City 2.0 is viewed from the perspective of existing, established cities facing new challenges. These cities must consider their existing social and spatial conditions rather than completely reimagining them [228]. This transition signifies a broader recognition that while technology is essential, a Smart City's ultimate goal is to enhance its residents' quality of life. Therefore, the evolution from Smart City 1.0 to Smart City 2.0 reflects a paradigm shift from a technology-centric to a citizen-centric approach, where technological solutions are designed and implemented to meet the specific and diverse needs of the urban population. This change emphasizes the importance of not only incorporating advanced technologies but also fostering community engagement and ensuring that technological advancements contribute positively to the societal and environmental fabric of urban areas. C. Martin [163] identifies five areas of tension in the development of smart yet sustainable cities:

Economic growth is not sustainable. Economic growth is often critiqued for its sustainability, particularly from the social justice standpoint. The distribution of financial gains derived from economic growth is typically left to market forces, which can exacerbate economic inequality rather than promote social equity [200]. This perspective underscores the inherent tension between pursuing economic expansion and achieving equitable outcomes.

In the context of urban development, especially within Smart City initiatives, the emphasis on economic growth can lead to a prioritization of investments and developments that favor higher economic returns over social benefits. This approach can result in a disparity in the quality of infrastructure, services, and opportunities available to different population segments.

Uneven distribution of benefits from digital innovations. The uneven distribution of benefits from digital innovations highlights a significant challenge in Smart Cities. The concept that economic resources will trickle down to lower-income residents is, at best, a manifestation of trickle-down economics, which often fails to deliver substantial improvements to the less affluent segments of society. The advent of "smartness" in urban environments can create cities of two speeds: one that favors highly skilled worker-consumers and another that neglects the basic needs of a growing number of economically vulnerable citizens [110].

To address this, Smart City strategies must be inclusive, ensuring that the advantages of digital innovations are accessible to all city residents, regardless of their economic status. This requires a deliberate focus on equitable policy-making that aims to distribute the gains from technological advancements more evenly across the urban population. By doing so, cities can avoid the pitfall of creating a two-tier society and instead move towards a more inclusive and sustainable urban future.

Digital innovations have the potential to disempower and marginalize citizens. Public participation is often characterized as digital connections between citizens and smart urban infrastructure [17, 39]. Critics argue that through this form of participation, citizens become, either voluntarily or involuntarily, mere sensors and data sources within the Smart City framework. Instead of being empowered to engage in the Smart City actively, citizens are at risk of being instrumentalized as another efficient component of the digital infrastructure [84].

This critique highlights a fundamental concern about the nature of citizen engagement in smart urban environments. The risk is that every technology meant to enhance urban life could reduce individuals to data providers, stripping them of agency and reducing their role to passive elements within a technologically driven ecosystem. Such a dynamic can lead to alienation and disenfranchisement among residents, undermining urban communities' social fabric and cohesion.

Digitalization alone contributes little to environmental protection. There is a consensus that integrating digital solutions into urban infrastructure is crucial for the transition to a Smart City [67, 84, 110, 124, 248]. These digital solutions are intended to achieve higher operational efficiency, but the claim that they contribute to sustainability and environmental protection is criticized as a form of greenwashing [260, 277]. While increased efficiency is often justified with sustainability concerns, the broader environmental impact, such as how Smart Cities affect surrounding ecosystems that provide resources and absorb waste, is frequently overlooked and unaddressed [126, 201].

Consumer cultures are deemed unsustainable. Critics argue that these cultures, characterized by increasing material consumption, are incompatible with environmental conservation, leading to escalating environmental degradation [111, 162, 260]. This viewpoint suggests that the very ethos of consumerism, which drives much of the Smart City development, exacerbates resource depletion and environmental damage.

It becomes evident that the aforementioned tensions cannot be resolved solely through technological innovations and solutions. Instead, social structures must be considered, and solutions from social research fields need to be integrated into Smart Cities. By doing so, all city residents can benefit from digital transformation. Additionally, citizens should be encouraged to rethink their consumption habits and consider their impact on the environment. While digital solutions can assist in this regard, they cannot offset the increasing consumption driven by ever-greater efficiency. This approach highlights the necessity for a balanced strategy combining technological and social initiatives to foster sustainable urban development and a more conscientious societal ethos towards consumption and environmental stewardship.

2.2 DATA MANAGEMENT

In the era of digital transformation and the Internet of Things (IoT), characterized by an almost exponential increase in collected data and the diversity of underlying data sources, the necessity of a usually widely distributed data management system emerges as a central component of a modern Smart City [141, 180, 226, 261]. The scope and variety of data create a complex and dynamic data landscape, necessitating an innovative infrastructure capable of efficiently capturing, storing, analyzing, and distributing the data [107, 147]. This infrastructure must not only handle the sheer volume of data but also accommodate the varying nature of data types and sources, ensuring robustness, scalability, and adaptability in the face of evolving urban data requirements.

A distributed data management system is aimed to offer a solution to the challenges arising from the collection of vast amounts of data and the heterogeneity of data sources [72]. Its decentralized structure enables efficient scalability to keep pace with the continuous increase in data volume. This is crucial not only in various economic sectors but also in a Smart City environment, where many sensors, devices, and applications constantly generate data that must be processed in real-time to ensure data-driven decisions and services [107].

The decentralized architecture of such a system also contributes to increased resilience and fault tolerance. In a Smart City, which relies on comprehensive data availability, redundancy, and data distribution are essential to ensure uninterrupted services, even in the event of partial system failures or network disruptions [88].

Another significant aspect is the integration of privacy and security principles into the distributed data management system. The modern Smart City faces the challenge of managing not only a large amount of data but also sensitive information. A distributed data management system provides mechanisms for decentralized access rights management and the implementation of advanced encryption technologies to ensure data confidentiality and integrity [88].

In the convergence of Big Data and the Smart City concept, a distributed data management system thus acts as a key instrument to manage the complexity and dynamism of the contemporary data landscape. The ability to effectively organize, process, and protect data is crucial for Smart Cities, contributing to creating a sustainable, efficient, and technologically advanced urban environment [107, 141].

In implementing distributed data management systems, various approaches can be distinguished based on the nature and location of the stored data, as well as the division of responsibility for data management. Here, three fundamental principles are introduced: **Data Warehouses** serve as a central storage location that collects and unifies data from various sources; **Data Lakes** use a central storage location as well but focus on storing raw data, with the application of the data not yet clearly defined; **Data Spaces** represent a semantic abstraction layer over a set of decentralized managed data stores.

2.2.1 *Data Warehouse*

The term *Data Warehouse* is particularly known in the fields of logistics and supply chain management [15]. With the digitization within companies, there is now an increasing amount of data regarding various manufacturing processes, logistics between different locations, and sales. Traditionally, this data was maintained in separate areas, but Data Warehouses enable a unified view of relevant data [134]. In this system, data from various sources are consolidated and standardized. This aggregated dataset provides specialized data views for different parts of the organization. This can serve to optimize logistics processes, marketing, and further development and efficiency enhancement within production [15, 134].

Classical characteristics of a Data Warehouse include central data storage, centralized data management, and the preprocessing and standardization of data from various sources[86, 134]. It is also crucial that all this data belongs to a single organization, ensuring full access to the various data sources. Figure 2 illustrates the typical data flow of a Data Warehouse. On the left side, various data sources are shown, from which data is stored in the Data Warehouse using various Extraction, Transformation, and Loading (ETL) processes. By linking the different data sets, dedicated views for different areas within the organization can then be created.

While traditional Data Warehouses were based on static structures and queried new data only periodically, various approaches have emerged to utilize Data Warehouses for real-time data processing [183, 219, 223]. The demands for real-time processing and the

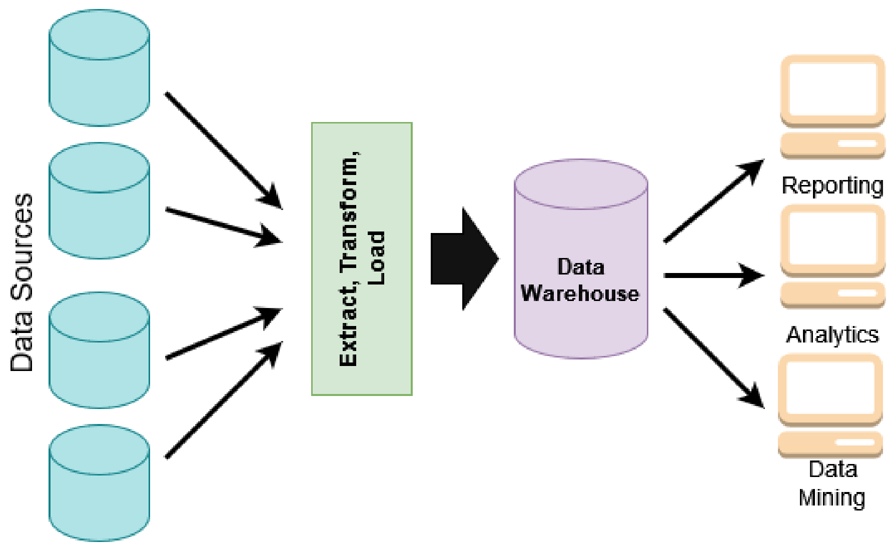


Figure 2: Data flow in an abstract Data Warehouse.

ever-changing data structures led to the development of Data Lakes, which are discussed in the following section.

2.2.2 *Data Lake*

Besides Data Warehouses, there is also the abstraction layer of Data Lakes, a concept initially introduced by James Dixon in 2010. It focuses on a central infrastructure for raw data, whose application and analysis cases are not yet fully known [71]. Since its inception, the concept has evolved and expanded, leading to various definitions. The primary focus remains on storing raw data, but contrary to Dixon's initial idea, data often originate from many sources, not just one [89]. One of the most cited works, by Nargesian et al., describes a Data Lake as a large collection of data that also possesses the following four characteristics [184]:

1. Data is managed across various storage systems.
2. Data exists in diverse formats.
3. Data may be present without associated metadata or utilize metadata in various formats.
4. Data may change over time.

These points highlight the differences from traditional Data Warehouses. The data structure is neither fixed nor standardized and can change over time. Moreover, the system in which the data is stored is not standardized but can consist of a variety of different data management systems, referred to in this context as repositories [97]. By separating the storage of data from its processing, better scalability

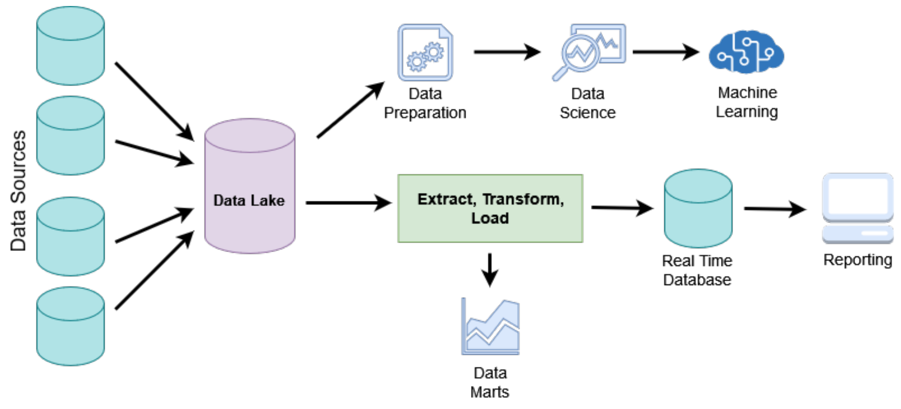


Figure 3: Abstract data lake architecture highlighting the shift to extract and transform data individually for the tasks using collected data.

is achieved, and real-time data can continuously be fed into the system [50, 97]. Data transformation and processing occur only when the data are retrieved from the Data Lake for a specific use case.

Figure 3 illustrates a Data Lake's abstract architecture and data flow. On the left side, all the data sources are shown. These can have various data formats. On the right side, data from the Data Lake are processed and utilized, and traditional methods like ETL are applied here. Machine Learning procedures on the raw data are also possible, which do require separate preprocessing [97].

While Data Lakes enable handling large volumes of data with varying structures and real-time processing, they still require a central location for storage or at least a central organization of distributed storage repositories. This means that data sources relinquish control and cannot track how their data is used. Data Lakes become impractical when data from various stakeholders are needed, and none want to provide full access to their raw data. In such instances, the concept of Data Spaces has been developed, which is explained subsequently.

2.2.3 *Data Space*

Data Spaces were first defined by Michael Franklin in 2005, introducing necessary principles[83, 98]. Since then, the term has been used in both business and research, and even the EU has promoted a European data space in its 2020 data strategy[74]. A Data Space is defined by the following four characteristics:

DISTRIBUTED DATA INFRASTRUCTURE: This means that data is not stored centrally; instead, all participants manage their data locally.

NO COMMON DATABASE SCHEMA: Data integration should occur at the semantic level, requiring a common data model that offers interoperability.

NO SINGLE SOURCE OF TRUTH: No central entity knows the absolute truth. Due to the distributed infrastructure, redundancy can occur, and consistency does not need to be enforced.

DATA SPACES CAN BE NESTED: Data Spaces can also be linked to each other and do not need to have a disjoint set of data sources. Thus, different Data Spaces can build upon one another and still share some of the same data sources.

Data Spaces, through their distributed data infrastructure and the autonomy of data management without the need for a common database schema, offer significant advantages over Data Lakes and Data Warehouses for the Smart City. A citywide data space enables linking various data sources from industry, private individuals, research institutions, and official bodies without all participants needing to agree on a universal database schema. It is crucial that the existing data are semantically described. This way, even without a common data schema, semantically related data can be found and linked at the semantic level. Another advantage is that the respective owners retain sovereignty over their data and can continue to decide for themselves with whom they share their data or for which external services their data can be used. To facilitate this, participants communicate via a so-called *Connector*. This Connector should, in turn, be technology-independent or available as a microservice to be integrated into existing infrastructure. The Connector manages all data space-relevant information and exists independently of the previous infrastructure [116].

2.3 SENSOR NETWORKS

Data is one of the most crucial assets today [259]. When analyzed and interpreted correctly, it offers various organizations operational advantages [203] and fosters innovation [289], enabling them to proactively identify and address new challenges. Only with enough data can relevant information be extracted and used to support decision-making processes. Increasingly, reliance is placed on distributed sensor networks that monitor their environments and make this data available as a continuous stream. The decreasing cost of sensor hardware also plays a crucial role, enabling the deployment of extensive distributed sensor networks in an expanding range of areas in the daily live [8]. Sensor networks are wireless networks comprised of numerous small, autonomous sensors capable of collecting data from their surroundings and transmitting it wirelessly to a central data processing unit. These networks often gather, process, and transmit information in real-time, making them invaluable in various applications within the Smart City and in other sectors like agriculture, disaster warning systems, and detection of abnormal behaviors. Their

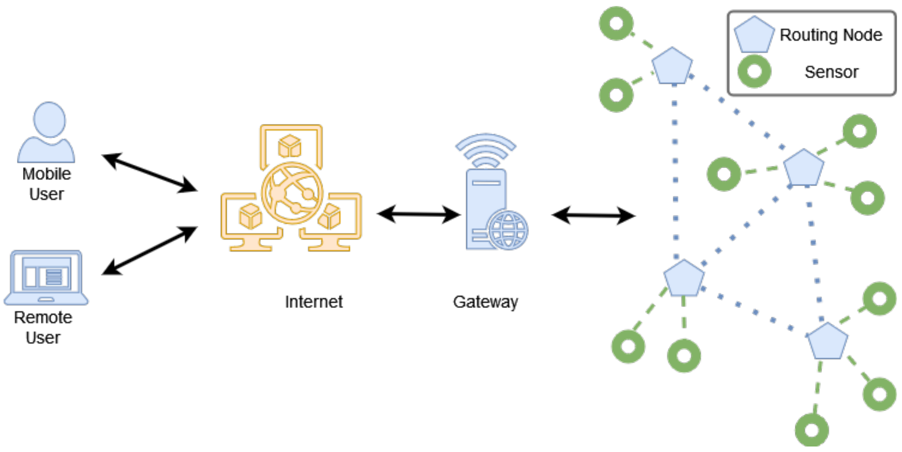


Figure 4: Abstract architecture and communication model for a centralized sensor network. On the right, a distributed network of sensors connects to the internet via gateway nodes and sends their data to the end-user or application.

real-time data collection and processing capabilities provide the foundation for responsive and adaptive systems that can significantly enhance efficiency, safety, and quality of life in various settings.

2.3.1 *Common Properties*

While sensor networks are versatile and often deployed for specific purposes, they generally share the following four characteristics [8, 44, 61, 254]:

Wireless communication is a key feature of sensor networks, allowing sensors to communicate with each other and with a base station or gateway wirelessly. This enables flexible and cost-efficient deployment. Various wireless communication solutions can be employed in this context. An abstract sensor network consists of sensors and routing nodes, as shown in Figure 4. Sensors collect data and send it to the gateway via routing nodes, which forward it to the application or end-user.

An alternative architecture is depicted in Figure 5, where individual sensor nodes form an ad-hoc network. This setup requires balancing the added complexity of dynamically routing sensor data to the gateway or so-called sink node against the benefits of increased energy efficiency and flexibility [51, 149, 173, 278]. Additionally, ad-hoc networks are applicable in areas without existing wireless infrastructure [169, 170, 186], offering the ability to establish connectivity in remote or undeveloped regions, thus enhancing the reach and utility of sensor networks.

Distributed sensors in a network are installed across a wide area, allowing for data collection at varying densities depending on the specific requirements. The deployment of sensors in large numbers

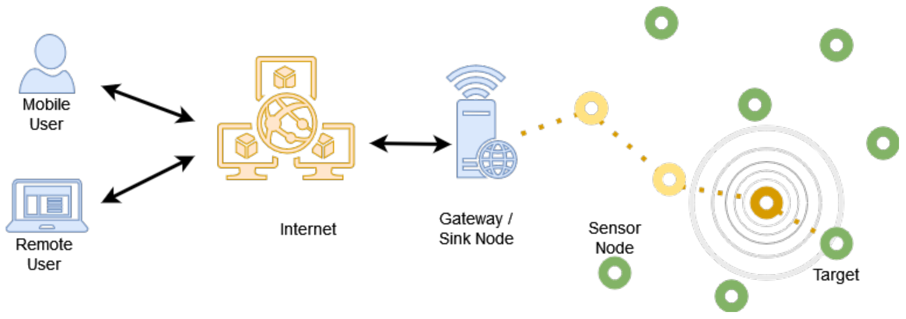


Figure 5: Abstract architecture and communication model for an ad-hoc sensor network. Within this architecture, sensors route their collected data via other sensors to the closest sink node with an internet connection. From here, it is made available to the end-user or application.

enables the gathering of comprehensive information about the entire area. This extensive distribution ensures that detailed and varied data are obtained, providing insights into different aspects of the environment or monitored phenomena.

Autonomous sensors are often battery-powered and must therefore operation must be focused on energy efficiency. They are typically designed to function autonomously and require minimal maintenance. However, it is expected that individual sensors may fail or transmit erroneous data, which must be considered particularly in processing collected data.

Sensor networks deliver **real-time data**, making them particularly useful for applications requiring rapid information or responding to real-time changes. However, the continuous stream of data results in a large volume of transient measurements, which individually may contain little information. Collected data gains significance when contextualized and related to other measurements, as the aggregation and analysis of these data points over time can reveal trends, patterns, and anomalies that inform more comprehensive insights and decision-making processes.

2.3.2 *Application Areas*

Sensor networks play a crucial role in various aspects of modern life, with environmental monitoring being a domain where vast and extensive sensor networks are deployed. In this field, widely distributed sensors provide comprehensive, accurate, and timely data essential for understanding and managing natural resources and ecological processes. In the following, typical applications of sensor networks in environmental monitoring are presented:

CLIMATE AND WEATHER MONITORING: Extensively distributed sensor networks collect data on temperature, humidity, wind speed and direction, precipitation, and other meteorological parameters. This data is pivotal for weather forecasting and analyzing the impacts of climate change [164].

AIR QUALITY MONITORING: Sensors measure the concentration of pollutants such as CO2, NOx, SO2, and particulate matter in the air. This data is crucial for assessing air quality and formulating air pollution control strategies [161, 204].

WATER QUALITY MONITORING: Sensor networks in rivers, lakes, and oceans measure parameters such as pH level, dissolved oxygen, salinity, temperature, and the presence of contaminants. This data is vital for monitoring water pollution and managing water resources [4, 208].

SOIL MONITORING: Sensors embedded in the soil collect moisture, nutrient content, and temperature data. These insights are crucial for agriculture, particularly precision farming, and for assessing soil quality and health [158, 171].

EXPLORATION AND PROTECTION OF ECOSYSTEMS: By monitoring various environmental parameters, sensor networks contribute to understanding and protecting ecosystems. They enable researchers to observe and analyze changes in ecosystems [218, 243].

ENERGY EFFICIENCY AND SUSTAINABILITY: Sensor networks also facilitate the monitoring and managing of energy consumption, which is crucial for developing sustainable practices and reducing the ecological footprint. This is particularly used in individual smart buildings or entire sectors of a Smart City [177].

By providing real-time data, sensor networks in environmental monitoring enable proactive and data-driven decision-making, which is crucial for addressing environmental issues and promoting a sustainable future. Sustainability and environmental monitoring are also gaining significance in Smart Cities. Given their substantial impact on the environment [27, 185], various measures are being implemented to minimize these effects. Urban sensor networks [60, 152] assist in validating the impacts of the measures taken or identifying areas of the city where further actions are needed, thereby enhancing the ability to manage urban environments sustainably and responsibly.

2.3.3 *Citizen-Operated Sensor Networks*

In addition to officially installed sensor networks, distributed sensor networks organized by the scientific community or motivated and active citizen networks are increasingly emerging [122, 144]. Furthermore, private households are integrating more sensor technology [70],

https://sensor.community/

serving as tools in Smart Homes or being installed out of interest. Collected data in these cases is often only used locally, which adds no value to the broader community. Official citywide sensor networks must be established, maintained, and frequently defended by the city against public concerns regarding privacy and data protection. Additional challenges include data transmission, storage, and processing [5]. Given the high costs associated with Smart Cities, many cities are trying to integrate active citizens into the data collection process, utilizing existing hardware and knowledge. This movement towards citizen involvement is referred to as *Crowd Sensing* or, in the context of Smart Cities, as *Citizen Science* [76].

2.4 SUMMARY

This chapter introduced the fundamental concepts and technologies, outlining the origins of the term Smart City and its evolving definitions. The focus is particularly on the shift from a technology-driven to a citizen-centric Smart City, often referred to as Smart City 2.0. Even with this shift towards a citizen-centric approach, a significant challenge remains in managing the vast amounts of data collected from various areas of a Smart City. The chapter presents three different conceptual data management mechanisms to address this issue. In Smart Cities, widely deployed sensor networks typically serve as data providers, enabling decision-makers to identify and address challenges proactively.

The next chapter addresses Citizen Science platforms and their goals, analyzing established requirements for a citywide data platform and discussing them with a focus on a citizen-centric approach.

3 REQUIREMENT ANALYSIS

Following the establishment of key contextual foundations in the previous chapter, this chapter delves deeper into the motivation and challenges behind the current state of citizen science. It analyzes existing surveys to understand the established requirements for Smart City platforms. First, the two main citizen science platforms and their goals are presented. Subsequently, an analysis of established requirements based on a literature review of existing Smart City platforms and their enabling technologies is conducted. These requirements are then discussed with the previously presented goals, focusing on a citizen-centric Smart City platform. This analysis helps to identify requirements and challenges that must be considered in implementing a citizen-centric approach.

3.1 CITIZEN SCIENCE

Citizen Science describes the integration of citizens into scientific studies and research, where tasks that require only minimal training and no deep expertise are distributed among large groups. These volunteers then undertake a significant portion of the often manual tasks voluntarily. Through the sheer number of participants, both costs and time expenditure can be reduced, and citizens become actively involved in local and current research [76].

Beyond manually analyzing or evaluating data, the task of data collection itself is increasingly being distributed to volunteers. This approach allows for the gathering of a much larger data set. A classic example is the *Annual Bird Count* event, conducted annually by various organizations worldwide. In this event, volunteers globally assist in observing and counting birds to better estimate global bird populations.

In February 2024, global participants reported approximately 7920 bird species in more than 210 countries and sub-regions.

3.1.1 *Existing Citizen Science Platforms*

There are several platforms where scientists can initiate projects with a Citizen Science component, inviting interested citizens to participate. Projects on these platforms focus on distributing manual tasks from a small research team to a broad base of volunteer helpers. Participating and contributing to these projects is not paid or monetary motivated; rather, it seeks individuals who are personally passionate about these topics and willing to help voluntarily. These volunteers contribute either to gain new experiences for themselves or to support research efforts actively. Following, the two largest publicly sponsored platforms are presented in detail.

EU-Citizen.Science

https://eu-citizen.science/

EU-Citizen.Science (ECS) is the largest Citizen Science platform in Europe. This platform is a central hub for all interested in Citizen Science within the EU region. It offers a wide array of Citizen Science projects with which citizens can collaborate or participate. Additionally, it provides best practices for initiating, planning, and conducting Citizen Science projects. The platform was supported by the *Science with and for Society* (SwafS) program and has been continued and funded by the European Citizen Science Association since August 2022. ECS has set itself the following goals:

1. Expanding the Reach of Citizen Science by strengthening the community among participating citizens and project partners.
2. Enhancing Digital Competencies for FAIR (Findable, Accessible, Interoperable, Reusable) and open science communities.
3. Further Development of the ECS Platform through co-design, involving the platform's users in the design and development process.
4. Development of the ECS Academy. Offering free training to increase capacity for conducting Citizen Science projects.
5. Promoting Inclusion and Diversity to bring Citizen Science to the forefront.
6. Advocating for Citizen Science and working on the policy impacts of this form of science.
7. Researching the Impacts of Citizen Science on research, society, and the economy.

SciStarter

https://scistarter.org/

One of the largest platforms globally is *SciStarter*, an organization based in the USA, developed in 2011 by Darlene Cavalier at the University of Pennsylvania. Since 2014, Arizona State University and SciStarter have collaborated to create a series of ongoing research and development initiatives. Initially, their efforts were primarily funded by the National Science Foundation, but later, additional organizations like the Institute for Museum and Library Services also supported them. SciStarter now provides access to over 3,000 formal and informal research projects, events, and tools, continually striving to engage more people in science. It serves as a directory for projects and offers a coordinated way to track contributions and make the tools and instruments needed for participation in citizen science projects accessible.

Over 100,000 global citizens are part of the SciStarter community. These members can search ongoing projects by location, topic, age level, etc., which have been registered by individual project leaders or imported through partnerships with federal governments, NGOs, and universities.

SciStarter's official goals are:

1. **Empower and Encourage**: Enable people to learn about, participate in, and contribute to real science through both informal leisure activities and formal research efforts.

2. **Foster Appreciation and Understanding**: Promote greater appreciation and understanding of science and technology.

3. **Create a Collaborative Space**: Establish a shared space where scientists and project leaders can collaborate with people who want to work on their research projects or learn more about them.

4. **Make Building and Exploring Accessible and Fun**: Simplify and enhance the process of crafting, building, and exploring for people from all walks of life, helping them see the impact of their curiosity, interests, and concerns.

3.1.2 *Goals and Challenges*

The goals and challenges of Citizen Science are reflected in the platforms previously discussed, ECS and SciStarter. The common objectives across these platforms can be summarized as follows:

Collection of Larger Data Sets: A foundational goal of Citizen Science is the collection of larger and more representative data sets. The platforms discussed facilitate this by enabling interested citizens to explore ongoing projects and learn more about participation opportunities, thereby enhancing the breadth and depth of scientific data collection. While most of the launched projects rely on manual data collection, due to the increasing amount of privately owned sensors, automated and continuous data collection comes more and more into focus.

Promotion of Civic Engagement: Citizen Science aims to actively involve citizens in scientific endeavors, making science more accessible. The platforms provide access to a wide range of projects, which helps to facilitate participation and public engagement, fostering a deeper connection between science and society.

Networking of Engaged Citizens: Through collaborative efforts and community building, Citizen Science connects participating citizens. Project creators—whether they are public organizations or private companies—also engage more closely with participants, fostering a sense of community and shared purpose.

Inclusion and Diversity: Active participation from diverse individuals with varying interests and concerns promotes inclusion and diversity within the scientific community. Citizen Science platforms strive to make projects accessible to a broad and diverse participant base, thus enriching the scientific process with a range of perspectives.

Education and Training: Active participation in Citizen Science projects facilitates ongoing education and skill development for citizens. Platforms support this with dedicated courses for project leaders as well, aiming to improve the quality and effectiveness of Citizen Science projects. This educational aspect not only enhances participants' knowledge and skills but also increases the overall impact and credibility of Citizen Science initiatives.

While motivating citizens can quickly lead to larger datasets, scientists face new challenges arising from the broad collection of data by mostly untrained participants. The main challenges in Citizen Science can be summarized as follows:

- **Data Quality and Scientific Rigor**: With the increasing amount of unsupervised collected data, ensuring data quality and scientific rigor is one of the greatest challenges in Citizen Science. Platforms address this by offering educational resources and providing guidelines for both project leaders and participants to help maintain standards.

- **Representativeness and Bias**: Ensuring data representativeness is crucial. Citizen Science projects often attract participation from specific demographic groups, which can introduce bias into the results. It's important to implement strategies to encourage broader and more diverse participation to mitigate this issue.

- **Data Privacy and Protection**: Protecting personal data and maintaining the privacy of participants are critical, especially in projects that collect personal or sensitive data or record data for quality assurance purposes. Robust privacy policies and secure data handling practices are essential to safeguard participant information.

- **Engagement and Motivation of Participants**: Maintaining volunteers' engagement and motivation over time can be challenging. Projects need to be designed to be interesting and accessible to encourage ongoing participation, especially for those that require active and continuous input from participants.

- **Integration into Formal Science**: Integrating Citizen Science effectively into established scientific processes and disseminating results within the broader scientific community is another challenge. This involves not only ensuring that data and findings from Citizen Science projects are credible but also fostering recognition and acceptance within the scientific community.

Addressing these challenges requires careful planning, community engagement, and continuous improvement of practices and tools used in Citizen Science. By doing so, the field can significantly contribute to scientific research and public education while enhancing the societal relevance of science.

3.1.3 *Smart Cities and Citizen Science*

With the shift from a technology-driven to a citizen-centric Smart City concept, the inclusion of citizens not only as users but also in data collection and knowledge generation processes is gaining increasing interest [96, 111, 150, 233, 234, 245]. As mentioned before, the main goal for a Smart City is to increase the quality of life for its citizens [37, 54, 105, 175]. Smart cities have frequently failed to meet their objectives because citizens were not properly involved in the planning process, and the impact on their daily lives was not adequately considered [58]. By empowering citizens to take active roles within a Smart City, and therefore democratizing innovation should be part of the Smart City concept to fulfill its goals [197, 225]. To achieve this goal, Smart Cities must no longer view citizens merely as data sources. Instead, citizens should be regarded as equal participants who can take initiative and actively contribute to the development and improvement of the city.

3.2 SMART CITY REQUIREMENTS

After examining Citizen Science's goals and challenges, this section analyzes existing requirements analyses for Smart Cities and investigate how these requirements change with active citizen participation. Several survey papers are examined to gain an overview of the developed requirements for a Smart City platform and its enabling technologies. The surveys were selected primarily based on their recency; all of them have been published within the last ten years. An exception is a survey by da Silva et al., published in 2013; nonetheless, it was added due to its sensor-focused approach [232]. Furthermore, the surveys do not focus solely on individual aspects of Smart Cities but provide a more comprehensive view of the Smart City concept as a whole. For an easier overview, these surveys are grouped by the three major technologies driving the development of Smart Cities: Big Data, IoT, and Cloud Computing [23, 104, 222, 279]. Within the investigated surveys, defined requirements are extracted and compared. These requirements are then analyzed with a focus on Citizen Science, discussing how they might change through the active involvement of engaged citizens.

Santana et al. [222], Javed et al.[118], Goumopoulos [94], and Mohanty et al. [175] agree that with the increased data collection due to the amount of IoT devices within a Smart City, Smart City platforms need to manage large amounts of data. Here, the four main characteristics are Volume, Variety, Velocity, and Veracity [45, 68]:

- **Volume**: The rapid proliferation of data generation and collection necessitates the development of robust tools capable of addressing this challenge. In the context of Smart Cities, the volume of data is substantial, originating from a large set of distributed sources. With the increased volume of data, the placement of data sources and existing redundant data collection by different providers amplify the need to manage, filter, and evaluate these data sources effectively.

- **Variety**: Within a Smart City, collected data comes from a diverse array of sources and exists in structured, semi-structured, or unstructured formats, such as images and video recording, environmental measurements, and raw text, respectively. This complexity is particularly relevant in the context of Smart Cities, where data is sourced from a large variety of sensors and personal devices owned by citizens.

- **Velocity**: Even with a large amount of collected data, processing must be rapid and, in some instances, real-time to maintain its utility. Operators must respond quickly to urban issues like water leaks, accidents, and fires.

- **Veracity**: Data quality is crucial due to the large volume of data collected. Even when utilizing a heterogeneous set of data sources, faulty data or the use of unreliable sources can significantly compromise data analysis. In a Smart City environment, poor data sources can include incorrect readings from falsely deployed or malfunctioning sensors or even collected data from malicious users.

As a result of these challenges and the need to deploy and manage data sources that collect required data, Table 2 gives an overview of the requirements for Smart City platforms and applications discussed by each investigated survey paper. It shows that for functional requirements *Data Management* and *Data Processing*, are the most discussed and found in literature, given by the data driven nature of Big Data, IoT, and Fog Computing various solutions for challenges within this domain are proposed. The next most discussed requirements address data sources, particularly focusing on *Sensor Management* and *Data Discovery*. These aspects are often referred also to as resource management within the IoT domain. As the discovery and the

	Group	Smart City Platforms								Big Data			IoT			Cloud & Fog			
	Paper	[222]	[125]	[232]	[104]	[88]	[118]	[64]	[244]	[10]	[63]	[190]	[194]	[143]	[20]	[246]	[252]	[196]	[62]
Functional Requirements	Data Management	●		●				●	●	●	●	●	●	●	●		●	●	●
	Application Runtime	●																	●
	Sensor Management	●		●		●		●			●		●		●				●
	Data Processing	●		●				●	●	●	●	●				●	●	●	
	External Data Access	●						●	●			●	●				●		
	Service Management	●		●	●			●	●				●						●
	Software Engineering Tools	●													●				
	City Model Definition	●		●				●	●					●				●	
	Mobile Sensors Support			●					●									●	●
	Citizen Integration			●	●			●	●	●		●	●	●		●			
	Network Infrastructure				●	●	●			●	●		●	●				●	
	Government Integration							●		●			●						
	Data Discovery					●			●		●	●	●		●	●		●	
None-Functional Requirements	Interoperability	●	●	●		●	●	●	●		●	●	●		●	●	●		●
	Scalability	●				●	●	●					●		●				●
	Security	●	●		●	●	●	●	●	●	●	●		●	●	●	●	●	●
	Privacy	●	●	●	●	●		●	●	●	●	●		●	●	●	●	●	
	Context Aware	●						●									●	●	●
	Data Quality		●		●		●		●	●	●	●		●			●	●	
	Adaption	●	●	●				●					●	●					
	Extensibility	●	●	●			●	●		●	●								●
	Configurability	●				●												●	
	Sustainability			●	●		●		●									●	
	Real Time-able			●				●		●		●	●	●		●		●	●
	Availability		●	●		●		●							●	●		●	●
	Performance		●			●		●								●		●	
	Usability		●					●						●	●		●		
	Robustness						●									●		●	

Table 2: Comparison of survey regarding their investigated requirements.

proliferation of distributed data sources increase, network infrastructure becomes a critical focus area. Additionally, it is evident that the evolving definition of Smart Cities increasingly *Incorporates Citizen* engagement as a key requirement. Interestingly, the involvement of city officials (*Incorporating Government*) is less frequently mentioned as a requirement. This is likely because their participation is generally assumed, given that official bodies typically drive various projects and the local Smart City concept.

In the domain of Smart City platforms, *Service Management*, the *City Model Definition*, and *External Data Access* are frequently discussed requirements. Services often refer to end-user applications that utilize existing data within the Smart City. These services rely on a diverse set of often heterogeneous data sources, necessitating a defined semantic model to facilitate the interpretation and use of data from various sources. Additionally, some services are not directly deployed on the Smart City platform, thus many systems depend on external data access, allowing third parties to access the collected data within the city. Finally, the least frequently mentioned requirements are *Support for Mobile Sensors*, *Application Runtime*, and *Software Engineering Tools*.

Overall, more surveys have addressed and examined non-functional requirements, with a significant focus on *Security* and *Privacy*. While security is important in all areas within a Smart City, privacy, in the context of Smart Cities, is particularly concerned with the data collected from or about citizens. Behavioral and movement data are considered especially sensitive because they can easily lead to inferences about individuals and have a high potential for misuse in surveillance or manipulation.

Interoperability and *Data Quality* closely relate to data processing and utilization. Interoperability refers to using heterogeneous data sources with diverse protocols and communication channels that need to be interconnected. Given the vast number of distributed sensors, it is crucial to assess the quality of the collected data to identify outliers from misconfigured or faulty sensors.

Next, the requirements for the underlying system and infrastructure are considered. In many areas, city-wide or critical services expect *Real-Time Data* to respond promptly to new circumstances. It is important to note that this requirement often stems from the cloud, fog, and IoT domains and is less commonly addressed at the more abstract level of Smart City platforms. *Availability* also plays a crucial role, as these services must always receive required data, which is particularly challenging in the IoT sector. Additionally, *Scalability* and *Extensibility* are critical requirements for the ever-growing number of sensors within Smart Cities. Given the vast number of data sources and the volume of collected data, city-wide networks must scale efficiently and be prepared to handle increasing data loads. While *Perfor-*

mance is less frequently mentioned as a standalone requirement, it is often associated with scalability.

Long-term planning is challenging due to changing requirements, new technologies, and evolving use cases. Therefore, a city-wide system should be *Extensible*, *Context-Aware*, *Sustainable*, *Usable*, and *Adaptive*. The least mentioned requirements in the analysis were *Configurability*, and *Robustness*. Here, configurability is often correlated with extensibility and adaptability but focuses more specifically on the individual data sources and services. The robustness of the network is often closely related to its security, scalability, and performance.

3.3 REQUIREMENT EVALUATION IN THE CONTEXT OF SMART CITIES

The shift from a technology-driven to a citizen-centric Smart City approach often alters individual Smart City requirements interpretation. With the active participation of largely untrained citizens in various processes, existing solutions for established requirements must be rethought and adapted to new circumstances. This does not mean that previous requirements lose their validity; rather, an additional perspective focusing on the needs of participating citizens is added. This section evaluates and discusses the most important requirements, focusing on Citizen Science.

Given the abundance of sensors already present in a citizen's life, a participatory citywide sensor network should utilize these existing devices rather than requiring new, homogeneous sensor hardware. This approach increases the need for *interoperability* and *data quality* but avoids additional acquisition costs, thereby lowering the barrier to participation in such a network. It also focuses on *sustainable* solutions and decreasing consumerism.

Another advantage is that by reusing existing hardware, the need to monitor and repair a multitude of redundant sensors is eliminated, reducing maintenance overhead and costs. To facilitate the integration of existing sensors, the system should allow data to be fed in various ways, highlighting the need for an *extensible* platform. It should support multiple existing protocols to ensure compatibility and provide an accessible means to create custom data integration.

With the growing support of existing sensors, the network needs to be *scalable* and *adaptable* regarding the changing landscape within the citywide sensor network. While professional networks can expect some degree of uptime for their managed sensors, in a private network, sensors might join and leave the network continuously, resulting in data discovery and sensor management challenges.

To simplify *sensor management*, participants should be provided with a clear interface to view information, monitor status, and adjust settings for each sensor, as the sensors allow.

In addition to managing sensors, *data management* is equally crucial. When sensitive or private data is collected and utilized, it is essential to provide citizens with an overview of their collected data and its usage, thereby enhancing *privacy* and *security* transparently. Citizens should be able to categorize the sensitivity of their collected data, enabling them to actively decide which data to share and which to keep private. This process should be supported with examples and best practices to aid citizens in making informed decisions.

This classification ensures that citizens are aware of their data's sensitivity and understand which data can be accessed by external applications or services. By fostering such awareness, citizens can make informed data privacy and security choices.

In addition to data management, it is crucial to motivate citizens to utilize and process existing data from the network. Citizens should be encouraged to not only act as data sources but also to actively participate by defining services that visualize or analyze data from the network, thereby creating value for themselves and the community.

Given the vast scope of a citywide sensor network, a single citizen cannot have an overview of all existing data, increasing the need for *adaptable* and *context-aware data discovery* tools. Enabling citizens to create services themselves motivates them to engage with the existing data and foster a sense of community. This engagement makes conveying the collected data's benefits easier, enhancing community involvement and collaboration.

To facilitate the development of services by citizens and communities, various tools should be available to simplify common challenges. These tools could include *development tools* for creating custom services, connecting existing sensors, or even initiating citizen science projects requiring data from other participants. Since these tools can vary based on different use cases, the community should be able to contribute their own custom tools. This approach further strengthens the community in the long term and promote the development of a diverse and resilient community.

Once individual services have been created or collected, data is shared with external services, providing an interface for managing these services and data provision is crucial. Firstly, this interface should enable service developers to manage connected data sources, including discovering new data sources as needed or adding external public data sources. Secondly, service maintainers should continuously assess the data quality and, if necessary, manually remove specific data sources to ensure better data quality. Thirdly, service developers need the capability to improve their services and implement updates, which may require collecting new data or changing the data processing methods of existing sources. These sources must be informed about any such updates.

For citizens who collect data, the service management interface should provide an overview of the services using their provided data, along with information about data processing and usage. Services that request access to collected sensitive data should be listed with all relevant details, enabling citizens to make informed decisions about their participation. Lastly, data providers should be able to stop data provision for individual services they no longer wish to support.

Using various heterogeneous sensors with different specifications and characteristics necessitates defining a comprehensive citywide data model that enables data sources and services to describe and understand collected data. Such a data model allows data sources to be represented on a semantic level, simplifying the interpretation of data regardless of individual specifications and enabling the network to identify suitable data sources based on semantic descriptions.

This data model must be widely adopted throughout the network to empower citizens to describe their data. Additionally, provisions should be made to expand the data model, continuously supporting new data types and accommodating a growing data space.

3.4 SUMMARY

The transition from a technology-driven to a citizen-centric approach for Smart Cities requires a reinterpretation of existing requirements and the adaptation of existing solutions to the active participation of unskilled citizens. Utilizing existing sensor hardware reduces the costs and barriers to participation, promoting more sustainable solutions and less consumption. The participation of many citizens as equal actors increases the heterogeneity and size of the network, posing new challenges for interoperability and data quality. A scalable and adaptable network that supports various protocols is necessary to ensure sensors' continuous integration and management. Transparent data management practices and clear user interfaces for sensor management are crucial to maintaining citizens' data sovereignty and privacy. Citizens should be able to categorize the sensitivity of their data to make informed decisions about its use and strengthen their data sovereignty. This promotes citizen engagement in developing their own services and actively contributing to data utilization. Tools to support service development and data integration are also important to facilitate participation and strengthen the community. A comprehensive data model that semantically describes heterogeneous sensor data enables unified data interpretation and eases data discovery and utilization. Such models must be continuously expanded to support new data types and accommodate the growing data space. These requirements align with the goals of Citizen Science platforms, such as ensuring data quality and scientific rigor. Broad participation increases the representativeness of the data, reducing biases and strengthening scientific validity. Protecting personal data and maintaining participants' privacy are essential to gaining and maintaining citizens' trust. Engaging and accessible projects can foster motivation and long-term commitment from participants. Finally, integrating the results of Citizen Science projects into formal science is crucial to ensure the credibility and recognition of the collected data and insights.

In the following chapter, existing Smart City platforms are examined, and their solutions for the presented requirements are analyzed. In the analysis, the examined platforms are divided into four groups. The largest group deals with general platforms not developed for a specific use case. This is followed by three frequently studied topics: *Smart Infrastructure, Smart Traffic, and Smart Environment*. Based on these works, open research gaps are identified, and their relevance is highlighted.

4 RELATED WORK

In the previous chapter, the goals and challenges of Citizen Science in relation to Smart Cities were examined in detail, and a comprehensive requirements analysis was conducted. It became evident that modern cities must use modern technologies and promote active citizen participation to achieve their goals of improved quality of life and sustainability. With growing awareness of data security and privacy, Smart Cities must develop new methods for citizens to autonomously collect, manage, and provide their data for various citywide services.

The increasing proliferation of sensor hardware in everyday life presents an opportunity to reduce acquisition and initialization costs for new projects by reusing existing devices. This distributed responsibility also helps to lower maintenance costs. However, this approach introduces new challenges regarding sensor placement and the resulting data quality.

Following the requirements analysis, this chapter investigates related work and everyday use cases for Smart Cities. To achieve this, papers are grouped into general architectures applicable to various use cases and specific use-case groups. After a brief overview of the related papers and their use case, they are compared based on their architecture, data storage and processing methods, and the integration of citizens in their proposed systems and solutions. The chapter concludes with a discussion of how different use cases can benefit from citizens' active participation, both as additional data sources and by allowing them to create custom services.

4.1 EXISTING SMART CITY PLATFORMS AND THEIR USE CASES

When researching existing Smart City platforms, there are two main differences within the state of the art. First, citywide architecture proposals claim to be use-case independent and work with different kinds of data, applications, and services. Second, there are use case-specific platforms. These target one of the challenging areas within modern cities and present an architecture that is especially well suited for that given use case but might not apply to other use cases. Nonetheless, important lessons can be learned from these specific approaches that can help to create a citywide platform. Therefore, these are also briefly described and included in the comparison and discussion.

4.1.1 *General Smart City Platforms*

In the domain of citywide sensor networks, various solution approaches have been implemented to differing extents. One of the largest and most comprehensive Smart City projects is SmartSantander, where over 15,000 sensors were deployed to create a citywide sensor network accessible for various applications [47]. Due to its scale and the diverse range of initiatives in Santander, this project is presented in greater detail, highlighting its relevant works. This is followed by examining several other platforms that focus on creating a reusable software platform rather than on a single city. The underlying architecture of each platform is briefly explained, and the integration of citizens is discussed.

4.1.1.1 *Smart Santander*

In the Smart City sensor networks realm, the SmartSantander project [221] stands out as one of the most renowned and frequently cited research initiatives. Launched in 2010 with funding from the EU's "Future Internet" program, SmartSantander has since taken a leading role in developing and testing Smart City technologies. As one of the first large-scale urban test environments for the IoT, SmartSantander has significantly contributed to the research and advancement of Smart City concepts. Implementing a dense network of over 15,000 sensors throughout the Spanish city of Santander collecting and analyzing real-time data on environmental conditions, traffic flow, energy consumption, and other urban parameters. This extensive data repository facilitated the optimization of urban services and promoted citizen engagement. By making all this data available, citizens are encouraged to give feedback on the project and actively support innovation processes.

The architecture of the SmartSantander project is divided into three layers: the IoT Node Tier, the IoT Gateway Tier, and the Server Tier. The Node Tier consists of the actual hardware sensor nodes, which are distributed throughout the city. These nodes are often autonomously installed and thus have limited resources (energy, storage, computing power). Additionally, they are exposed to various environmental factors. The Gateway Tier connects these individual sensor nodes to the underlying project infrastructure. These gateways have a permanent internet connection and are programmable in operation, as they are intended to be used for investigating various solution approaches. Finally, the computationally powerful Server Tier can run various applications and collect nearly unlimited data. No specific protocols are mandated for communication between the layers, meaning different sensors can be connected to the gateway in various ways.

Overall, the SmartSantander project has already considered many previously established requirements. However, the most significant difference lies in active citizen participation and using existing sensor infrastructure. The SmartSantander project focuses on technical solutions and deploying new dedicated hardware, which the city and participating partners own. This approach is strongly aligned with the concept of Smart City 1.0. While the initial paper mentions citizen participation and suggests that citizens could collect data using their smartphones [221], it does not address the storage and processing of this collected data. Subsequent papers refer to a central Big Data repository where all sensor data is stored [47], implying a centralized storage system for all collected data, including that from citizens.

In the paper by E. Theodoridis et al. [250], two studies are presented where citizens can participate in data collection using their smartphones. In this use case, the data is also centrally stored. The requirements also touch on security and privacy. The authors acknowledge that "behavior tracking with smartphones raises personal and public privacy/security concerns" [250, p. 3] but do not delve further into solutions for these challenges. V. Gutiérrez et al. present another application that relies on participatory sensing and depends on citizen cooperation [96]. In this case, although citizens are encouraged to participate, they are merely used as data sources. They can report damages or accidents, but they have no influence over the storage of their collected data.

Therefore, SmartSantander provides many interesting insights into technical solutions but needs further improvements to be an inclusive and socially sustainable city.

4.1.1.2 *City Independent Platforms*

The following section investigates city independent platforms and discusses their view on citizen participation and, if applicable, Citizen Science.

The Smart City platform by Chamoso et al. focuses on Smart Home owners and aims to reuse existing Smart City applications and their data [41]. To achieve this, a Smart Home platform was developed to present locally collected data to the user in a web-based user interface, enabling citizens to participate in external services. Citizens can choose which services they want to participate in. Nonetheless, citizens must give services full access to their collected data, which might lead to private information being leaked within data patterns. Additionally, other citizens cannot benefit from the data provided by others, meaning this approach cannot be directly classified as crowdsensing but rather a distributed collection of services targeting individual citizens.

SmartCityWare by N. Mohamad et al. aims to address various challenges in the development and operation of Smart City services by enabling effective integration and utilization of the concept of Cloud of Things and Fog Computing [174]. The system abstracts the functionalities and components of Smart City applications as services accessible through the service-oriented model. This facilitates the integration, flexible utilization, and combination of various required services, improving efficiency and reducing costs. This work also focuses on city-operated sensors; one example of its applicability is intelligent traffic light control based on fleet data. While citizens are intended to provide feedback through various interfaces, they are not actively integrated into the system. Overall, the system's architecture is highly centralized and city-focused, without considering active participation from local businesses or other stakeholders.

The Smart City architecture developed within the SOFIA (Smart Objects For Intelligent Applications) project employs an event-driven approach to manage and collaborate with various sensor types for monitoring public spaces [82]. The main components of the proposed architecture are Knowledge Processorss (KPs) and Semantic Information Brokers (SIBs). KPs generate and consume event notifications that are stored in SIBs and distributed via a publish-subscribe paradigm. This structure enables the aggregation and correlation of data from diverse sensors, allowing for the detection of complex events and achieving higher levels of abstraction. Given the focus on public spaces, it is evident that public authorities are the primary drivers of the system. Although personal data is also collected, it is treated purely as simple events and data points to facilitate the control and detection of more complex events. Thus, participating citizens are considered solely data sources and triggers for further events, with no provision for active participation through other sensors or self-managed data.

Zhang's approach to Smart Cities focuses on shifting the computation of sensor data closer to the data providers to manage better the high volumes of collected and processed data [282]. The network load is reduced by utilizing Fog Computing, thereby increasing scalability. While this also enhances privacy protection, Zhang does not make this a primary focus of his approach. Besides, his work centers on centrally managed sensor nodes, thus not incorporating active citizen participation.

Civitas is a citywide middleware project to connect various stakeholders within a Smart City [262]. Participating nodes connect via a *Civitas plug* to the middleware's central nodes, managed by public organizations and city administration. These central nodes handle the communication and discovery of other Civitas plugs. Further, they provide and host available services. Through the Civitas plug, data is managed by the participating citizens and remains on their local machine. While the project emphasizes the reuse of existing hardware rather than building a completely new hardware network, it is not clear how the data collected by citizens is utilized and processed. The Civitas platform supports only services operated by the city and industry partners, meaning that interested citizens cannot actively contribute their own developed services. Due to centralized management, public institutions have increased knowledge of the entire network and, with growing numbers of participants, form a bottleneck for discovering and managing connected Civitas plugs.

The VITAL platform [198], introduced by R. Petrolo et al., utilizes existing sensors connected through provided Application Programming Interfaces (APIs). This allows individual sensor operators to retain control over which data they wish to make available via the APIs. Services that want to use this data are hosted in the cloud and have access to all available APIs in the network. The platform employs various ontologies to categorize and define the provided data to semantically describe provided data and, therefore, make the APIs easier to use. While active citizens can also make their sensor data available via public APIs, the technical barrier is relatively high. They must have the knowledge and capability to create public APIs, and accurately providing the semantic description of the data also requires substantial expertise. Therefore, the focus is primarily on industry partners and city officials. Furthermore, although data providers can decide which data they want to share, it becomes publicly available, meaning there is no distinction between sensitive data, which they only share with specific services, and public data.

In his work, Khan et al. present a hybrid cloud infrastructure similar to a data space, where the platform integration layer connects with a large amount of data sources and repositories [129–131]. Various stakeholders, such as citizens, city officials, and companies, can manage their own data sources. While the data is locally managed on the stakeholder's server, the service composition layer used to discover existing data sources and other services is a centralized part of the architecture. Participants can create custom services using these central entities on top of the provided service application layer, which provides standardized discovery, processing, and visualization endpoints. Therefore, citizens are actively encouraged to participate and create their own services, but as data providers, they can only define which data is publicly available. This means that they have less control over sensitive data. Additionally, participating citizens can't decide whether to provide their data to specific services; this is all abstracted within the platform's discovery layer.

OpenIoT [237] is an open-source project to connect and make various cloud-based IoT solutions searchable through semantic descriptions of sensors and data. The project focuses on the W3C Semantic Sensor Network (SSN) ontology, which is used to unify different data sources semantically. Additionally, OpenIoT provides a versatile middleware that dynamically filters linked data sources and selects them based on their semantic descriptions. These semantic descriptions are centrally stored in the OpenIoT middleware and made available for searches.

The platform's openness ensures that it is specialized for Smart Cities and can be used across cities. Any publicly available data sources that can be described using the SSN ontology can be linked. This allows citizens to actively participate in shaping the network by providing both data sources and services that utilize the network's data. The current state of the OpenIoT platform is available on GitHub, but it has not been maintained for the past two years. Further, OpenIoT does not support services and citizens in terms of data quality. Each service needs to implement its own quality control and manually remove invalid data sources.

https://github.com/OpenIotOrg/openiot

The *CityPulse* framework focuses on facilitating the development of Smart City services through a distributed system, with particular emphasis on semantic discovery and data analysis [207]. CityPulse aims to enable citywide data integration by utilizing advanced data analysis modules to transform uncertain and incomplete data into reliable information. Additionally, it offers decision support through intelligent data aggregation, event detection, and quality assessment. CityPulse consists of two main components: data stream processing modules interacting with various data sources and adaptive decision support modules providing context-based recommendations. CityPulse applications are cloud-based components that enable continuous monitoring and processing of data streams, allowing applications to retrieve real-time information about the city's status via APIs. However, the primary focus of CityPulse is on utilizing sensor data from the city and local industry. While citizens can use various services and provide feedback, they are not actively integrated in the data collection and idea development processes.

FIWARE [57] is an open-source IoT platform that partially originated from the SmartSantander project. The FIWARE platform offers components for various functions such as context data management, security, and interfaces to IoT data sources. Through these interfaces, FIWARE enables real-time access to sensor data, control of IoT devices, and data analysis. FIWARE uses the Next Generation Service Interface (NGSI) to standardize the various data sources within a city [26]. Unlike other projects that use standardized ontologies, FIWARE defines and utilizes specific data models adopted by the *Open & Agile Smart Cities* network. To encourage local industries to publish their existing data sources, an IoT Marketplace has been initiated. This marketplace allows entities to publish their data sources for a fee, making them available to other companies. FIWARE primarily targets city officials and the local industry through this monetization approach. While citizens benefit from the new services provided, they cannot actively participate in the Smart City platform.

https://oascities.org/

FogFlow [48] is an adaptation of the FIWARE platform that extends it by providing a more streamlined development environment for services that can execute processing not only in the cloud but also on edge devices. The authors illustrate this with an example of an application in public spaces. A missing child's picture is uploaded to the central camera system, and edge nodes near the cameras use image analysis tools to find and track the child in real-time. This approach reduces network traffic and distributes the processing load more efficiently by distributing the computation across multiple nodes. While enhancing FIWARE to distribute service processing, FogFlow focuses on city officials and local industry, providing services to citizens but not actively including them in the data collection, processing, or interpretation.

Another middleware is presented by Apolinarski et al. called GAMBAS [18, 101]. Within this middleware, citizens are actively motivated to collect data with their smartphones and provide it to different services built with the GAMBAS Software Development Kit. While GAMBAS also supports connecting other data repositories, the focus lies in a privacy-preserving use of citizen-collected data via their smartphones. Citizens can decide which data they want to share and whether sensitive data like location should be obfuscated to increase privacy. Therefore, a local preprocessing can be enabled within the GAMBAS smartphone app so that no raw data leaves the citizen's device. A central discovery mechanism provides information to interested services to discover data sources.

The *DIMMER* platform is a microservice-based architecture with a very specific scope: "The goal [...] is to build a service platform [...] and a number of applications aiming at involving different stakeholders to increase the energy efficiency of a city at the district level" [145, p. 4]. The main part of the dimmer platform is its middleware and Smart City services. While the middleware services focus on data discovery, data storage, and sensor connection, the Smart City services provide microservices specific to the energy efficiency use case. Using this specific use case, the authors showed that "the lack of complex middleware technologies and use of simple communication protocols and APIs instead significantly reduces the amount of coordination work involved" [145, p. 6].

Using the learnings from the DIMMER Platform, *InterSCity* was developed as an open-source platform that provides "a high-quality, modular, highly scalable middleware infrastructure to support Smart City solutions that can be reused across cities [...]" [155, p. 2]. To achieve this, Arthur de M. Del Esposte et al. propose a different microservice-based architecture that connects to city resources and stores their data in a centralized manner. Via a provided resource discovery microservice, other services can access a context-aware search API to find relevant data sources. While citizens profit from the services provided, they do not actively participate in the platform or data collection processes.

By using a multi-agent system, J. Aguilar et al. propose their architecture *Autonomic Reflective Middleware for Smart Cities* (MiSCi) that focuses on "making context-aware smart decisions" [6, p. 1]. To achieve this, each physical device, such as sensors, smart objects, and others, is represented as agents characterized by metadata describing the properties of its represented device. Additionally, all prior existing applications, as well as persons within the Smart City, are represented by context-aware agents. These agents help the users to perform their tasks by coordinating and cooperating with each other. While this system divides complex tasks into multiple subtasks within their agents, citizens are not part of the underlying infrastructure and can only use provided services.

Semantic Social Network of Things Middleware (S2NetM) [202] follows the Social Internet of Things approach [159, 215] to create a decentralized network based on different relationships between IoT devices and their owners. Here, citizens are actively motivated to participate not only as users but also as data providers and service developers. Connected sensors are semantically described using a system-specific ontology based on Web Ontology Language (OWL). S2NetM services can find trustworthy and reliable data sources via a centralized semantic search engine by utilizing a relationship model like in other social networks. Citizens can decide which services they want to share their collected data with. Nonetheless, collected data, relationships, and authorization are stored in a central S2NetM node, resulting in a hybrid architecture. Creating a bottleneck for the ever-increasing network size in citywide sensor networks.

4.1.2 *Smart Infrastructure Platforms*

Smart Infrastructure refers, on the one hand, to using advanced technologies to make urban infrastructures more efficient, sustainable, and adaptable. Enabling technologies such as IoT, Big Data, and Cloud Computing are employed to optimize various infrastructure sectors, including energy consumption and distribution, waste management, and water supply. On the other hand, smart infrastructure tries to motivate and integrate its citizens to rethink their usage and behavioral patterns to implement sustainable resource usage. Following sections focus on Smart Energy and Smart Waste, as these areas already feature several approaches for active citizen participation.

Smart energy is one of the primary research fields in smart infrastructure. Due to the increasing energy demand and the growing number of private renewable energy sources, intelligent solutions are needed to distribute generated energy. These solutions should be context-aware to allocate energy to various consumers efficiently.

There are various approaches to improving energy efficiency. One set of proposals aims to enhance the energy efficiency of individual buildings by analyzing the behavior patterns of residents [9, 178, 288]. This involves using various sensors within Smart Homes to reduce heating and energy costs. Another set of proposals seeks to democratize energy trading, encouraging the development of sustainable energy sources and decentralizing energy production and management [7, 151, 179]. This aims to motivate participating citizens to engage actively with their energy consumption. Additionally, localized energy exchange can reduce transfer costs.

Alongside energy consumption, waste production increases with population density and rising affluence [52]. The total amount of waste produced is projected to increase by approximately 70% by 2050 compared to 2016, reaching a total of 3.4 billion tons [251, 281]. More cities are attempting to establish smart waste management systems to address this growing challenge and to reduce infrastructure costs. As with energy solutions, there are individual solutions aimed at helping citizens reduce waste and minimize waste production [30, 193, 231]. Additionally, there are solutions focused on the overall waste collection and management within the city [1, 2, 168, 230]. While these focus on collection and route optimization depending on the state of smart trash bins for trucks, they do not consider any private information that can be gained from people's trash production and share all raw data with waste management.

Depending on the collected and analyzed data, smart infrastructure also involves a significant amount of personal data. Personal energy consumption is closely linked to user behavior in Smart Homes [7, 167], and the type and amount of waste produced can reveal information about residents and their living conditions [36, 257]. When such data is shared and processed citywide, it is essential to actively involve citizens and inform them about their data usage to counteract feelings of technological surveillance [167].

4.1.3 *Smart Mobility Platforms*

Smart Mobility systems are integral to Smart Cities as they significantly improve urban mobility and promote sustainability [95]. Urban traffic congestion, a primary source of pollution and time wastage, negatively affects the environment and residents' quality of life. Various approaches have been explored to address these critical issues.

One of the most extensively studied challenges in the research field of Smart Traffic is optimizing traffic light timings to improve traffic flow. There are two different approaches to estimating traffic volumes and optimizing traffic light signals accordingly. The first one involves using Floating Car Data (FCD) [66, 127, 270, 284], which is data collected from vehicles or their occupants, and the second approach relies on fixed hardware installations on roads [117, 172, 189, 191]. When collecting FCD, privacy considerations are crucial, as the entire movement profile of individuals can be recorded [210]. Similarly, external hardware can also capture private information, such as license plate numbers, through identifiable features [269]. Different approaches implement various solutions to ensure a certain level of user anonymity. Still, the collected data is mostly directly sent to a central server, and only Zhang et al. propose an architecture where citizens can manage their collected data and decide with which service data should be shared [284].

While previous studies have primarily focused on motorized traffic, other approaches concentrate on pedestrian and bicycle traffic. These efforts emphasize not traffic flow optimization but rather the safety of individuals. Typically, smartphone users are employed for this purpose [12, 241, 242], although there are also studies that aim to estimate pedestrian and bicycle traffic density through counting from individuals or dedicated sensors [102, 280]. These numbers are then used to train AI models to estimate the total numbers even when no actual count data is available [280]. Similar to before, when smartphone data is used, privacy requirements are similar to those of motorized traffic data.

4.1.4 *Smart Environment Platforms*

One focus in the Smart Environment area lies in monitoring and enhancing environmental quality in urban areas [95]. A primary concern is measuring air and water quality to detect harmful extremes early and enable prompt intervention. Additionally, the increasing temperatures caused by climate change, particularly the phenomenon of urban heat islands, significantly impact the health of vulnerable populations, such as the elderly and young children. These systems often integrate urban monitoring technologies with active citizen participation and community initiatives to collect and analyze comprehensive and accurate environmental data.

In their 2024 air quality status report, the European Environment Agency concluded that 96% of Europe's urban population was exposed to fine particulate matter concentrations exceeding the levels recommended by the World Health Organization [77]. To better understand air quality in various urban locations, several approaches have been proposed [34, 69, 85, 161, 204, 290]. The availability of affordable, compact, and energy-efficient sensors necessary for air quality measurements has led to increased efforts to use mobile sensors next to fixed measurement stations. To increase data coverage, different approaches try to engage citizens in data collection actively [34, 69, 85, 290]. While these projects focus on mobile data collection and ensuring data quality, other approaches rely on fixed sensor installations with a standardized setup and provable data quality and investigate the efficient distribution and storage of data [161, 204].

In the past few decades, water quality monitoring systems have evolved from manual lab-based monitoring to manual in-situ monitoring and finally to modern Wireless Sensor Network (WSN)-based solutions [4]. While the manual lab-based approach required water samples to be collected and sent to a laboratory for analysis, modern sensor technology allows for automated, real-time data collection [4]. Despite these advancements, the specific and cost-intensive nature of sensor hardware, coupled with the challenges of deploying sensors in public waters, has led to the adoption of centralized architectures based on sensors deployed by city officials [46, 92, 121]. Some studies involved citizens manually assessing water quality by handing simple water sample kits and forms to fill out [35, 123]. However, these methods lacked the technical infrastructure for continuous monitoring, resulting in only single timestamp measurements.

Urban Heat Islands (UHIs) pose a significant threat to vulnerable groups in the city, particularly the elderly and young children. Knight et al. placed affordable thermometers at various schools in Manchester to measure temperature differences across the city and engage students with the topic [142]. However, this involved a lot of manual work, and no automated system was established. In contrast, Fekih et al. used mobile sensors that individuals can wear, actively encouraging citizens to collect data to identify UHI [81]. Nevertheless, the participants' privacy is not adequately considered, even though movement data can reveal clear behavioral patterns of the participants.

4.2 COMPARISON OF PLATFORMS

After describing existing platforms, architectures, and data collection mechanisms, a comprehensive comparison is conducted. This comparison aims to identify any remaining gaps in current research. For clarity, the table does not cover all the requirements established in the previous chapter; instead, it focuses on different criteria, such as architectural differences, data storage, data sources, and citizen participation. Therefore, several requirements are grouped into these criteria in the following table to highlight the differences among the various platforms.

4.2.1 *Comparison Criteria*

In total, related work was compared using eight different criteria. While the first three focus mainly on non-functional requirements, the last five concentrate on groups of functional requirements. Before presenting and discussing the table, the individual criteria are explained in detail, and the associated requirements, if applicable, are listed.

Architecture

The architecture of a system defines how the data flows and who is in charge of service deployment, data discovery, authentication, and authorization. Within a central (*c*) architecture, there is a single coordinator. On the contrary, in a decentral (*d*) architecture, the platform is orchestrated by a multitude of systems working together. Often, each system has a specific role within the network, but redundant components can also be deployed to increase availability and resilience. Other architectures make use of a hybrid (*h*) approach. Here, some components are still centralized (most often discovery or authentication), while other functionalities are deployed in a decentralized manner. Since various functional and non-functional requirements can be implemented using different architectures, there are no requirements associated with this criterion directly, but a systems architecture influences nearly all of them.

Data Storage

One of the most crucial questions in a Smart City platform is where collected data is stored. With the number of data sources, the collected data also increases drastically. Some platforms integrate a centralized (*c*) storage, where all connected sensors push their data. This leads to a data warehouse or data lake, depending on its data representation. Another approach is a decentral (*d*) data storage where multiple repositories exist, and sensors decide on a defined strategy for storing their data. Nonetheless, these data repositories are still managed by the same authority. Lastly, data can be stored locally at the sensor owner in the form of a distributed data space (*ds*). Data owners can then give access to their data to interested services or other network participants. This criterion encompasses the following non-functional requirements: availability, privacy, control over data usage, and scalability.

Description of the data warehouse and data lake concepts can be found in the fundamentals section.

Data Sources

Existing platforms focus on different data sources. These data sources can either be owned by city officials (*co*), industry (*i*), private citizens (*pc*). Depending on the data sources different requirements can occur for the Smart City platform. When focusing on city official and industry-owned data sources, a supervised sensor setup can ensure data quality, and data exchange, access, and authentication are often clearly defined by contracts. When including citizen-owned devices, securing data quality becomes a major task, and securing private data, creating a user-friendly interface, and engaging with communities are additional requirements.

Third-Party Services

While some Smart City platforms are designed for a specific use case, others function as middleware that allows third parties to create their own services utilizing the existing data from the platform. Introduced platforms can be categorized into three distinct types: platforms that do not permit the deployment of new services (*n*), platforms where only the platform operators can deploy new services (*p*), and platforms that allow the deployment of services developed by any third party (*y*). When opening the Smart City platform for external services, platform developers need to create service development and management tools, as well as secure data access and transparency.

Service Processing

When services can be deployed on the Smart City platform, they can either run centrally (*c*) on the platform or developer infrastructure or decentral (*d*), which means that the computation of the service is divided and only partially runs on the service providers hardware. One common use case is local preprocessing to reduce network load and increase privacy for the data providers. By dividing a service's processing into multiple devices, the following requirements need to be focused on: Service Management, Control over Data Usage, Software Development Tools, and a user interface that helps data providers understand which data is preprocessed and what is transferred to the service itself.

Quality as a Service

When a Smart City platform provides Quality as a Service (QaaS), it helps service developers determine the quality of specific data sources and their provided data. QaaS can range from local outlier detection and removal to analyzing the reliability depending on surrounding sensors within the network. By providing quality information, service developers can make informed decisions on which data sources to use and which additional steps are needed to ensure service quality. This significantly impacts the data quality requirement, data processing, and service management.

Citizen Participation

Smart City platforms can support citizen participation in various ways; the simplest form is to allow for feedback and citizens profiting from the provided services of the platform (+). The next step is to involve citizens in the data collection and motivate them to actively participate within the platform and data generation (++). Finally, citizens can also create their own services, manage their own data, and contribute in different ways to the Smart City platform (+++). With increasing citizen participation, a lot more requirements need to be considered. First, a user-friendly interface should be created so that even non-technical proficient citizens can participate. Here, citizens need to manage connected sensors, their collected data, and the services they provide their data for. Second, privacy and control over collected data usage are of focus for privacy-conscious citizens.

Data Sharing

While data sources can provide their data to specific services, sharing it without a pre-specified purpose is another usage of collected data. This can be useful when communities or research groups work together and want to share all their data to help others with a similar intent. This allows data providers to opt out as they wish but also allows participants of these data-sharing communities to use the data without requesting access for each use case or study. Therefore, this mainly focuses on data management and data usage control requirements.

4.2.2 *Comparison*

The results of the comparison can be seen in Table 10 and Table 4. The results have been split into two tables: the first contains all the general platforms, and the second contains use case-specific platforms and approaches. Papers with the same comparison results are grouped into a single row to make the tables less redundant and easier to read. In total, 17 general purpose, 13 smart infrastructure, 13 smart traffic, and 11 smart environment platforms and approaches have been investigated.

While most general platforms rely on hybrid architectures with distributed data storage, most use case-specific approaches employ a central architecture and data storage. Exceptions are found in the Smart Infrastructure domain, particularly in Smart Energy applications, where collected data is typically stored at the respective buildings or consumers. Interestingly, few platforms in this group support adding new functionalities, meaning the collected data is usually only used within the system for its initially defined purpose.

Despite this, many approaches involve citizens, although they are mostly viewed solely as data sources and cannot actively participate. Only Moreno et al. where "Occupants play a crucial role in the system's operation to achieve energy efficient building performance" [178, p. 1]. Further, the only approach that allows for distributed service execution is [7], which employs a blockchain for managing data and smart contracts executed in a distributed manner across participating nodes.

In each group, some platforms consider not only individual interest groups such as city officials, industry, or private citizens. However, city officials are the most frequently represented group overall. This prevalence can be attributed to the funding of research projects and the provision of city-wide infrastructure. Notably, twelve of the presented platforms do not rely on data from the city itself but instead utilize data solely from private citizens or industry partners.

While all but two general platforms allow the deployment of third-party services, only six of the specific use case platforms enable services to be deployed and use collected data. Nonetheless, only six platforms allow for a distributed service execution, dividing the computing load between participating hardware. Further, only five platforms provide any QaaS, even when large amounts of citizen collected data is processed. This results in services using these platforms first needing to be deployed on powerful hardware nodes that can process various data sources and live data streams. Second, all quality assurance and filtering must be done for each service individually, resulting in a redundant load on individual services.

In total, only four investigated platforms support a high degree of citizen participation by letting citizens collect and manage their own data as well as giving them the tools to create their own services and access data from others. Most data platforms see participating citizens as data sources and do not give them control over their collected data (twenty-one times). Three platforms allow participants to collect and manage their own data but provide no possibilities to participate in different ways with the Smart City.

Lastly, only four platforms allow participants to share their data freely with others without a specified service or use case. This allows participants to provide their data to open data communities, making it easier for start-ups or research groups to test new ideas and approaches with real data.

4.3 RELEVANT RESEARCH GAPS

After analyzing the established tables, the following section interprets the results and identifies the remaining research gaps.

One requirement discussed in the requirements analysis pertains to the sustainability of Smart Cities and the goal of operating resource-efficiently. The platforms examined, however, often relied on use case-specific new hardware. Only the use of existing smartphones in individual studies did not require new acquisitions and was based on already existing hardware. Sustainability can be improved by integrating the various use cases into a general platform that allows collected data to be shared and utilized across different use cases.

For instance, data from the three groups can be combined to identify new dependencies. Traffic patterns are likely influenced by weather conditions: on dry, warm days, more people are willing to use alternative modes of transportation (motorcycles, bicycles, e-

Group	Paper	Architecture	Data Storage	Data Source	Third-Party Services	Service Processing	Quality as a Service	Citizen Participation	Data Sharing
General Platforms	[47]	c	c	co	y	c	p[1]	++[2]	n
	[41]	d	ds	pc	n	c	n	++	n
	[174]	h	c	co	p	d	n	+	n
	[82]	c	c	co	p	c	n	+	n
	[282]	c	d	co	n	c	n	-	n
	[262]	h	ds	co	y	d	n	+	p
	[198]	h	d	co, i, pc	p	c	n	++	p
	[129, 130]	h	d	co, i, pc	p	c	p[4]	+++	y
	[237]	h[3]	ds	co, i, pc	y	c	n	+++	y
	[207]	h[3]	ds	co, i	y	c	p	-	n
	[57]	h	ds	co, i	y	c	n	-	p
	[48]	h[3]	ds	co, i	y	d	n	-	p
	[18]	h	d	co, i	y	d[5]	n	+	p
	[145]	d	d	co	y	c	n	-	n
	[155]	c	c	co	y	c	n	-	n
	[6]	h	d	co	y	d	n	-	n
	[202]	h[6]	d	pc	y	c	n	+++	y

Table 3: Comparison of existing Smart City Platforms regarding the prior introduced criteria.
c = central, d = decentral, h = hybrid, ds = data space, co = city officials, pc = private citizens, i = industry, y = yes, n = no, p = partially

[1] Anomaly Detection
[2] An example study used a smartphone app to collect citizen data, which are not counted as citizen-owned sensors since the collected data is directly uploaded to the app's server. Nonetheless, it increases citizen participation.
[3] Central discovery
[4] Service developers can run stochastic analysis on provided data to help the analyze the quality.
[5] Only Smartphones
[6] Central lookup for relationships and access rights

Group	Paper	Architecture	Data Storage	Data Source	Third-Party Services	Service Processing	Quality as a Service	Citizen Participation	Data Sharing
Smart Infrastructure	[178]	h	c	co, i, pc	y	c	n	+++	y
	[288]	d[7]	d	pc	n	c	n	+	n
	[9]	h	c	co, pc	n	c	n	+	n
	[7]	d	d	i, pc	n	d	n	+	n
	[179]	d	d	i, pc	n	c	n	+	n
	[151]	h	d	co, i	n	c	n	-	n
	[231]	h	c	pc	n	c	n	+	n
	[30]	d[7]	d	pc	n	c	n	+	n
	[193]	d	d	co, pc	n	c	n	+	n
	[1, 2, 168, 230]	c	c	co, i	n	c	n	-	n
Smart Traffic	[66, 127]	c	c	i, pc	n	c	n	+	n
	[270]	d	d	i, pc	n	c	n	+	n
	[284]	d	d	pc	y	c	n	++	y
	[117, 172, 189, 191]	c	c	co	n	c	n	-	n
	[12]	c	c	co, pc	n	c	n	+	n
	[241, 242]	h	d	co, pc	n	c	n	+	n
	[102]	c	c	co, pc	n	c	n	+	n
	[280]	c	c	co	y	c	n	-	n
Smart Environment	[34]	c	c	co	n	c	n	-	n
	[69]	c	c	pc	n	c	n	+	n
	[85]	c	c	pc	n	c	p[8]	+	n
	[290]	h	d	pc	p	c	n	+	n
	[161]	c	c	co	y	c	n	-	n
	[204]	c	c	co	n	c	p[9]	-	n
	[92, 121]	c	c	co	n	c	n	-	n
	[46]	c	c	co	y	c	n	+	n
	[35, 123][10]	c	c	co, pc	n	c	n	++	n

Table 4: Comparison of existing approaches focusing on different use cases regarding the prior introduced criteria.

[7] Data is only managed locally and not shared between smart homes.

[8] Local outlier removal.

[9] Central anomaly detection.

[10] One-time citizen science projects. Therefore, they show only a snapshot at a specific time.

scooters) instead of cars, while higher traffic volumes are expected on rainy or extremely hot days. Conversely, local traffic conditions and weather data, such as wind, humidity, and temperature, can be used to classify areas in the city by their estimated air quality.

Additionally, the consumption of resources by citizens and the entire city is linked to current weather conditions. On warm days, heating decreases, but water consumption increases [43]. Traffic conditions can also impact citizen behavior and energy consumption. If poor traffic conditions are expected to result in longer travel times, citizens might leave for work earlier, heating their homes less on cold days, or opt to stay home and increase private energy consumption.

These simplified examples illustrate that city-wide data exchange is desirable and can lead to identifying previously unconsidered dependencies in various areas. This work aims to build on the existing knowledge from general platforms to enable a city-wide citizen-operated sensor network that integrates data from diverse scenarios and sectors.

As outlined in the requirements, participating citizens should be able to collect and manage their own data. Given the growing awareness of privacy and the increasing sense of constant surveillance by embedded systems [120, 160], returning control over data usage to the citizens is essential.

In previous approaches where citizens could manage their collected data and decide on its usage, centralized discovery, authentication, and authorization mechanisms were used. While this centralization ensures a single source of truth and reliable information about all available data sources and participants, it also means that the central infrastructure and its administrators can gather extensive metadata about searches, accesses, and individual behaviors. Moreover, a centralized discovery system quickly becomes a bottleneck, negatively impacting the performance and scalability of the entire system. This issue is further exacerbated by the expected highly fluctuating network of resource-constrained sensor nodes.

Previous works have assumed stable networks with constant connectivity between participants. However, continuous network joinings and disconnections are expected in a network composed of many privately managed data sources. The network is thus in a constant state of flux and must be able to handle this uncertainty. The requirement for adaptable and context-aware data discovery tools is unmet in current works and needs further investigation.

Another disadvantage of using a central management instance is the preservation of individual citizens' data sovereignty. Proposed solutions partially allow citizens to store their collected data themselves; however, they must share their metadata with a central instance. This metadata generally contains access rights and the individual citizen's available data. The extent to which this data is subsequently processed and analyzed falls outside the citizens' decision-making authority. From a sociological perspective on Smart Cities, the concern is that citizens are viewed primarily as data sources while being less integrated into the active domains of a Smart City [40, 110]. For these reasons, this thesis conducts a deeper analysis and evaluation of these issues.

In addition to the increasing demands on the underlying network for connecting and discovering relevant data sources and privacy concerns, the growing number of citizen-operated data sources necessitates a focus on data quality. This aspect has been largely overlooked in previous works and has not been provided as QaaS at the network level. There are two primary methods for ensuring data quality: outlier detection, which identifies data points in a dataset that deviate from expected norms and may indicate faulty data sources or incorrectly placed sensors, and anomaly detection, which typically examines entire data streams to detect unusual behavior. While outlier detection focuses on individual erroneous data points, anomaly detection can identify patterns, such as recurring errors at specific intervals, suggesting an external influence rather than a faulty sensor.

For example, temperature sensors placed on different sides of a building may record higher temperatures when exposed to sunlight earlier than others. Such sensors are not faulty, but their data might need to be ignored or corrected during certain times. Only the work by Cheng et al. includes an anomaly detection system, but it does not support the integration of citizen-operated sensors [47].

External factors cause most expected anomalies, but they can also result from the behavior of the sensor owners. For instance, grilling in the garden might temporarily reduce air quality due to smoke, while being on vacation would result in consistently low energy consumption. Previous works have not addressed the private data contained in the raw data. Collected data has been passed to services in its raw form, and it remains to be explored how these private details can be extracted and kept private without negatively impacting the services, making sure to adhere to the data sovereignty of the participating citizens.

4.4 SUMMARY

In this chapter, existing research works were summarized and examined in relation to their solutions for the requirements from the previous chapter. From this overview, relevant research gaps were subsequently investigated. The identified relevant research gaps can be summarized as follows: First, there is a need for deeper investigation into how discovery mechanisms can be implemented in a network of equal participants, meeting scalability, efficiency, and context-awareness requirements without a central entity that can collect metadata through connections and become a bottleneck. Second, to address citizens' concerns regarding data privacy, services must account for data sovereignty and respect their privacy. Further investigation is required into how to ensure that no private information about data providers is embedded in raw data shared with services, especially those not fully trusted. Lastly, research is needed on how services can ensure data quality when sensor placement cannot be monitored.

After identifying these research gaps, the following chapter presents an overall architecture as a proposed solution for a citywide citizen-centric platform. This platform aims to network the citizens' existing sensor hardware and additionally provide participating citizens with the opportunity to offer various Citizen Science services. These services should then be able to access the available data streams while maintaining the data sovereignty of other citizens.

5 DATA SPACE FOR CITIZEN SCIENCE SERVICES

Previously, related works in various Smart City domains built upon data-driven networks were presented. These often involve citizens but rarely beyond the role of data sources. As the requirements analyses for citizen science (Section 3.1.3) highlights, it is crucial that citizens are not marginalized into being mere data sources by technological advances but are actively integrated into other aspects of the Smart City concept [84]. Additionally, the importance of reusing the widely distributed and existing sensor infrastructure to enhance sustainability and reduce costs was highlighted.

Three relevant research topics have been identified based on these two fundamentally different approaches. First, new distributed context-aware discovery mechanisms are needed to connect and make an ever-growing number of autonomous sensor nodes searchable. Second, data processing should be investigated in greater depth to determine how services and data sources can share the processing tasks. This would give data sources more privacy and data sovereignty while distributing the preprocessing and filtering load between the data sources and the services. Third, methods to assess the quality of self-managed data sources should be explored so that services can define quality criteria for the data they use.

This chapter introduces the concept of a citizen-centric network that allows citizens to manage their data sources and the collected data. Further, they can connect services to that network to use the existing data in a citizen science approach. These services need to adhere to the privacy settings and maintain the data sovereignty of the data sources to increase transparency and, therefore, the citizens' trust in the network. These services offer added value for individual target groups or the entire city by utilizing collected data.

To achieve these goals, a concept is discussed, and required mechanisms are highlighted. The proposed system architecture implementing the developed concept is then explained in two steps: First, the main component of the network, the *Connector*, and its functionalities are described. Next, the overall network architecture is outlined, detailing the interaction between participating Connectors and provided Services.

5.1 CONCEPTUAL CITIZEN-CENTRIC NETWORK

To enable a citizen-centric sensor network as described in the requirements analysis (Section 3.3), the fundamental question is how citizens can be empowered to manage their collected data autonomously, retaining data sovereignty. Multiple distributed data management architectures were presented in Section 2.2: *Data Warehouses* prescribe a central entity with a fixed data structure, which, given the heterogeneity of the expected sensors and data, requires significant organizational effort. Furthermore, the centralization results in the loss of citizen data sovereignty, making this architecture unsuitable as a foundation. While *Data Lakes* allow data to be stored in distributed repositories without a fixed data schema, access rights are not fine-grained but binary, allowing either full or no access to the raw data. Additionally, the lack of a standardized description means services for various data sources would need to define different preprocessing steps to link data from diverse sources. This is undesirable from a citizen's perspective; only a few would be willing to grant full access to all collected data. From a service perspective, this architecture would entail substantial effort to use data from various sources. Lastly, *Data Spaces* offer a solution through decentralized management and fine-grained access rights, enabling participating citizens to store their data locally and individually decide whether a service can access their data. Another advantage is that participants can use different data representations. Services can use the data without needing individual reformatting through a standardized access interface and semantic data descriptions.

To implement a data space, each participating citizen needs a software platform to manage the locally collected data and enable access to it for other participants through the data space. To achieve this, this platform must function as a *gateway*, allowing the owner to transmit data from existing sensors through the gateway to interested services. This is reflected in the functional requirement of *Sensor Management*. Due to the heterogeneity of existing hardware, sensors must be integrated through a variety of standards and communication methods. In addition to managing sensors, the gateway must also allow for the management of collected data, as described in the *Data Management* requirement. This includes functions for managing storage and sharing in the data space. It is essential that citizens can not only configure data access but also transparently view network visibility for the owner. This strengthens privacy and prevents third parties from obtaining more meta information about existing data sources.

As previously highlighted, participating citizens should also be able to provide services that utilize the provided data to offer functions or extract new knowledge. To provide local services, the *Service Management* requirement must be fulfilled. Here, citizens need the ability to create new services, connect new data sources, and manage already running services. These services can be implemented in two ways. First, they can be directly integrated into the data space software and, thus, the gateway. To enable this, the system must meet the functional requirements of *Application Runtime*, *Software Engineering Tools*, and *Data Processing*. An advantage of directly integrating services within the gateways is that service providers must adhere to established rules and can only use the provided functions within this runtime. However, a disadvantage arises for existing services that wish to find new data sources through the data space. These services would need to be re-implemented within the own application runtime. Additionally, it must be ensured that the application runtime is powerful enough to enable various services while being easy to understand and learn. Second, the gateway can also offer interfaces for externally connected services, allowing them to access data in the data space through the gateway. This separates the logic between the services and the gateway and enables services to be developed using various tools and languages, giving citizens more freedom to choose preferred tools and further connect existing services to the data space. Contrary to the first approach, the gateway must now meet the requirements for *External Data Access*, particularly addressing *Privacy* and *Security* aspects to ensure that services can only access and use the data they are authorized for.

With the local linking of sensors and the provision of services, the gateway must also provide the ability to connect with other gateways efficiently and securely, enabling data exchange between services and relevant sensors functioning as data sources. Three requirements must be considered for this purpose.

First, there must be a *Network Infrastructure* that meets the non-functional requirements of *scalability*, *extensibility*, *availability*, and *robustness*.

Second, services should be able to efficiently find existing data sources using this network infrastructure to support the requirement for citywide *Data Discovery*.

Finally, a common understanding must be established using a *City Model* due to the heterogeneity of sensors, services, and the transferred data. This model should be defined in the context of the data space but must be able to be extended and adapted.

5.2 LOCALLY MANAGED DATA GATEWAY - CONNECTOR

Based on the previously discussed concept, the following section describes the architecture of a citizen-operated gateway called *Connector* that enables locally available sensors to connect with a citywide citizen-centric data space. It also provides the ability to store and manage data locally and create and manage services.

By connecting these decentralized Connectors, a city-wide data space is created, allowing citizens to collect and manage their data in a distributed manner and to decide at a fine granularity with which services or other participants' data should be shared. As defined in the Data Space architecture, each participant is connected to the data space with their own Connector, which acts as a gateway to its internal data.

The interplay of the Connectors and the resulting overarching functionality are then described in the form of so-called layers, each representing different functions of the citywide data space.

5.2.1 *Functionalities of the Connector*

While this thesis mainly focuses on citizens, participants can also include institutions or industries that wish to contribute data to the network or benefit from the available data.

Connectors represent the most crucial component of the proposed network structure. Each participant connects to the existing network through their Connector and can communicate with other Connectors in the network via a common protocol.

Connectors are software executed on the participant's hardware. They must provide the following functionalities, as established by the requirements analysis:

SENSOR MANAGEMENT: Participants should be able to connect existing sensors to the Connector through various interfaces and make their data streams available to the citywide data space.

DATA MANAGEMENT: Participants should manage their own data, including data persistence and an overview of how their data is used by services.

SERVICE MANAGEMENT: Participants should manage their provided services and connected data sources. Additionally, they should have an overview of the services they provide data to or those interested in their data.

ACCESS CONTROL: Connectors must enable participants to set access rights for their provided data streams, allowing them to share only the data they wish to share consciously.

While these functions represent the user interface of the Connector that participants interact with, additional background functionalities ensure the fulfillment of other functional requirements. These background functions include:

DATA DISCOVERY: When a service wants to use data streams from the network, it must define the required data. The network must be able to identify all relevant data sources that meet the defined criteria and request their data.

DATA PROCESSING: When services use data, the sensor network should preprocess the data as much as possible on the Connector of the data source, as described in Section 4.3, to protect the privacy of the data source and distribute the load.

DEFINITION OF A CITY MODEL: To enable services to semantically describe their required data and identify it within a network of heterogeneous data sources, all participants in the network must share a common semantic understanding. This should be ensured through a universally applicable City Model.

Additionally, *Data Quality* was identified as another requirement in Section 3.3 that should be offered as network services. Since assessing data quality requires access to data from multiple sensors from different participants, therefore, this service should be provided as opt-in function. This approach ensures the data sovereignty of the participants, allowing them to decide to what extent they want their data to be evaluated.

5.2.2 *Component-Based Architecture*

The Connector consists of several modular components, each performing a dedicated function. To give users the possibility to configure each component, each provides their own user interface. These modular interfaces are then integrated into an overarching Connector interface that allows users to manage all relevant information and gives them an overview of its collected data, provided, and participating services. Due to the modular architecture and independent user interface, each component can be paused or deactivated as needed. This may be relevant in cases where, for instance, local data persistence is not required, or the Connector does not provide any services.

Figure 6 abstractly depicts the individual components. Each component provides a single functionality and communicates via so-called events. A central event engine facilitates communication between components. This communication architecture decouples the components, allowing individual components to be deactivated without affecting the functionality of the other components.

A network node must provide various integration options for existing data sources to share collected data with the network. The data interface component can either be actively called by existing data sources (push method) or continuously query the data source to request current data for local processing (pull method).

When new data is received, it is distributed via the local event engine to existing components interested in the data. This can include persistence if data needs to be archived locally and notifying all services that have subscribed to this data stream. These services can be provided locally on the same Connector or remote Connectors within the same data space. Access to the provided data streams can be finely controlled for each service using access control. Once access is granted, new data records are proactively forwarded via the network interface.

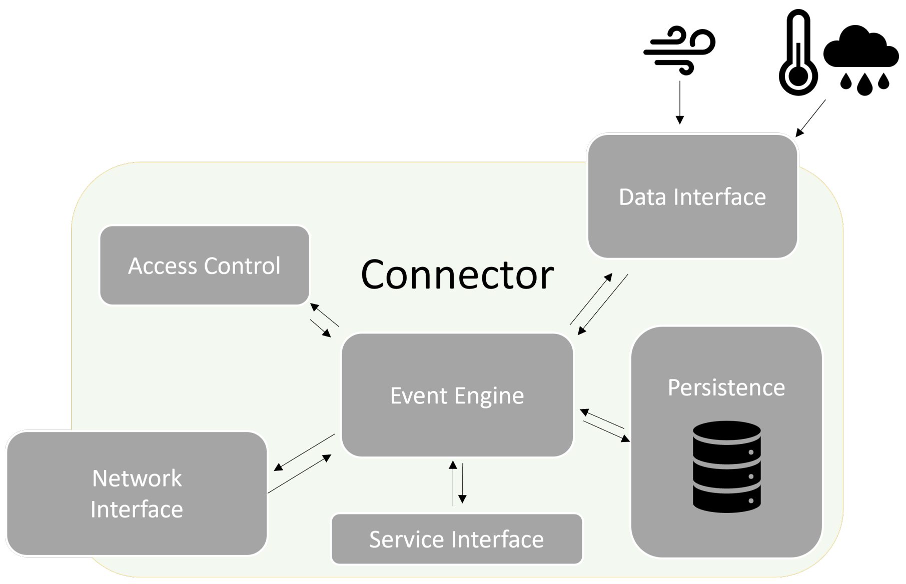


Figure 6: Component-based Architecture of a single Connector. Showing the communication channels between each component and a central event engine.

The following sections provide a detailed analysis of each individual component.

Data Interface

Since the citywide networks focus on increasing the sustainability and reuse of existing sensor infrastructure, one of the major requirements is to integrate existing data sources and, therefore, make already available data more accessible. Thus, existing infrastructure should be reused as much as possible, and new data should be provided through the proposed network structure alongside existing processes.

In the following multiple possible communication methods to integrate existing data sources are proposed:

1. If new data sinks can be defined at the data source itself, the data interface provides endpoints through which new data can be published to the network. This is the simplest solution but requires configurable data sources that use the same protocol as the data interface.

2. If data sources offer an interface to query the latest data, the data interface can request the newest dataset at configurable intervals. This method does not require any changes to the data source itself but assumes a common protocol for data exchange.

3. If data is already processed or stored locally, data can be mirrored from the existing data flow or storage to the network node

using *integrators*. An integrator is a dedicated software solution that must be adapted and modified based on the existing infrastructure. Thus, similar to point 2, no changes to the data source itself are necessary, but the complexity of the integrator can vary depending on the existing infrastructure and software used.

These different communication methods are not mutually exclusive; they allow various data sources to publish their information through the data interface in different ways. When a new dataset is published, this information is forwarded to interested components and services via the internal event engine. This ensures that new datasets quickly and efficiently reach the appropriate recipients. This type of data distribution enables seamless integration and interaction between the different components and services, ensuring effective communication and collaboration within the system.

When a new data source is initialized, various information must be provided to describe the data source semantically. As stated in Section 3.3, the semantic description of data sources and their measurements is a crucial component of an interoperable, extensible, and reusable data space. Only through the semantic description of the data source can the heterogeneous sensors available in the network be interpreted and utilized by various services. Section 5.4 describes the data model used and the descriptions of various data sources in-depth.

Persistence

In a distributed network for streaming data, persistence is an optional component that allows for storing collected data in a permanent repository, enabling access to historical data upon request. This component allows network nodes to archive and store their collected data for future analysis and evaluation.

There are three configuration options the persistence component needs to provide:

NO PERSISTENCE: Current data is processed in real-time but not stored anywhere. This setting does not require additional storage space but does not allow for requests for historical data. This setting makes it lightweight and simple to set up.

INTERNAL PERSISTENCE: Current data is continuously stored in an internal data repository on the hardware node and made available for requests for historical data. Various settings can be configured for data retention to adjust the available storage space.

EXTERNAL PERSISTENCE: Existing infrastructure can be set as external data repositories to avoid duplicate storage. This requires the existing infrastructure to provide interfaces that allow access to the stored data and enable search queries for defined data.

Integrating a persistence component into the citywide sensor network allows for responding to requests that include historical data, in addition to subscriptions to the current data of a data source. These requests can either be performed automatically at predefined intervals or as one-time manual queries. Such manual queries can be significant for analytical purposes or for reviewing past events. For instance, historical data can be analyzed to identify long-term trends or uncover recurring patterns.

As mentioned earlier, the persistence component is optional and can be deactivated. This function primarily targets users with limited resource-limited hardware and storage, where no persistence is applicable. However, this node can then only provide live data.

Event-Engine

As the central communication component within a Connector, the event engine interacts with other components by distributing local events to the available components. Through an internal publish-subscribe communication model, individual components can publish their own events and subscribe to the events of other components. This communication method ensures that local components are loosely coupled. Therefore, the overall system continues to function even if a single component fails, is shut down, or triggers an internal error. Local events can be categorized as follows:

DATA EVENTS: A new dataset has been collected, which can either originate from a locally managed data source or from an external data source that provides data for a local service.

REQUEST EVENTS: These are requests sent to the local Connector from other Connectors and services. They can be either subscription requests or requests for historical data.

INTERNAL EVENTS: Internal events are those relevant to internal communication between components. These are mostly used for configuration but can also include abort or error events.

Figure 7 shows an exemplary communication sequence between the Data Interface, Event Engine, and Persistence components. In the first step, the Data Interface publishes a data event; however, no other component is yet subscribed to this event. As a result, the event is not forwarded and is ignored by the Event Engine.

In the next step, the user configures settings to store the data, and the Persistence component sends a data subscription event for the specified data sources to the Event Engine. When the Data Interface

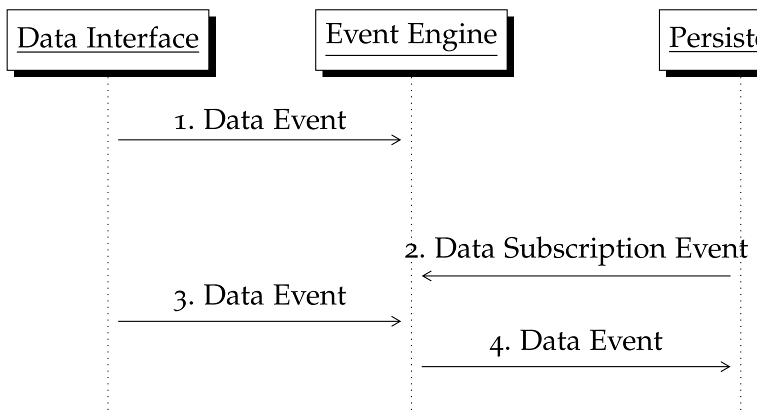


Figure 7: Sequence diagram showing the communication events between the Data Interface, Event Engine, and Persistence component.

publishes a new data event, the Event Engine forwards it to the Persistence component, as the latter has expressed interest in these events.

Access Control

With the help of the access control component, the owners of a Connector and its linked data sources can distribute fine-grained access rights to services. The focus here is on the decision-making power of the owners and the transparency of how their data is used. As outlined in the requirements, a citywide sensor network focused on citizen participation is expected to include not only tech-savvy participants. Therefore, an intuitive and transparent access control mechanism should assure every participant that their collected data is not misused. Participants must also be confident that their data cannot be used to infer details about their private lives or behavior.

Every newly linked data source is initially set to *private* to ensure this. This means the data source is not published in the citywide sensor network. Consequently, no other participant knows that this data source exists and cannot request or use it. If the user wants to make the data from the new data source available to other participants, there are two options: *protected*, used for potentially sensitive information, and *public*. Sensitive data may contain personal information or allow inferences about the owners' behavior. Although protected data sources are published in the citywide sensor network, an external service must obtain active consent from the owner to use the data. For this purpose, the service declares how the data is preprocessed on the owner's Connector and what content is ultimately sent to the service. Lastly, there are public data sources. These are data sources where the owners are confident they do not contain sensitive information. Thus, they can be used without a request for manual approval. Subscribed services receive the data in its raw form. Local preprocessing is also possible, but it requires manual owner approval, as it

Local preprocessing ensures that not all data is sent as raw data. Rather, the owner can understand how the data is used and can make an informed decision about whether their data may be utilized.

actively uses the owner's resources and might create energy costs or negatively influence the performance of other components and services.

Network Interface

The network interface component is the communication channel between the connected Connectors within the citywide sensor network. The Connectors form a decentralized network responsible for discovering data sources and establishing connections between two Connectors. The main function is to allow services to discover matching data sources as input and safely transfer the data streams from the data source to the service.

Figure 8 shows a sequence diagram with two Connectors who want to exchange data. A service on *ConnectorA* (top) is interested in a data stream provided by *ConnectorB* (bottom). To receive their collected data events, it sends a *Request Data* event with the remote address to its internal *Event Engine*. The *Network Interface* is subscribed to all events with a remote address as a target. Therefore, the event is forwarded to the local *Network Interface*, which serializes the event and sends it over the network to the *Network Interface* from *ConnectorB*. The message is deserialized and sent to the *Event Engine*. As described in Figure 5.2.2, the *Access Control* component manages all access rights and is, therefore, subscribed to all *Request Events* from remote addresses. The access control now needs to check the access rules for the requested data source. If it is *public*, it adds the remote subscription directly to the event engine. When access rules are set to *protected*, it waits for the user's decision on whether a subscription should be added. When the subscription is added successfully, the access control component creates an *Accept Request* event sent to the remote service. When the service receives the *Accept Request*, it sends a *Data Subscription* event to the local event engine to forward future data events.

Since few participants in a citywide citizen-operated sensor network are expected to have a publicly accessible Internet Protocol (IP) address, the network interface must establish secure communication between participating Connectors and ensure the reachability of each Connector for other network participants. Due to the absence of a central authority, all participating Connectors are equal, and as described Section 4.3, managing and discovering remote data sources is one of the open research questions this thesis tackles. Chapter 6 therefore focuses on the used network architecture and the utilized discovery mechanisms within the citywide data space.

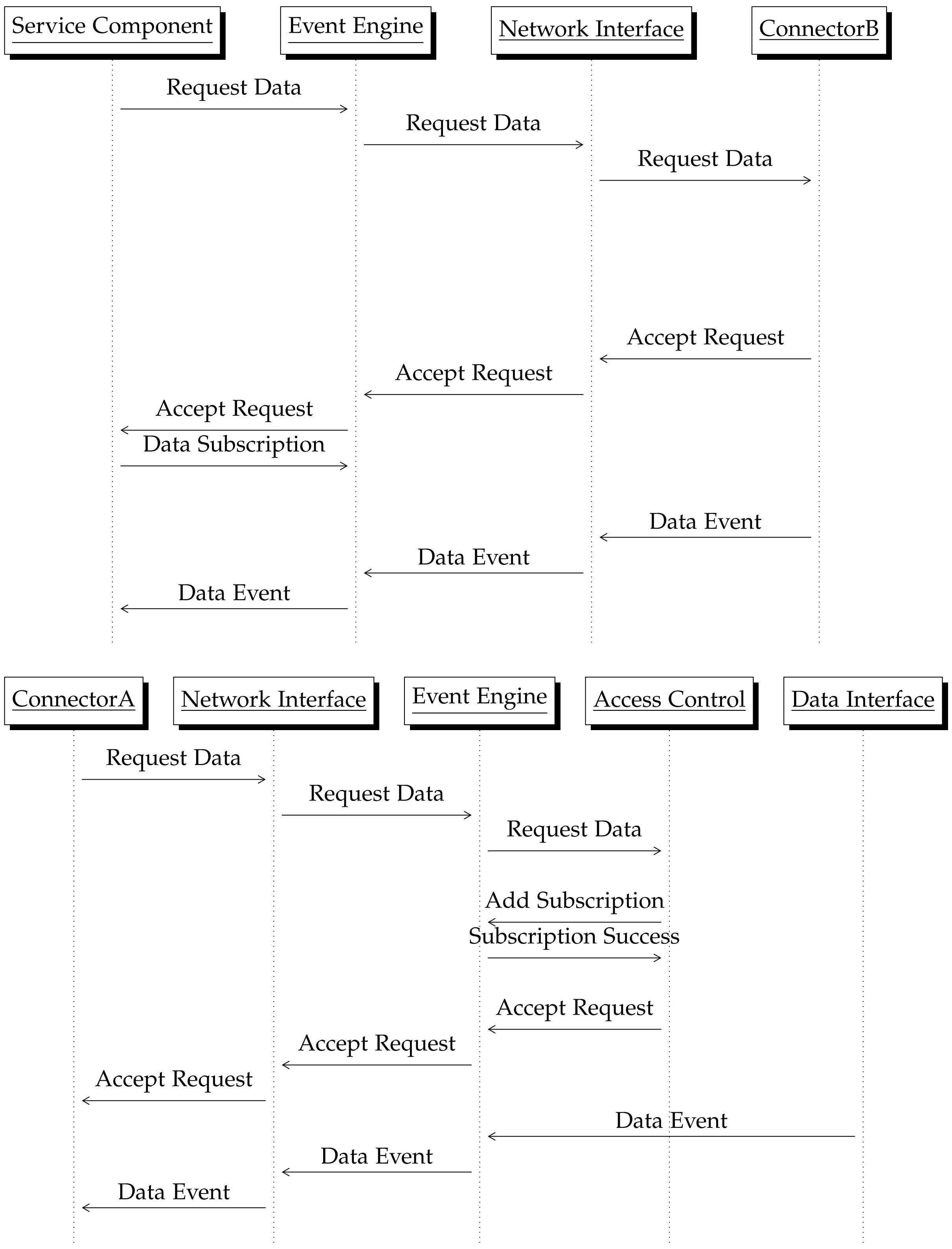


Figure 8: Communication between two participating Connectors to subscribe to a remote data stream. Where a service on ConnectorA (top) is interested in a data stream provided by the Data Interface on ConnectorB (bottom).

Service Interface

Finally, the service interface is introduced. This interface manages the integration of services with the citywide data space. Services are external dedicated software components that can interact with the data space via the service interface. These services can be public, providing functions to end-users through an external user interface or delivering results in the form of an API for other services and programs.

Each service has to provide two endpoints through which the Connector interacts with it. The Connector relays new data events to the service via the first interface. The Connector uses the second interface to transmit management events, which include the following:

ACCEPT REQUEST: When a new data source agrees to provide its data to the service, indicating that future data events from this source are to be expected.

DISCONNECT EVENT: This event informs the service that the network interface cannot currently connect to that data source, meaning no new events are expected for a certain period.

RECONNECT EVENT: When this event occurs, a connection to the data source has been reestablished, and data events are expected again. Depending on the reason for the disconnect, there may be a backlog of older events that are now forwarded to the service.

REMOVE EVENT: A data source has terminated the subscription and informs the service that no further data events follow.

How the service handles these events and the extent to which it implements fault tolerance is up to the service. The Connector acts solely as a manager of data sources and a mediator for the provided data.

When a service is initialized via the service interface, the developer must provide three pieces of information: First, the service and its function or goal must be described semantically. This description is provided to data sources to help them decide whether to provide their data to the service. The description includes both the processed data and the goal of the service. This can be the resulting benefit for its users in the form of a user application or the provided API to be used by other services and applications.

Discovering relevant data sources is done by the network interface. Chapter 6 focuses on an efficient discovery mechanism to locate all relevant data sources within a citywide data space.

Second, the required data must be described. The developer uses the citywide data model to define the data type needed. This includes, on the one hand, the type of required data and, on the other hand, restrictions on which data sources are relevant. For example, a service might only be interested in data about a certain area within the city or only if the same data source collects two different data types at the same location. With this description, the citywide data space can discover suitable data sources, request them, and forward their data to the service.

Finally, the developer must define whether and what kind of local preprocessing the data sources should perform. This preprocessing description is transmitted to the data sources along with the service description, allowing data owners to decide whether to provide their data for the described service in the specified preprocessed form. One of the major research gaps discussed in Section 4.3 is how preprocessing is realized. In Chapter 7, the proposed approach is described and evaluated in detail.

5.3 SYSTEM ARCHITECTURE FOR DECENTRALIZED DATA SPACES

After describing the individual Connector and its components, this section provides an overview of the resulting data space architecture and the interactions between participating Connectors. The data space is represented as six interdependent layers highlighting the Connectors' different functionalities. These layers range from the *sensor layer*, where raw data is collected, to the *application layer*, which enables applications to process, display, or extract information from the provided data. Figure 9 provides an overview of these layers and depicts the connections and transfer of information between them. The blue dots represent the participants' Connectors, with vertically aligned dots indicating the same Connector across different layers. Arrows illustrate, as an example, how data is transferred from sensors to services. The following sections describe each layer.

Sensor Layer

The sensor layer describes all data sources connected to the Connectors of the citywide data space. These can be hardware sensors or so-called virtual sensors. Virtual sensors represent any conceivable form of digital data (e.g., a news feed on a topic or from a person, stock market information, etc.). Each Connector can represent and manage any number of sensors. It is important to note the *one*-to-*many* relationship between a Connector and its linked data sources. How the data is transmitted from the data source to the Connector can vary depending on individual data sources, as described in Section 5.2.2.

Various standards already provide solutions for transmitting data from a data source to a data sink; therefore, this thesis does not address this topic further. It is assumed that the connection is established and continuous data is transmitted from the data sources to the Connectors.

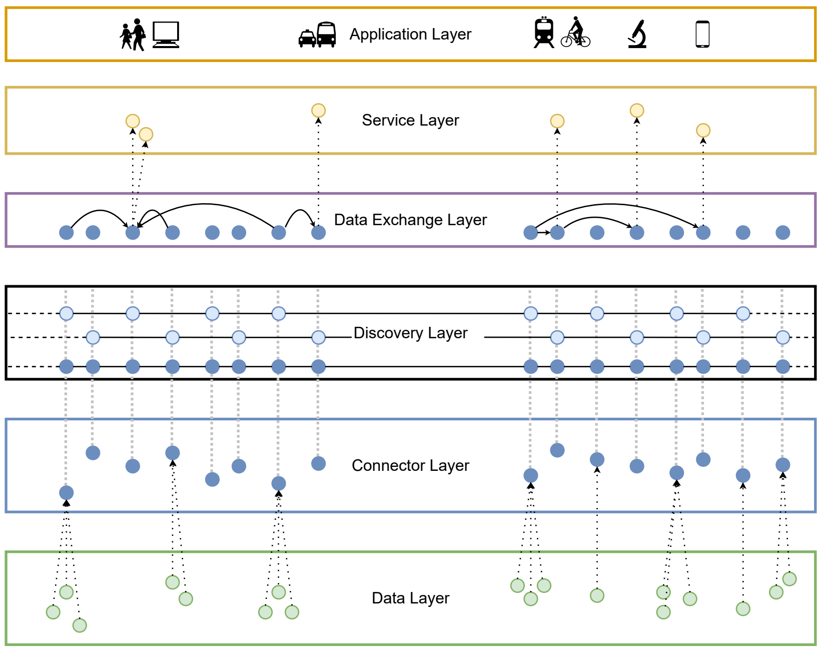


Figure 9: Abstract network architecture composed of six layers. Vertical blue dots represent the same Connector on different layers. Lines represent connections, and arrows show the data transfer between layers and Connectors.

Connector Layer

While caching and distributed storage can improve data availability, owners might lose control over which services can access their data.

The Connector layer represents the independent Connectors, each of which represents a locally isolated data silo. When considered individually, a participant can already use a Connector to connect local services and data sources. The Connector software serves as the core for all layers above it. As previously mentioned, a Connector can run on dedicated hardware or be embedded into existing infrastructure. For a stable and efficient data space, the Connector should ideally be continuously connected to the internet and accessible from it. Since no caching or distributed data storage is planned, the sensor data is only available to external services when an active internet connection exists.

Discovery Layer

The Connectors connect on the so-called discovery layer to allow provided services to find relevant data streams from other Connectors. Here, the Connectors form an overlay network that enables efficient search based on semantic descriptions. This search allows services to find groups of Connectors that provide relevant data streams efficiently. The semantic description can include one or more sensor types, the location of data collection, or the temporal resolution.

While the discovery layer is responsible for finding data streams, it only exchanges address information, not the actual data. The discovery layer and its specific functions are presented and evaluated in depth in Chapter 6.

Data Exchange Layer

After Connectors have exchanged their connection data on the discovery layer, data exchange occurs on the data exchange layer via a direct connection without using the overlay network. This way, other network participants do not know which and how many data packets are exchanged between the two Connectors. For example, the *Subscription Request*, already familiar from Figure 8, is sent and answered in this direct connection. The transfer of data streams is also carried out via a direct connection. This direct connection prevents third parties from collecting meta-information about the individual Connectors and their connections, which enhances both the security and privacy of the individual participants.

Service Layer

As described in Section 5.2.2, each Connector can manage any number of services and forward data to them. Each service defines in its service description which data is required and how data sources should locally preprocess their data before making it available to the service. During the initialization of the service, all relevant data sources are automatically discovered through the discovery layer, and the request, along with the service description, is transmitted via the data exchange layer. The owner of these data sources can then decide whether to make their data available to the service under the specified conditions. If they agree, all new incoming data from the point of agreement is preprocessed and sent via the data exchange layer to the responsible Connector, which then forwards this data through the service interface to the deployed service. One of the key innovations in this layer is the privacy-enhancing distributed preprocessing of raw data from distributed data sources. How this is implemented is elaborated in Chapter 7.

Interface Layer

The interface layer describes all user interfaces provided by the various services outside the network. Various use cases are conceivable: On the one hand, data can be visualized through different tools to give viewers a better overview of the city or specific applications; external hardware may be able to use data from the network through API accesses and make decisions based on the data; classic applications (such as routing services) can also be realized as services, allowing users to utilize the network's data through user interfaces (web or app applications) without having to understand the complex underlying structure and origin of the data.

5.4 A CITYWIDE DATA MODEL

This section examines using a citywide data model and the unified semantic description of sensors, data, and services. At first, the concept of ontologies, their functionality, and the advantages they can bring to citywide data spaces are introduced. This is followed by a brief analysis of existing ontologies, concluding with a selection of ontologies used for the proposed citywide data space.

5.4.1 *Advantages of Ontologies*

Ontologies are formal representations of a knowledge domain through a group of concepts and the relationships between these concepts. They serve to structure and standardize knowledge in a specific area systematically. An ontology typically includes:

CONCEPTS represent categories of objects in a particular knowledge domain.

INSTANCES are specific objects or events defined within concepts.

PROPERTIES describe characteristics or attributes of the concepts.

RELATIONS define the relationships between different concepts.

With the advent of the World Wide Web and the development of the Semantic Web by Tim Berners-Lee in the early 2000s, ontologies have played a central role in structuring and ensuring the interoperability of information on the Web. Standards such as Resource Description Framework (RDF) and OWL were developed to create and utilize ontologies on the web. Due to the clearly defined structure of an ontology and its components, they offer several advantages in heterogeneous knowledge environments. They establish a common vocabulary for various stakeholders, facilitating communication and understanding. Using standardized terms and structures, ontologies enable interoperability between different systems and data sources. They ease data integration from various sources by providing a common structure and meaning. Additionally, ontologies can be reused across different applications and domains, allowing for the automated derivation of new knowledge through logical reasoning based on defined relationships and rules. Furthermore, by adding application-specific concepts, properties, and relations, existing ontologies can be adapted and extended to various special use cases.

Many of the requirements outlined in Section 3.3 are evident in these advantages. Using a clearly defined ontology helps to make existing heterogeneous sensors in the city usable and interpretable. Interoperability is promoted through a unified semantic understanding, and by improving the extensibility, a citywide data space can adapt to new circumstances and meet new requirements.

5.4.2 *Existing Ontologies for the IoT and Data Spaces*

For a citywide data space based on a decentralized structure of sensors and services, various ontologies already exist that describe some of the most important concepts, instances, properties, and relations. The following ontologies are considered, which integrate and standardize a wide range of heterogeneous and extensive data:

SSN (SEMANTIC SENSOR NETWORK ONTOLOGY): Developed by the Word Wide Web Consortium (W3C), the Semantic Sensor Network (SSN) Ontology describes sensors, sensor observations, and measurements, as well as the related properties and units. This ontology provides a detailed framework for representing the capabilities and outputs of sensors in a standardized way [266].

SOSA (SENSOR, OBSERVATION, SAMPLE, AND ACTUATOR): To simplify SSN W3C create SOSA. It focuses on the core concepts of sensors and observations, providing a lightweight framework for representing these elements. The simplicity of SOSA makes it suitable for a wide range of applications where the full complexity of SSN is not needed [267].

IOT-LITE: IoT-Lite is a lightweight ontology designed for the IoT. It models IoT resources, data streams, devices, and services, providing a streamlined framework for representing the components and interactions within an IoT ecosystem. This ontology is particularly useful for applications requiring efficient and scalable data management [265].

QUDT (QUANTITIES, UNITS, DIMENSIONS, AND DATA TYPES): To model quantities, units, and data types, the QUDT ontology was defined. QUDT makes it simple to describe measurements and their units. By providing a standardized approach to representing these elements, QUDT facilitates integrating and comparing data from different sources [209].

TIME ONTOLOGY: Developed by the W3C, the Time Ontology models temporal concepts such as time points, time intervals, and events. This ontology provides a comprehensive framework for representing and reasoning about time-related information in various applications [268].

VCARD ONTOLOGY: The vCard Ontology provides an RDF representation of the vCard format. It models addresses and contact information, offering a standard way to represent and share personal and organizational contact details in semantic web applications [264].

WGS84 GEO POSITIONING ONTOLOGY: The WGS84 Geo Positioning Ontology defines fundamental classes and properties for describing geographical positions and locations. It is particularly useful for specifying latitude and longitude coordinates [263].

FOAF (FRIEND OF A FRIEND): FOAF models social networks, people, places, and organizations. It can be used to describe places and associated information, providing a framework for representing social and geographical relationships. This ontology is widely used in social networking and web applications to enhance interoperability and data sharing [78].

All of these ontologies are developed for a specific use case and build upon each other to avoid duplicate definitions for the most basic concepts and instances. A combination of these ontologies covers different aspects of the overarching architecture to create a citywide data space. Representing sensors, Connectors, services, and collected data, the SOSA ontology is a broadly accepted standard and is used; the additional concepts from the SSN are not needed but can easily be added as the architecture might change in the future. The QUDT ontology is used to describe collected data in detail and makes it comparable between different heterogeneous sensors. The vCard and WGS84 ontologies are used to describe the geological placement and distribution of data. While vCard contains address-based geolocation definitions such as city, borough, street, and number, WGS84 describes geographical coordination using latitude and longitude. vCard even contains more concepts that can be used to describe personal or organizational information, similar to the FOAF ontology, but these are not yet used.

5.4.3 *Exemplary representation of a Connector and Sensor*

With the help of the previously introduced and selected ontologies, the concepts, instances, and relations in a citywide sensor network can now be described. This allows data sources and services to be automatically compared and related even in a heterogeneous network. The following is an exemplary representation of a Connector that provides two sensors (data sources) and a service for calculating the hourly average temperature in its own district.

Initially, custom definitions with the prefix 'cwds' (citywide Data Space) are introduced. These application-specific definitions are not present in any of the used ontologies. Therefore, these definitions must be recognized in the citywide data space and known to each participating Connector. After defining these namespaces, the Connector defines its sensors and observed properties. In this example, the Connector provides two sensors: a temperature sensor that observes the surrounding temperature in degrees Celsius and a humidity sensor that observes the humidity in percentage. Following the sensors, the Connector is described with all hosted sensors and services, as well as the location that was set by the user when initializing this Connector. With this information, other services can now search for all sensors that observe Temperature within 10014 New York, and the data space can identify this Connector as a possible data source.

Listing 1: Semantic description of a Connector with two sensors and a provided service

```
@prefix sosa: <http://www.w3.org/ns/sosa/> .
@prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> .
@prefix vcard: <http://www.w3.org/2006/vcard/ns#> .
@prefix qudt: <http://qudt.org/schema/qudt/> .
@prefix unit: <http://qudt.org/vocab/unit/> .
@prefix cwds: <https://citywide-dataspace.org/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

# Definition of Sensors
cwds:tempSensor a sosa:Sensor ;
    sosa:observes cwds:Temperature ;
    sosa:hosts cwds:Connector .

cwds:humiditySensor a sosa:Sensor ;
    sosa:observes cwds:Humidity ;
    sosa:hosts cwds:Connector .

# Definition of observable properties with QUDT
cwds:Temperature a sosa:ObservableProperty, qudt:QuantityKind ;
    qudt:unit unit:DegreeCelsius ;
    rdfs:label "Temperature" .

cwds:Humidity a sosa:ObservableProperty, qudt:QuantityKind ;
    qudt:unit unit:Percent ;
    rdfs:label "Humidity" .

# Definition of the Platform
cwds:Connector a sosa:Platform ;
    sosa:hosts cwds:tempSensor, cwds:humiditySensor, cdws:
        avgTempService ;
    vcard:hasAddress cwds:address ;
    geo:location cwds:location .

# Definition of the Platform’s address
cwds:address a vcard:Address ;
    vcard:street-address "90 Bedford St" ;
    vcard:locality "New York" ;
    vcard:region "NY" ;
    vcard:postal-code "10014" ;
    vcard:country-name "USA" .

# Definition of the Platform’s geographic location with WGS84
cwds:location a geo:SpatialThing ;
    geo:lat "40.732013"^^xsd:float ;
    geo:long "-74.005688"^^xsd:float ;
    geo:alt "15"^^xsd:float .

# Definition of the service for calculating the hourly average
    temperature
cwds:avgTempService a sosa:Actuator ;
    sosa:actsOnProperty cwds:Temperature ;
    sosa:hostedBy cwds:Connector ;
    cwds:computes "HourlyAverageTemperature" .
```

5.5 SUMMARY

This chapter discusses the overarching architecture of a citywide decentralized data space. The Connector and its functionalities are introduced as the central component of the citizen-operated network. Each participant connects their locally managed data sources and services to the citywide data space through this Connector, enabling access to their own data and, in return, gaining access to data from other participants. The Connector acts as a gateway where the owner can set fine-grained access controls and track which services use their data for which purposes.

While this chapter provides an overview of the entire system’s functionality, the following chapters examine and evaluate individual aspects in detail. The most significant research gap identified is the decentralized networking and associated discovery mechanisms, which are presented in the subsequent chapter. In addition to discovery, the distributed preprocessing of raw data to enhance privacy and distribute load was identified as an essential research gap, which is analyzed in depth in Chapter 7. Due to the self-managed sensors and the lack of a controlling authority for standardized placement, data quality evaluation is an essential aspect of a citizen-operated data space. As justified in Section 5.2, data quality should be implemented as a separate service. This service is introduced and evaluated in Chapter 8.

6 DISCOVERY OF DATA SOURCES IN A CITYWIDE DATA SPACE

After presenting the citywide data space architecture based on Connectors deployed by individual participants in the previous chapter, this chapter analyzes the first identified research gap in greater detail. In Section 3.3, the discovery of groups of similar data sources was established as an important requirement for citywide data spaces. In Section 4.3, it was shown that previous approaches follow a centralized discovery approach. However, the instance managing this centralized discovery can analyze metadata about the relationships between other participants, potentially inferring data usage and access rights. Additionally, a centralized discovery infrastructure can become a bottleneck, especially in a constantly changing network with joining and leaving Connectors, which continuously burdens that single discovery service.

For these reasons, a decentralized discovery network is introduced, enabling services within the data space to search for groups of suitable data sources. The service describes the type of collected data and additional criteria, such as temporal resolution or spatial restriction. Without a central instance managing all the knowledge, the network the search request to all matching data sources, allowing them to respond to the searching service as potential data sources.

The following sections first introduce the fundamentals of decentralized overlay networks. This is followed by a differentiation from related works that propose solutions for searching in decentralized networks. After the differentiation, the network structure used is presented. Building on this network structure, the search algorithm and additional extensions for optimization are introduced. The proposed approach is evaluated and discussed at the end of the chapter. This chapter's provided ideas and figures have been previously published in [137].

6.1 DECENTRALIZED OVERLAY NETWORKS

Research and development of decentralized overlay networks, which play a fundamental role in designing distributed systems and networks, began in the early days of the internet and Peer-to-Peer (P2P) technologies. These structures are crucial for the efficient organization, storage, and discovery of data in networks without a central coordination unit. Decentralized data structures can be categorized based on their organizational form and underlying network design.

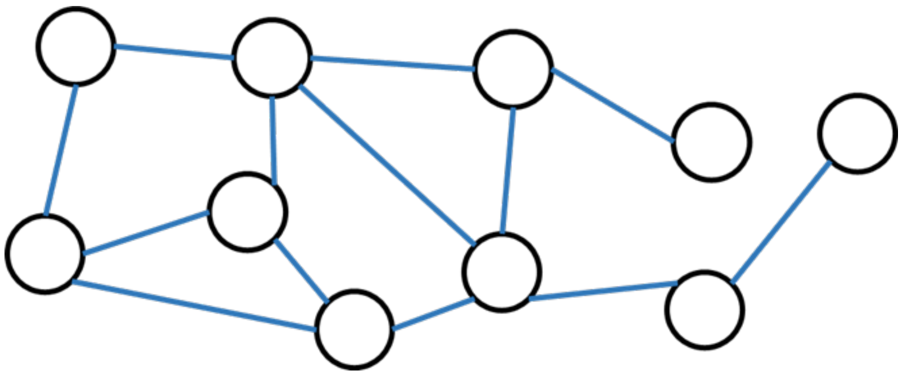


Figure 10: Exemplary unstructured overlay network containing nine peers and thirteen connections.

These categories include unstructured, hierarchical, and structured networks, each with unique characteristics and use cases.

6.1.1 *Unstructured Networks*

Unstructured networks, as illustrated in Figure 10, were among the first P2P networks to be developed. One of the most well-known examples is Gnutella [213]. Unstructured P2P networks are characterized by a random organization of peers, with files and resources distributed without a fixed schema. The lack of structure simplifies the organization of nodes and leads to easily scalable networks. However, a significant drawback is the search mechanism. Due to the absence of a structure determining where data is stored, searches in unstructured networks typically occur through flooding algorithms. These algorithms distribute search queries based on various heuristics [133, 275] to as many nodes as possible in the network to find the desired data quickly while attempting to minimize the number of peers queried to keep the network load low. As the network size increases, more participating peers must be queried, stored data must be duplicated more frequently, or searches are less likely to find the desired data. All these solutions can lead to inefficient searches in unstructured networks as the network grows [154]. Furthermore, it cannot be guaranteed that all sought-after information within the network has been found unless every peer in the network has been contacted. Due to the inefficient search mechanism and the lack of guaranteed search results, such unstructured networks are unsuitable for discovery in Smart City data spaces.

6.1.2 *Hierarchical and Hybrid Networks*

Hierarchical, or hybrid networks, represent an attempt to solve the scalability problems of unstructured networks by introducing a cer-

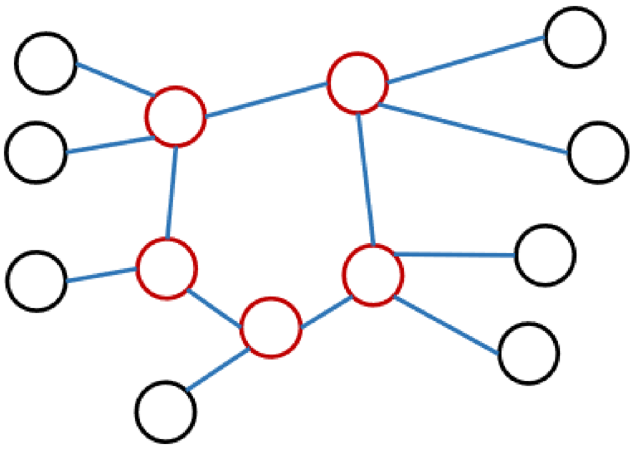
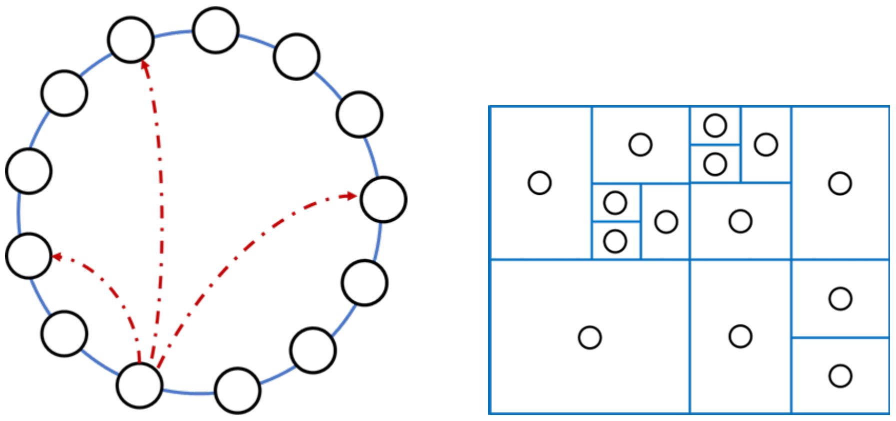


Figure 11: An example of a hierarchical network, where there is an inner circle with so-called superpeers (red) responsible for managing the peers connected to them.

tain form of organization. An example of such a network is illustrated in Figure 11. In this structure, peers always connect to a higher-ranking superpeer. Superpeers are responsible for indexing information and forwarding requests. This structure improves search efficiency since superpeers have more knowledge about data distribution, thus eliminating the need for flooding across all peers. However, this structure introduces new challenges regarding the load and reliability of superpeers. Superpeers possess additional information about the peers connected to them, requiring more storage space for data management, and the message traffic at these nodes is significantly higher than at the peers. If a superpeer fails, replacing it is a significant effort since a suitable successor must first be identified. This successor must then gather all the necessary information from the other superpeers and the connected peers to reach the same knowledge level as the previous superpeer. A high fluctuation rate is expected in a citizen-operated sensor network, which can negatively impact the network's efficiency due to the increasing overhead. Additionally, the Connectors are mostly built on consumer hardware, which affects the load capacity and may lead to more frequent failures under high load conditions.

6.1.3 *Structured Networks*

Structured networks offer a solution to the disadvantages of inefficient searching in unstructured networks by providing a deterministic structure for the cost of increasing the administrative burden. Additionally, all peers are identical in their roles within the network. To achieve this, established structured P2P networks like Chord [239], CAN [211], Pastry [217], and Tapestry [287] use a deterministic method for data organization and localization, enabling efficient and



(a) Ring-based structured overlay network, where the search efficiency and robustness are increased by using so-called finger tables, referring to peers further away.

(b) Structured overlay following a two-dimensional ordering, where each peer is responsible for a specific area in that two-dimensional space.

Figure 12: Two exemplary structured overlay networks, managing peers in different representations of an underlying namespace.

complete searches. This is done by assigning data to peers through an agreed-upon hash function used to hash peer and data identifiers allowing to determine data responsibility, hence the name Distributed Hash Table (DHT). One advantage of using a hash function is the even distribution of data and, consequently, the load within the network. However, a disadvantage of using a hash function for data organization is decoupling data from its source. By hashing the data identifiers, an independent peer within the network is determined as the data storage location. This is a desirable feature in a network that is responsible for distributed data storage and replication, as the distribution of data also distributes the load within the network. In a data space where participants want to manage their collected data (see Section 3.3), forced equal data distribution to other peers is not desired. Participants prefer to manage their collected data themselves without compromising search efficiency. This requirement led to the development of structured P2P networks that are not only based on a hash function but also a distributed skip list [206]. Two parallel approaches emerged: SkipGraph [19] and SkipNet [106]. In addition to the evenly distributed IDs of the hash function, these use a self-chosen identifier. This identifier allows peers in the network to be grouped and data to be localized. This has the advantage of *locality* of data and peers. Locality in this context means that peers with similar identifiers are close to each other in the network, and data can be stored on designated peers without reducing search efficiency. Furthermore, descriptive names for the peers can be chosen using self-chosen identifiers, allowing for semantic searches.

6.2 RELATED WORK

The efficient discovery of Connectors defined by a semantic description becomes more important as the number of semantic properties and the overall network size increase. Existing discovery approaches in distributed systems can be classified into two categories: Finding static data defined by unique identifiers and finding specific peers participating in the overlay network. In the first category, DHT-based approaches [211, 217, 239, 287] are used to improve load balancing and achieve faster lookup times through data caching and replication. Due to the volatility of generated data within the IoT context, these methods can only be used to a limited extent since caches and replicas have to be updated and replaced continuously. Similarly, searches in IoT are rarely performed for a single data point due to its dependence on other data and short lifetime. Instead, searches for categories of data from different IoT devices are to be expected. This indicates that full identifiers are most often unknown when starting a search. Instead, descriptive semantic properties are used to search for matching Connectors.

6.2.1 *DHT-based Networks*

More recent research focuses on discovering peers within DHT-based overlay networks based on one or multiple properties. S. Cirani et al. propose an architecture based on a DHT utilizing gateway nodes with a stable internet connection [55]. Sensors connect with a locally managed gateway node, which participates in a network based on a distributed location service [56] and distributed geographic table [199]. Nodes can be efficiently discovered based on their physical location. However, other properties (such as sensor type) cannot be used for the discovery.

Also based on a DHT, F. Paganelli and D. Parlanti propose a discovery service that can search for multi-attribute range queries [192]. To achieve this, the authors map multidimensional attributes into a one-dimensional namespace using space-filling curves. The resulting identifiers are then sorted using a Prefix Hash Tree (PHT). Finally, the PHT node identifiers are hashed and distributed over a DHT. This results in an equally distributed load across the network while enabling range queries performed over the PHT. In addition, multi-attribute searches are possible, but they have the drawback of potential false positives, which must be filtered out afterward. While this work allows for multi-attribute searches, false positives and the repetitive use of the underlying `get()` on the DHT leads to an unnecessarily high load of messages. Furthermore, it is impossible to select multiple specific attribute values that are not adjacent in the namespace (e.g., humidity and temperature sensors).

6.2.2 *Skiplist-based approaches*

In parallel approaches utilizing distributed skip lists [206] emerged. Harvey et al. [106] and Aspnes and Shah [19] independently introduced a distributed network architecture based on user-selected identifiers. These user-selected identifiers allow for more descriptive identifiers and group similar identifiers within the network. Further range queries based on these identifiers are supported to find all nodes within a network that share a common prefix in their identifiers.

Li et al. propose a locality-preserving context-aware service discovery called "LOCA" [148] based on distributed skip lists. The authors focus on reducing required message hops for queries within an organizational domain. Therefore, they introduced a more efficient naming scheme for services. While this effectively preserves locality and integrity within the specified organizational domains, an ontological model is required to receive all relevant services. While SkipNets supports range queries by default, Li et al. focus on ontology-driven discovery to find relevant services based on different criteria. A drawback of this approach is that each service must be contacted to evaluate whether all required criteria are met since the ontology is not embedded in the service's identifier.

Ishi et al. [114] introduced bounded range queries, allowing faster search results targeting a range of peers by splitting search messages into multiple subqueries. This approach has been further optimized by Banno and Shudo [22] to "reduce the average path length by roughly 30%". While these extensions enhance the efficiency of range queries, selections and ranges on multiple attributes are not supported.

However, none of the previously presented approaches can facilitate selection queries on multiple attributes. In the current state of the art, multiple overlays or search queries must be created to allow for multi-range queries. Which, in the end, further increases the overall message overhead. Due to SkipNets's locality-preserving property and its possibility of defining range queries, this data structure was selected as the foundation of the attribute-based approach.

6.3 OVERLAY NETWORK FOR CITYWIDE DATA SPACES

Organizing a large number of citizen-owned Connectors and their connected IoT data providers requires a scalable network overlay that allows services to discover available data providers and subscribe to their data streams. The overall architecture of such a decentralized system (Figure 13) is as follows: All data sources provided by private citizens (in this chapter referred to as publishers, based on the publish-subscribe architecture) generate continuous data streams. Without a centralized discovery manager, the network needs to or-

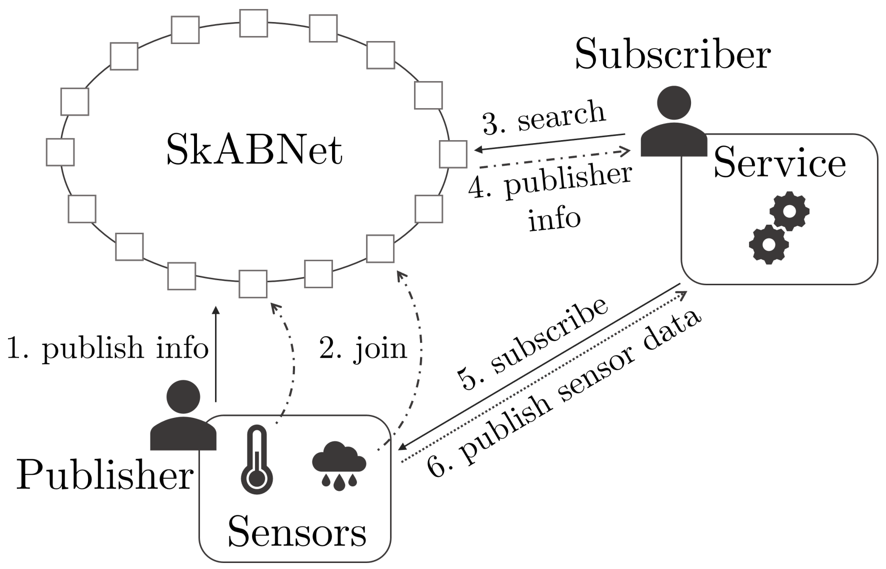


Figure 13: Discovery architecture of citizen-owned sensors based on a SkABNet.

ganize these data sources via a decentralized data structure. Hence, this thesis proposes Attributed-Based SkipNet (SkABNet). Each Connector participates in this discovery network by publishing information about provided data streams (1) and deploying dedicated *discovery nodes* in the network to represent described data streams (2). Therefore, not the resource-restricted IoT device (the sensor) itself joins the discovery network, instead a representation of this IoT device managed by the *Network Interface Component* of the citizen's Connector. Services that act as data subscribers are particularly interested in some of these data streams (e.g., all air pollution data in Manhattan) and search for the discovery nodes representing relevant data streams within this network. Services define a parameterized search query (3) that results in a set of publishers providing the requested data streams of interest (4). The exchange of the continuous data streams is not part of the SkABNet overlay network but can look like the following: Subscribers can now issue a subscription on discovered data sources (5), which forward data streams to the subscriber (6), who uses them to provide various services.

To organize these nodes in a decentralized data structure, SkABNet augments the architecture of SkipNet [106], a popular overlay network based on a distributed skip list [206]. Each node has two identifiers: a user-selected alphanumeric ID (*content ID*) and a randomly generated unique binary *numeric ID*. The remainder of this chapter refers to the SkipNet architecture as depicted in Figure 14.

Nodes are located on multiple double-linked rings, each level halving the set of nodes based on their numeric ID. Starting at level 0, a single ring contains all participating nodes. The following level com-

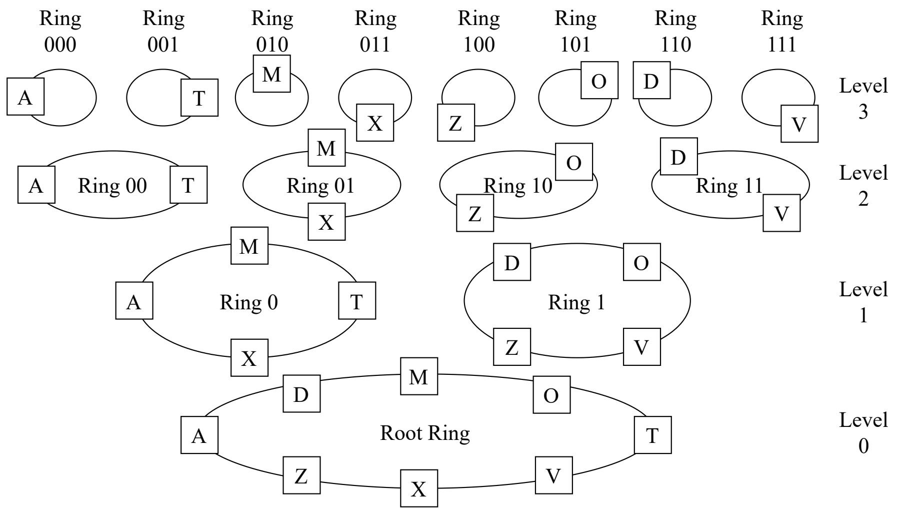


Figure 14: The SkipNet architecture arranges nodes onto rings, each halving the ring of the underlying layer [106].

prises two rings, each with 50% of the nodes; level two already has four rings with 25% of the nodes, respectively, and so forth. Each node stores its adjacent nodes (*neighbors*) from each of these rings in its neighbor tables. Due to the randomly generated numeric IDs, neighbors on higher ring levels have a large distance on the level-0 ring, allowing searches to skip over larger numbers of discovery nodes to find a searched discovery node in $O(\log n)$ hops where n is the total number of discovery nodes in the network.

The user-selected content IDs are then used to order discovery nodes within each ring. This ordering scheme ensures that discovery nodes with identical prefixes in their content ID are adjacent on all shared rings. Messages targeting discovery nodes with identical prefixes can be sent via discovery nodes containing this prefix only. Additionally, all discovery nodes with identical prefixes are found without leaving the prefix-based namespace.

To find discovery nodes within a SkipNet for a defined content ID, a participating discovery node starts the search at its highest-level ring to find another discovery node with a content ID located between its own and the target. If a closer discovery node is found, the search is forwarded until either the searched discovery node has been reached or if no closer node exists in the level-0 ring, meaning no node with the searched content ID exists.

$$\begin{aligned}&\mathit{Manhattan.Broadway\#657}\\ &\hookrightarrow \mathit{.Temperature.Sensor001} \qquad (1)\\ &Q :: \mathit{Manhattan.Broadway.} \qquad (2)\end{aligned}$$

Multiple discovery nodes can be searched using range queries on a common prefix within their content ID. Listing 1 shows an example content ID representing a data provider collecting temperature data at Broadway in Manhattan. Content IDs in this form allow for range queries matching all data providers in Manhattan located at Broadway, using the range queries shown in Listing 2. To find all matching discovery nodes, the search is forwarded to the first matching discovery node utilizing the content ID routing algorithm. The found discovery node finally forwards the search on the lowest level ring until all discovery nodes with the given prefix are found.

6.4 skabnet architecture

This chapter introduces SkABNet, a distributed data structure that builds upon the idea of SkipNet, and introduces **a**ttribute-**b**ased identifiers. This extension allows the expression of more complex ontology-based search queries and executes these queries more efficiently. This makes it more convenient for services to search for data streams defined by multiple attributes (referring to ontology properties) and reduces the total number of messages needed to find all matching data streams.

To achieve this, SkABNet extends the SkipNet architecture with three distinct enhancements: First, it provides *attribute-based identifiers*, allowing for more expressive identifiers by dividing them into a list of attribute-value pairs. Further, it utilizes the introduced attribute-value pairs to create more expressive search queries. By dividing identifiers into a list of attribute-value pairs, each attribute can be interpreted individually, thus allowing searches on single or combinations of attribute-value pairs. Second, it introduces four *attribute-based search operators* that enable ranges and selections on individual attributes. Lastly, to optimize for common search patterns, data providers can be represented by multiple *attribute compositions* reducing required messages to find matching discovery nodes within the overlay network.

6.4.1 *Attribute-Based Identifiers*

As mentioned before, SkABNet uses attribute-value pairs as identifiers. Attributes are based on the ontology properties used by citizens to describe their connected data sources, such as the sensor location, sensor type, or measuring interval. When citizens make their data streams available to the citywide data space, the *Network Interface Component* creates these attribute-value pairs based on the available semantic description of the sensors, making the data source identifiable for services in the network. The choice of attributes is similar for each publisher and subscriber in the network and is defined by the domain. The only requirement SkABNet has for attribute-based identifiers is that these need to be unique. Therefore, an attribute repre-

See the Connector architecture in Section 5.2.

senting a *UUID* should exist, ensuring that all participating discovery nodes have a unique identifier.

Having independent attribute-value pairs makes it possible to put these attributes in arbitrary order. In contrast to SkipNet, where the identifier is ultimately defined at node creation, SkABNet allows the definition of one or even multiple orders of attributes without requiring further input from the citizens. This order can significantly impact search efficiency, as discussed later in this chapter.

The idea of attribute-value pairs also benefits subscribers who search the network for relevant data streams. They do not have to define search queries that exactly match the identifiers as required in SkipNet. Instead, they can define search values for some attributes and omit those irrelevant to them. The latter is replaced by wildcards when the search query is executed.

Listing 3 shows the structure of a SkABNet identifier. The delimiter "/" separates attributes, values, and attribute-value pairs.

```
/Dist/Manhattan/St/Broadway/No/657/
↪ Type/Temperature/UUID/Sensor001                    (3)
```

6.4.2 *Attribute-Based Search Queries*

Services acting as subscribers use attribute-based search queries in SkABNet to look up relevant data streams. Therefore, when service developers initialize a service, they semantically describe required data (see Figure 5.2.2). Similar to the creation of attribute-value pairs for data sources, the *Network Interface Component* can use this semantic description to create a search based on the properties of the required data. When a data source's attribute values match this search query, it is considered relevant, and its information is sent to the searching service.

Search queries can have different complexities. In the most basic scenario, the Network Interface inserts exactly one value for each attribute. This results in a straightforward search looking for, e.g., the temperature sensor at a particular place, which can be identified by the combination of a city, a district, a street name, and a house number. This query type is also standard in SkipNet whenever the fully qualified identifier is used.

Continuing the example above, the subscriber may also want to retrieve data from the humidity sensor in this place. While this would require two distinct searches in SkipNet, both using the entire identifier, SkABNet accepts **selections** as an input. In the example, the subscriber could insert both values, *temperature* and *humidity*, in the *type* attribute. SkABNet interprets this selection and performs a search for nodes matching either sensor type. Selections can include not only two but any number of values.

Operator	Description	Use-Cases	Possible in SkipNet?
Single Value	Finds nodes where the exact value (*Manhattan*) matches for the district attribute. **Example:** /dist/*Manhattan*	This operator is the most basic and is used to find only specific data providers with the given value .	Yes. This is the standard search within SkipNet.
Selection |	Finds nodes with values specified in the selection. Nodes located between these values are skipped. **Example:** /St/1stAve|Broadway|ParkAve	Most commonly used for alpha numeric values where several values are of interest.	Partially. Multiple searches have to be performed, one for each element in the selection.
Range ~	Specifies a range of values with inclusive boundaries. Finds all values that are alphanumerically ordered between the upper and lower bound. **Example:** /No/003 ~ 065	Most commonly used for numeric values. Numbers are needed to be padded with leading zeros so that they are correctly ordered alphanumerically.	Restricted. Only possible if all values within the range are known or by implementing the extension discussed in [22, 114].
Wildcard *	Matches nodes regardless of the value for this attribute. **Example:** /Type/*	Used when attribute is irrelevant for the search.	Restricted. Only possible if the wildcard attribute is at the last position or all values for these attributes are known.

Table 5: Search operators in SkABNet. The additional operators (selection, range, wildcard) can be used to define complex queries intuitively. SkipNet, in contrast, requires manual effort to mimic these queries with multiple single-valued queries.

Selections are helpful when more than a single value is relevant. However, selections may be tedious and error-prone with a larger number of values. A second drawback of selections is that each value needs to be known in advance. What may be realistic for districts or streets, such as floating point values such as longitude or latitude, can typically not be provided as a list. For these reasons, SkABNet provides the **range** operator, which accepts a lower and an upper bound and matches every value in between. Thus, subscribers can define large search spaces with minimal additional input, even for floating point numbers. In contrast, SkipNet does not process bounded-range queries. Instead, a new search has to be started for every value within the range. This not only means that the number of individual searches can become quite large but also requires that all values must be discrete and known in advance.

There are many scenarios where some attributes are of no interest to the subscriber. When looking for, e.g., the entire set of temperature sensors in London, there is no need to indicate which streets and house numbers are relevant. Instead, all of them should be included in the query. For that, SkABNet provides a **wildcard** operator, which can be used as a placeholder for the whole value set of an attribute. In SkipNet, wildcards are challenging to implement. If the last value in the identifier is irrelevant, the identifier's stub (without the final value) can be used for a range query. Wildcards in any other part of the identifier result in a single search for each attribute value and would only work for discrete values known in advance (similar to bounded-range queries).

These additional operators summarized in Table 5 allow for greater flexibility when defining search queries. This advantage becomes even more apparent when multiple operators are combined, as in the following example. Listing 4 finds all publishers providing temperature and humidity within *dis1* in the *ExampleStreet* with house numbers ranging between 15 and 25. By considering attribute-value pairs individually, range and selection operators can be combined in a single search.

```
Q :: /Dist/dis1/St/ExampleStreet/No/15 ~ 25
↪ /Type/Humidity|Temperature/UUID/*                    (4)
```

To achieve a similar result in a standard SkipNet, an individual search has to be performed for each combination of house number and sensor type, leading to a total of twenty individual searches. In addition, results must be merged after all searches are completed.

Q :: /Dist/dis1/St/ExampleStreet/No/*15*
↪ /Type/*Humidity*/
Q :: /Dist/dis1/St/ExampleStreet/No/*15*
↪ /Type/*Temperature*/
Q :: /Dist/dis1/St/ExampleStreet/No/*16*
↪ /Type/*Humidity*/
Q :: /Dist/dis1/St/ExampleStreet/No/*16*
↪ /Type/*Temperature*/
...

6.4.3 *Efficient Publisher Discovery*

Next, the efficient discovery of publishers in SkABNet is discussed. The position of a discovery node representing a citizen-owned data stream is determined by its identifier, which consists of alphanumeric attribute-value pairs. In the lowest-level ring, all nodes are present and sorted in ascending order (compare Figure 14). The standard SkipNet can efficiently discover a single and a range of nodes, as long as these share a common prefix as described in Section 6.3.

SkABNet allows more complex search queries with selections, ranges, and wildcards on each attribute. As a result, matching discovery nodes may be scattered across the lowest-level ring. To find them efficiently anyway, SkABNet implements a new search algorithm that dynamically splits a search into multiple sub-searches, each focusing on different parts of the lowest-level ring with possible matching nodes. These sub-searches are forwarded in parallel, ensuring that no discovery node receives the query twice. This algorithm is discussed in the following. Further, this section demonstrates how compositions of SkABNet identifiers, i.e., the choice and order of attribute-value pairs, impact search efficiency.

Search Algorithm

SkABNet's attribute-based search is divided into two stages. Within the first stage, the search message is forwarded to the first node that matches the search query. This part is identical to the search algorithm in SkipNet. The first matching node can directly be calculated using the left values of selections and ranges. Once this node is found, the second stage begins, and the search splits up into branches that search the network in parallel. This is demonstrated with the example query from above (see Listing 4). For easier readability, attribute names are shortened, and single values are omitted. The search query now looks like this:

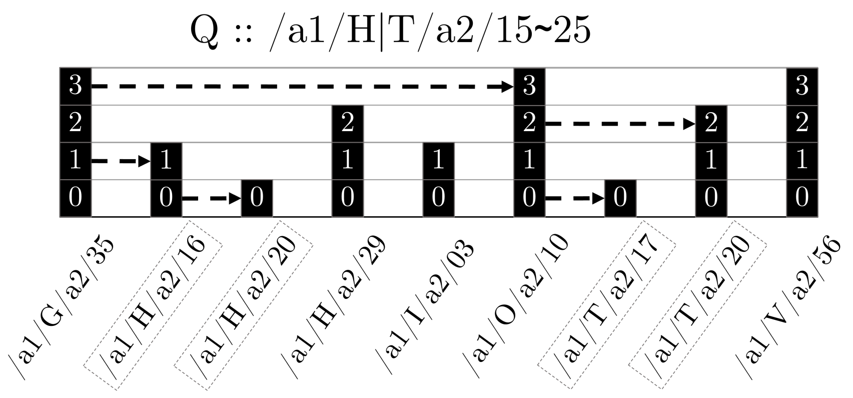


Figure 15: Nodes forwarding a search to neighbors on different ring levels to find all matching nodes for the given search.

$$Q :: /a1/H|T/a2/15 \sim 25$$

Attribute *a1* represents the sensor type, which can either be a *H*umidity sensor or a *T*emperature sensor. Attribute *a2* represents the house number, where values between 15 and 25 are relevant for the search.

The first relevant node, therefore, has the following identifier:

$$/a1/H/a2/15$$

To find this node, the search algorithm starts on one of the discovery nodes managed by the same Network Interface as the service and then checks its neighbors on each ring. Following SkipNet's search algorithm, skipping irrelevant discovery nodes to find the first matching one within $O(\log n)$ steps. If no node exists matching this identifier, the last node that forwarded the search starts the second stage. Here, the search is split up.

Figure 15 illustrates the second stage of the search algorithm. Each bar represents a node. The rows indicate the neighbor relationships between the nodes. Node /a1/G/a2/35, for example, is neighbor to node /a1/H/a2/16 on ring levels 0 and 1, to node /a1/H/a2/29 on level 2, and to node /a1/O/a2/10 on level 3. Arrows indicate to which neighboring nodes a search message is forwarded.

At some point, the search reached the node with the identifier /a1/G/a2/35, which detects that no node with the minimal identifier (/a1/H/a2/15) exists. Therefore, it starts the second stage and splits up the search message to find all matching nodes in parallel. /a1/G/a2/35 forwards a sub-search to /a1/H/a2/16 that matches the search. The upper boundary is set to /a1/H/a2/29 since this is the neighboring node on the next higher ring. Setting the upper boundary correctly is critical to assert that no nodes receive the same query twice. /a1/H/a2/29 receives no search message since no relevant nodes can be located between itself and the next neighbor /a1/O/a2/10. However, a search message is forwarded to node /a1/O/a2/10 even if it does not match the search since it is the neighbor on the highest ring level with possible matches behind it.

Node /a1/H/a2/16 receives the search message and forwards it to its matching neighbor /a1/H/a2/20. The message is not forwarded to /a1/H/a2/29 as this node is the upper boundary in the received sub-search.

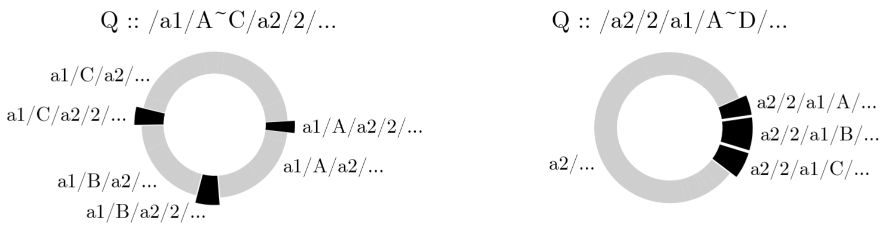
Parallel to that, node /a1/O/a2/10 receives the message and forwards the search to its neighboring nodes. The neighbor on the lowest-level ring with the identifier /a1/T/a2/17 matches the search and receives a sub-search with the upper boundary /a1/T/a2/20 since this is the neighbor on the next higher ring level. The search is also forwarded to the node /a1/T/a2/20 since it matches the search. /a1/V/a2/56 receives no message since it is located behind the last possible matching node.

/a1/T/a2/17 receives the search from /a1/O/a2/10 and does not forward it to /a1/T/a2/20 even though it matches the search since its identifier was set as the upper boundary in the search received. This ensures that /a1/T/a2/20 does not receive search messages multiple times. /a1/T/a2/20 receives the search message from /a1/O/a2/10 and has no other neighboring node matching the search; therefore, all matching publishers represented by these discovery nodes are found.

Algorithm 1 Forwarding searches to multiple neighbors while no neighbor receives duplicate messages

```
1:  procedure forwardSearch(Search, NeighborTable)
2:      for $i \leftarrow 0, Search.MaxRing$ do
3:          neighbor ← NeighborTable[i]
4:          if $neighbor > Search.UpperBound$ then
5:              return No more neighbors
6:          end if
7:          nextNeighbor ← NeighborTable[i+1]
8:          if Search.matches(neighbor) or
9:             matchBetween(neighbor, nextNeighbor) then
10:             $cSearch \leftarrow$ Search
11:             $cSearch.UpperBound \leftarrow nextNeighbor$
12:             $cSearch.MaxRing \leftarrow i$
13:             sendSearch(cSearch, neighbor).
14:         end if
15:     end for
16: end procedure
```

Algorithm 1 shows how the search message is forwarded in the second stage of the algorithm. The upper boundary is initialized with



(a) Attribute *a1* prefixing *a2*. (b) Attribute *a2* prefixing *a1*.

Figure 16: Distribution of matching nodes (highlighted in black) within two SkABNets with different ordering of attributes in their contentID for a search targeting a range of A ~ C for *a1* and a value of 2 for *a2*.

the last matching node which is determined by using the right values of its selections and ranges. Each traversed node is checking for relevant neighboring nodes until the upper boundary is reached (line 5). The search is also forwarded to irrelevant nodes, as long as there are possible relevant ones between a node itself and its neighbors on the next higher ring (line 8).

Attribute Composition

Following this search algorithm, each relevant publisher can be found in $O(\log n)$ network hops. The overall number of messages needed to find all relevant publishers depends on their distribution within SkABNet. As this location is defined by the attribute ordering in their identifier, searches targeting different attributes perform differently. For example, searches specifying attributes located at the beginning of the identifier have to search smaller parts of the SkABNet than searches with wildcards or ranges within attributes at the beginning.

Figure 16 shows the same SkABNet with different placement of publishers due to the order of the attributes in the identifier. In Figure 16a, publishers with the same attribute *a1* value are located next to each other. This makes searches for a single value in attribute *a1* and ranges or selections in *a2* efficient. On the contrary, a range over values for *a1* leads to relevant publishers scattered across the SkABNet as shown in Figure 16a. Using a different order of attributes, as displayed in Figure 16b, results in the relevant nodes being neighbors in the SkABNet. However, within this distribution of publishers, searches that target a specific value for *a1* and a range or selection of values for *a2*, matching publishers would be scattered again. Therefore, *attribute compositions* should be determined by the most common search pattern to group matching publishers and make searches more efficient.

To optimize for different search queries, a SkABNet starts multiple representations of publishers via so-called *virtual nodes*. The concept of virtual nodes was already introduced in the standard SkipNet as an optional enhancement [106]. Harvey et al. state that virtual nodes have size-reduced numeric identifiers to reduce the size of neighbor tables and, therefore, the required memory per node. However, this would lead to larger low-level rings and decrease search efficiency. Virtual nodes in SkABNet have their own content and a full-sized numeric identifier, allowing for different orders of attributes. This optimizes the network for multiple common search queries without decreasing search efficiency on lower-level rings.

Utilizing different compositions of attributes leads to another challenge: the number of possible compositions increases factorially with the number of attributes. Therefore, only a small set of attribute compositions should be started. Appropriate attribute compositions depend highly on the context and should represent the most common search queries. The number of compositions data providers can start depends on the hardware used in the context since managing neighbor tables within the SkABNet scales about linearly with the number of compositions. Furthermore, an adequate balance between an efficient discovery and the overall size of the SkABNet must be found.

In the previous example, attributes are: *district*, *street*, *house number*, *sensor type*, and *UUID*. In addition, a *latitude* and *longitude* representation would also be helpful to receive regional data independently from streets or districts. Using any combination of these seven attributes would already result in 5040 (7!) virtual nodes per publisher. This number can be considerably decreased by defining specific attribute compositions depending on the IoT context. Often, there are attribute pairs that are always searched in conjunction (e.g., searching by street name and house numbers). Also, some attributes are unlikely to occur together, such as a postal address and the representation of a location's latitude and longitude.

For named attributes such as sensor type, street, and district, single values, and selections are usually assumed, so they are set at the beginning of the identifiers. To avoid searches with a wildcard at the beginning of an identifier, since these result in scattered matching nodes and are costly hop-wise, two more compositions are added that either add the type at the end or omit the district. Attributes that most likely represent numeric values, such as the house number, are added next since ranges are costly hop-wise and are, therefore, preferably used on already limited name groups. The UUID is already unique and most likely unknown to subscribers. Therefore, in most cases, it is replaced by a wildcard, so it is appended at the end to fulfill the requirement of unique identifiers. For the case that specific publishers are searched for, an additional composition is added only using the UUID and the sensor type.

/Type/District/Street/HouseNumber/UUID
/District/Street/HouseNumber/Type/UUID
/Type/Street/HouseNumber/UUID
/Type/Latitude/Longitude/UUID
/Type/Longitude/Latitude/UUID
/UUID/Type

Having both orderings for latitude and longitude makes regional searches in the shape of rectangles either aligned with the latitude or longitude more efficient. Searches covering a wider range of latitude values, for example, identifiers with the longitude attribute before the latitude is chosen since matching nodes are located closer together within the SkABNet due to the smaller range of longitude values.

With these compositions defined, subscribers can search for publishers with specified attributes. SkABNet orders attributes searched according to the most efficient attribute composition and, therefore, only searches a fraction of the overall SkABNet.

6.5 EVALUATION

The previous section demonstrated that SkABNet provides more complex searches than SkipNet. It has further motivated that SkABNet searches are more efficient than comparable searches in SkipNet, leading to the same result. A quantitative analysis of the efficiency of both data structures is performed within this evaluation. Therefore, a set of 14 search queries, executed on networks of different sizes, is first evaluated. Second, a numeric example that shows the effect of different attribute compositions, i.e., what happens when the order of the attribute-value pairs in the SkABNet identifier is changed, is provided.

6.5.1 *Evaluation Setup*

To evaluate the SkABNet architecture, a simulation in *Omnet++* [258] was implemented to generate SkABNets of various sizes and run arbitrary search queries. SkipNets' search algorithm was implemented to benchmark SkABNet's attribute-based search. To quantify the efficiency of performed searches, the total number of search-related messages transmitted between discovery nodes in the network (*message complexity*) are compared.

For the simulations, the examples from before are used. Publishers describe their sensors with the following semantic properties: *District*, *Street*, *HouseNumber*, *Type*, and a *UUID*. Each publisher represents

Search Name	**Example Searches** Searching for ...	**#SkipNet Searches**
S1 Single Value	... a specific publisher participating in the network.	1
S2.1 Selection at the end	... publishers in a given district and street providing temperature and humidity data.	2
S2.2 Selection in the middle	... publishers in a given district located at 4 different streets providing temperature data.	4
S2.3 Selection at the start	... publishers located on a street passing through 5 districts collecting temperature data.	5
S2.4 Multiple Selections	... publishers located on 3 different street passing through 3 districts collecting temperature or humidity data.	18
S3.1 Range at the end	... publishers located at a street segment defined by a range of house numbers collecting temperature data.	19
S3.2 Range in the middle	... publishers located within an area defined by a range of streets providing temperature data.	15
S3.3 Range at the start	... publishers located on a range of districts providing temperature data.	7
S3.4 Multiple Ranges	... publishers located on a street segment defined by a range of house numbers passing through a range of districts providing temperature data.	57
S4.1 Wildcard at the end	... publishers of any data located at a specified street within a district.	1
S4.2 Wildcard in the middle	... publishers located on any street in a district providing temperature data.	40
S4.3 Wildcard at the start	... publishers located at a street traversing any district providing temperature data.	10
S5.1 Selection and range	... publishers providing temperature, humidity and air quality data located on a street segment defined by a range of house numbers.	48
S5.2 Selection, range & wildcard	... publishers of any data located on a street segment defined by a range of house numbers traversing two districts.	34

Table 6: List of searches issued on simulated SkABNets. These searches can be expressed with a single SkABNet search or several SkipNet searches indicated by the last column.

a sensor that publishes a data stream. Sensor types are randomly chosen from a set of ten possible data types. Further, sensors are randomly located in a fictive city with ten districts and 40 streets, each containing 200 house numbers. On initialization, sensors are assigned random values for these attributes, distributing them uniformly across the city.

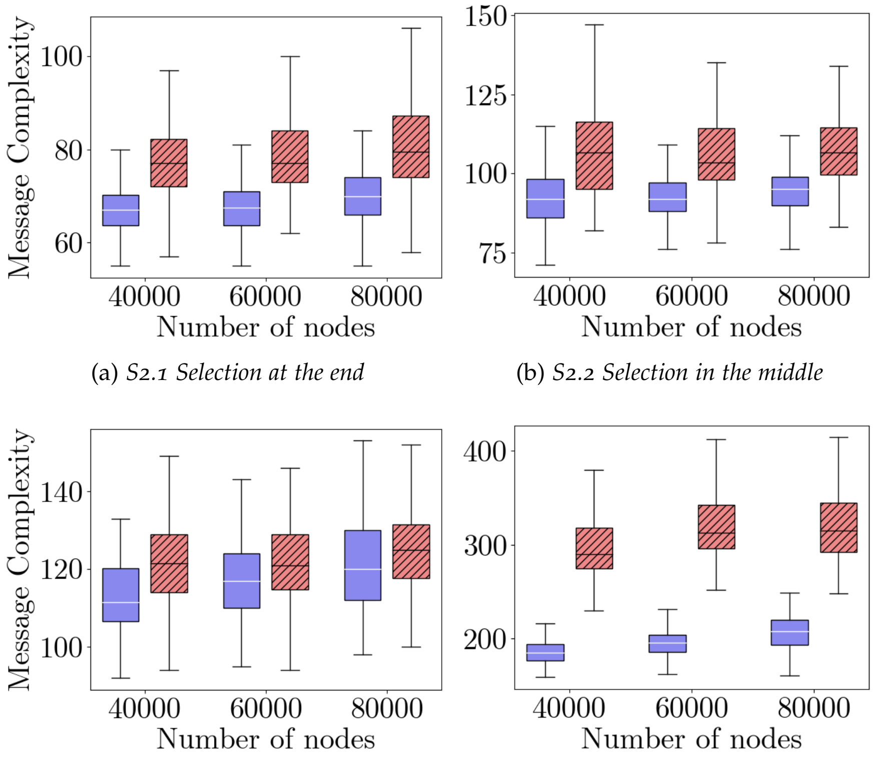
6.5.2 *Search Efficiency*

To evaluate SkABNet's search efficiency, 14 search queries containing operators such as selections, ranges, and wildcards in different positions are created. A set of corresponding queries that are required to get the same results in the SkipNet are created as a baseline. Table 6 provides an overview of these queries. The column *#SkipNet searches* indicates how many SkipNet queries are required to find the same publishers. It is to be noted that the translation from SkABNet searches into multiple SkipNet queries for ranges and wildcards only works in this case since all possible values for the attributes are known prior. Other attributes, such as the floating point values *latitude* and *longitude* as representations for a location, could not be represented by a basic SkipNet search due to the infinite number of values these attributes can represent.

S1 uses single values for all attributes and, therefore, finds a single publisher only. All other searches find exactly 50 publishers. This makes it easier to compare the efficiency of searches across all three network sizes (40 000, 60 000, 80 000 nodes). 100 networks are randomly created for each network size, and the 14 SkABNet searches and the equivalent 14 sets of SkipNet searches are performed. The message complexity represents the total number of search-related messages transmitted for the single SkABNet search (blue bar in the following figures) and the set of SkipNet searches (red hatched bar).

Table 6 shows that the number of SkipNet searches required to match a single SkABNet query varies significantly and depends on how many matching nodes are scattered across a network. It also stands out that for *S1* and *S4.1*, only one SkipNet search is necessary. Therefore, these searches behave exactly the same.

Figure 17 shows the effect of one or multiple selection operators in a search query. A selection at the end of a query requires fewer messages than a selection in the middle or the beginning. The effects add up for combinations of multiple selections, and even more messages are exchanged. The individual plots in Figure 17 also illustrate that SkABNet consistently uses fewer messages than SkipNet. The savings range from about 14% for selections at the beginning of the query to 7% when the selection is at the end. SkABNet even requires about 38% less messages for multiple selections than SkipNet.



(a) *S2.1 Selection at the end*

(b) *S2.2 Selection in the middle*

(c) *S2.3 Selection at the start*

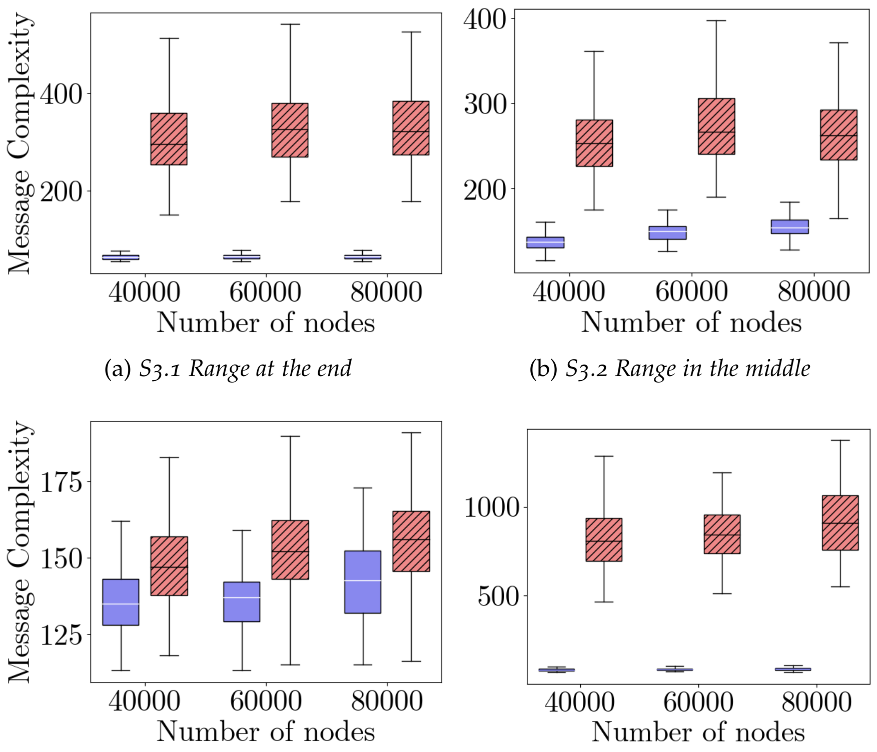
(d) *S2.4 Multiple Selections*

Figure 17: Searches utilizing the selection operator at different positions within a search query. *Blue bar: single SkABNet search; Red bar: Equivalent SkipNet searches*

Similar results can be observed for the range operator in Figure 18. Here, the effect is even greater. While SkipNet needs to perform many searches, SkABNet benefits from parallelizing the search process. On average, about 75% of all messages can be saved with a single range operator in the search query. Using multiple ranges, this number increases to roughly 90%.

Finally, Figure 19 shows the effect of wildcards (S4.2 and S4.3) as well as combinations of multiple operators (S5.1 and S5.2). Again, SkABNet's enhanced search algorithm results in significantly fewer messages than SkipNet, saving up to 52% using wildcards and 88% when all search operators are combined. Despite these promising numbers, the absolute values should not be emphasized too much here as they depend highly on how the network is created, how the parameter combination is chosen, and which searches are executed. However, the results show that, in general, large savings can be expected when using SkABNet.

Overall, the results illustrate that the efficiency of SkABNet searches depends mainly on two parameters. First, the number of replaced values defines the amount of basic SkipNet searches needed to achieve the same results. These differences can easily be seen by



(a) *S3.1 Range at the end* (b) *S3.2 Range in the middle*

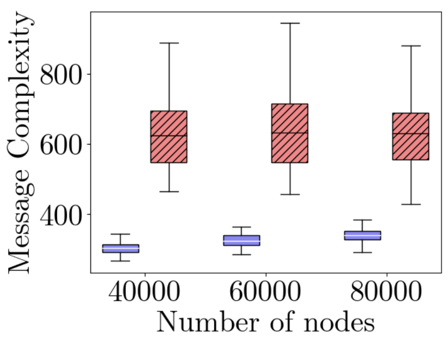
(c) *S3.3 Range at the start* (d) *S3.4 Multiple Ranges*

Figure 18: Searches utilizing the range operator at different positions within a search query.*Blue bar: single SkABNet search; Red bar: Equivalent SkipNet searches*

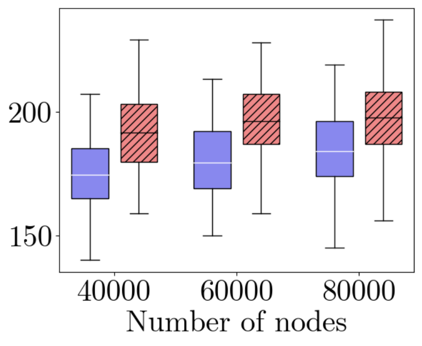
searches utilizing range and wildcard operators. Selection operators often replace only a small number of known values. Therefore, these searches can be expressed with a smaller set of SkipNet searches. Second, Figure 17c, 18c, and 19b confirm that search operators at the beginning of a search query reduce the advantage of SkABNet over SkipNet. This can be explained by the distribution of matching nodes within the network. Suppose a search ranges over multiple values within the first attribute. In that case, matching nodes are scattered over the network as shown in Figure 16a, leading to more messages needed to reach matching nodes. Nonetheless, worst-case scenarios lead to the same message complexity as in the standard SkipNet.

6.5.3 *Attribute Composition*

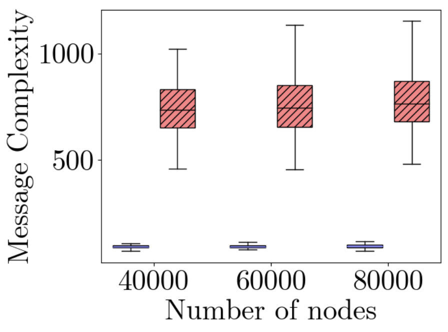
In Section 6.4.3, it was discussed how re-arranging attributes in the identifiers impact search efficiency. The last part of this evaluation provides a quantitative example to demonstrate this effect. Therefore, three search queries have been created, each benefiting from a different composition of attributes and matching 50 publishers. Then, three different SkABNets are started, each containing 50 000 publishers.



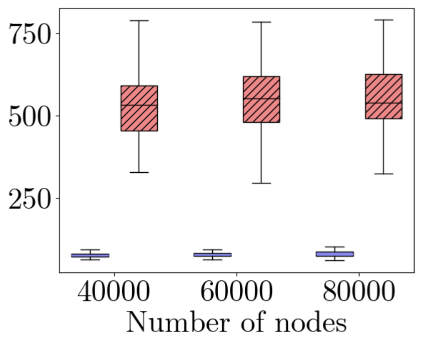
(a) *S4.2 Wildcard in the middle*



(b) *S4.3 Wildcard at the start*



(c) *S5.1 Combination of selection and range*



(d) *S5.2 Combination of selection, range & wildcard*

Figure 19: Searches utilizing the wildcard operator at different positions and searches composed of different search operators.*Blue bar: single SkABNet search; Red bar: Equivalent SkipNet searches*

The first provided a single attribute composition that was only beneficial for the first search. The second SkABNet added an additional composition, doubling the network size but having beneficial compositions for the first and second searches. Lastly, a third SkABNet defines an additional composition so that each search query has the most beneficial attribute order. As before, experiments are repeated 100 times.

Figure 20 shows how many search-related messages are transmitted for each individual search within the SkABNet containing one, two, or three compositions (x-axis) to find the 50 relevant nodes. When only a single composition is started, Search #1 performs well, while the other two suffer from the relevant nodes being scattered across the network (hatched boxes). Within the second SkABNet with two attribute compositions, Search #2 performs well, too. Finally, within the SkABNet, which provides three different compositions and therefore contains 150 000 nodes, all three search queries perform well, and the total number of search-related messages across all three searches can be reduced by 50%.

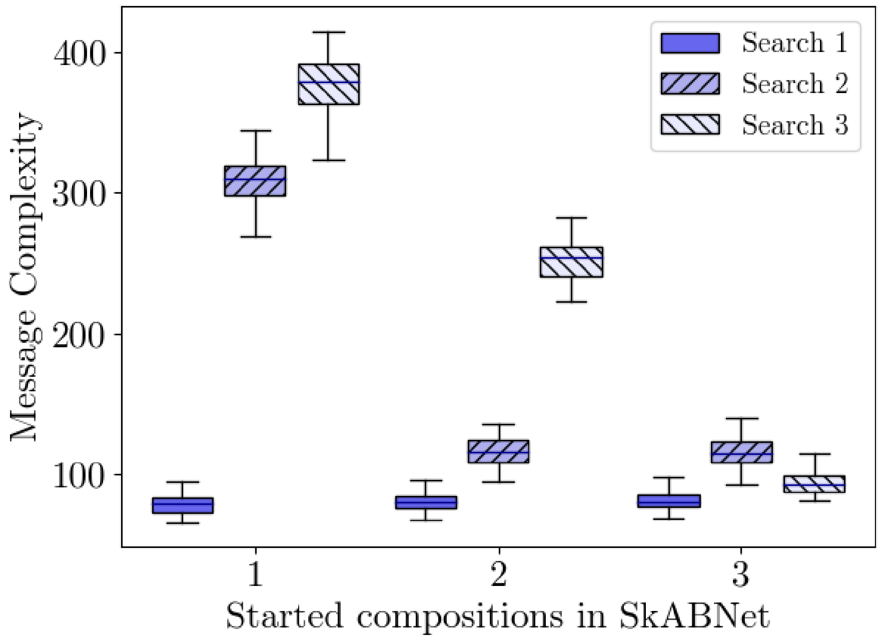


Figure 20: Message complexity for the three search queries within three SkABNet providing different attribute compositions.

Even though this scenario has been tailored to this particular use case, it shows the relevance of selecting "good" attribute compositions. The question of a good composition needs to be answered individually for each context. Results also suggest that starting multiple compositions can improve search efficiency significantly at linear cost for each node.

6.6 SUMMARY

This chapter presented SkABNet as a distributed discovery network based on SkipNets [106] and specifically introduced attribute-based identifiers for a citywide data space to increase the efficiency of search query executions. For that, SkABNet introduces four search operators that allow for more complex attribute-based search queries and enable services to discover relevant data streams by better expressing their required data. A related qualitative evaluation demonstrated that utilizing these complex search queries can reduce the number of messages needed to find all matching discovery nodes by up to 90% compared to SkipNet. To further increase efficiency across different search queries, data sources are represented by multiple attribute compositions. It was demonstrated that providing attribute compositions representing the most common search queries reduces the overall search messages by roughly 50%. While this approach already meets many established requirements, various extension possibilities are still conceivable.

The first extension possibility is adaptive attribute compositions, enabling the network to independently determine the most important ones over time and start them as needed. In the current implementation, the attribute compositions are fixed at the start of the network and must, therefore, be known from the beginning. The network could adapt to changing search queries by initiating new attribute compositions adaptively. However, the following points must be considered to achieve this. Firstly, the participants in the network must exchange information about common searches so that a democratic decision can be made about when each node should start a new attribute composition for a currently complex search. Second, there must be a mechanism to count searches across the network, and beyond a certain threshold, the participants must agree to start the new attribute composition. If not all participants do this, complete search results are no longer guaranteed. Therefore, new attribute compositions must be announced throughout the network and started by every participant.

The second extension is the additional function of data distribution in a citywide data space. For public data that is not preprocessed locally, both the services have to manage all data sources, and the individual data sources have to manage all services that are interested in their data. A group-based publish-subscribe approach is conceivable here, similar to the proposal by Teranishi et al. [249], where publishers and subscribers form a common namespace (a contiguous section in the Level 0 ring) and publish new data within this group through name-based routing.

Now that services acting as subscribers can find relevant data sources in a citywide data space, the following chapter examines the possibilities for local preprocessing of sensor data. Here, services define a set of preprocessing steps that each data provider runs locally. This shares the load of preprocessing between services and data sources and increases the privacy of the data providers by avoiding providing access to the collected raw data.

7 DISTRIBUTED PREPROCESSING IN A CITYWIDE DATA SPACE

After discovering relevant data sources, services can request these data streams by sending the owner's Connector a data request. Within this request, the semantic service description is added to give the owner an understanding of why sharing collected data might benefit the services and, therefore, all their users, maybe even the owner. Due to the growing privacy awareness of today's population, it is essential that the local preprocessing of raw data aims to reduce the reluctance to share personal data and to increase the privacy of each participant, reducing their privacy concerns. Therefore, services requesting citizens' data define a pipeline of preprocessing steps executed locally at the citizens' Connector. Only the preprocessed data is sent to the service, increasing citizens' privacy and decreasing network load by only sending relevant information. Local preprocessing of data on the Connectors of the data sources aims to meet the following requirements from Section 3.3: reducing network traffic to enhance the scalability and efficiency of the system and improving privacy and data protection by locally preprocessing raw data.

A major challenge, as already described within the problem statement in **SC3** (Section 1.1), is to explain these preprocessing steps comprehensibly, making them understandable not only for technically proficient citizens. This ensures that they can be understood by average citizens, enabling them to make informed decisions about sharing their data. Further, agreeing on a fixed set of modular preprocessing steps ensures that only data specified within the steps is sent to the service. A prototype has been implemented and used within a user study to evaluate the proposed approach regarding comprehensibility and usability. Some ideas and results from this chapter have previously been published in [140].

7.1 FUNDAMENTALS OF DISTRIBUTED PROCESSING

Before the architecture of distributed preprocessing in a citywide data space is introduced, this section establishes the basics of data sovereignty, private data, and edge processing. Especially in the digital age, private data has various definitions and numerous approaches to protect it. Additionally, the relevance of local raw data processing continues to increase with the growing amount of collected data. This is generally referred to as edge processing, as the data is no longer processed centrally on resource-rich servers. Still, directly at the edge, that is, on the sensors or their next instance with sufficient resources. This significantly reduces the data transmission and can also shorten response times due to the shorter paths, which is especially relevant in critical real-time applications.

7.1.1 *Data Sovereignty*

The term "Data Sovereignty" can be viewed from various perspectives, and its meaning varies slightly across different contexts. In this thesis, data sovereignty is considered from the perspective of individuals' private data. Here, data sovereignty describes the control over one's own collected data and its utilization across various services and platforms. This concept of data sovereignty underpins the notion that individuals should have the autonomy to manage, share, and restrict access to their data, aligning with broader aspirations for privacy, security, and personalized digital experiences in interconnected environments. At the core of data sovereignty in smart environments is the principle that residents and users should have unfettered access to their data. This encompasses the ability to view, understand, and make informed decisions about the data collected by smart devices and sensors in their homes and urban spaces. Such control ensures that individuals are not merely passive subjects of data collection but active participants in the data ecosystem. Furthermore, data sovereignty involves the capacity of individuals to authorize how third-party services and applications use their data. This means setting permissions for data access on a granular level and deciding which data can be shared, with whom, and under what circumstances.

7.1.2 *Privacy in Data Collections*

The following examines the concept of privacy in the context of sensor data and methods for protecting it in more detail. Privacy can be classified into two main categories: syntactic and semantic. Syntactic privacy ensures that it is impossible to associate individuals with specific data points in a given dataset. In contrast, semantic privacy asserts that the knowledge one has about an individual remains the same, regardless of access to the dataset.

Collected data is typically stored in tabular form, defining a set of attributes. These attributes can be categorized into four types:

- **Identifiers**: Attributes that uniquely identify an individual, such as `Social Security Number`.
- **Quasi-identifiers**: Attributes that, when combined, uniquely identify an individual. For example, 63% of the U.S. popula-

tion can be uniquely identified by combining `Postal Code`, `Sex`, and complete `Date of Birth`.

- **Confidential attributes**: Attributes that contain sensitive information, such as an individual's medical history or `Disease`.
- **Non-confidential attributes**: Attributes not considered sensitive by respondents, such as `Favorite Color`.

Syntactic approaches to privacy protection are based on two key assumptions:

1. The release of data tables can only compromise the privacy of individuals who contributed to the data collection.
2. The attributes that can be used to link sensitive information to individuals are limited to quasi-identifiers.

Under these assumptions, data can be protected against either identity disclosure or attribute disclosure. The concept of k-anonymity is employed to protect participant identities. A table is k-anonymous if each combination of quasi-identifier values appears either zero or at least k times. This is achieved through generalization and suppression of quasi-identifying attributes. For instance, birth dates can be generalized by omitting the day or the day and month. In cases where a single entry in August exists among many September entries, it is more information-preserving to suppress the August entry, allowing the birth dates to be generalized by only removing the day.

While generalization and suppression are applied to quasi-identifying attributes, sensitive and non-sensitive attributes remain unaltered. The challenge lies in finding an optimal k-anonymous table that minimizes generalization and suppression, an NP-hard problem.

Despite achieving k-anonymity, the sensitive attributes of a respondent might still be inferred through homogeneity or external knowledge attacks. If a specific combination of quasi-identifier values only occurs in table rows where the sensitive attributes are the same (homogeneity), the sensitive attribute's value can be deduced. Additionally, data recipients may apply external knowledge to narrow down the number of possible table rows corresponding to a specific respondent. To mitigate these risks, a table must also be l-diverse: each combination of quasi-identifiers must correspond to at least l different values for the sensitive attribute [156].

While syntactic techniques preserve data truthfulness, semantic approaches typically protect privacy by introducing noise into the data. A privacy-preserving dataset that still represents the original information is released in non-interactive scenarios. In interactive scenarios, individual queries over a data collection are evaluated in a manner that does not reveal sensitive information [65].

Ensuring data collection and dissemination privacy involves complex challenges requiring sophisticated methods. Both syntactic and semantic approaches provide valuable frameworks for protecting individual privacy while enabling data utility.

Syntactic methods become particularly relevant by considering privacy in self-managed data spaces, where only the citizens themselves grant access. Data providers must clearly understand which information or attributes the interested service uses to comprehend the extent to which the resulting dataset might include data that can be traced back to individuals.

Semantic privacy becomes crucial when entities attempt to gather different pieces of information about a person from multiple services, potentially accumulating more data than the owner initially intended to disclose.

In such scenarios, it is essential to ensure that data providers are aware of the attributes being utilized and implement measures that give citizens an overview of disclosed information to prevent personal data aggregation across different services without explicit consent. Nonetheless, a proposed approach must balance data utility and individual privacy, reinforcing the need for robust privacy-preserving mechanisms in a citywide self-managed data space.

7.1.3 *Edge Computing*

Edge computing, initially introduced through Akamai's Content Delivery Networkss (CDNs) in the late 1990s, represents a significant evolution in data processing and distribution. CDNs pioneered the concept by prefetching and caching web content on nodes located near end users, offering substantial bandwidth savings, particularly for video content. Modern edge computing extends this concept by enabling edge nodes to cache content and execute arbitrary code, providing several advantages over traditional cloud computing.

The proximity of edge nodes to data sources significantly reduces data transmission times, resulting in low end-to-end latency, high bandwidth, and minimal jitter. These characteristics are crucial for services requiring real-time responses, such as autonomous driving, intelligent manufacturing, and video surveillance. By processing data closer to the source, edge computing also diminishes the volume of data that must travel through the network and be stored in the cloud. High-data-rate sensors, like video cameras, generate extensive data that, if entirely transmitted to the cloud, would demand considerable bandwidth and storage capacity. Instead, performing analytics on edge nodes and transmitting only the results with associated metadata markedly reduces data flow.

Edge computing also enhances data privacy and security. Processing data on edge nodes within the sensor owner's trust domain allows for finer-grain control and enforcement of privacy policies. Reducing distance and time data travels minimizes the opportunities for attacks, and in the event of a security breach, only a limited portion of data is compromised. Also, edge nodes can serve as temporary fallbacks for cloud services, maintaining service availability during network failures, cloud outages, or denial-of-service attacks [38, 224].

Overall, edge computing represents a paradigm shift in processing and managing data. It substantially improves latency, bandwidth efficiency, privacy, and security, making it well-suited for citywide data spaces.

7.2 RELATED WORK

Sharing sensor data in raw form in various contexts is a subjective privacy concern for each citizen. Some want to maximize data usage, while others are more concerned about private information encoded within collected data streams. For example, one user shares temperature data measured regularly on the balcony with a requesting service. The service owner might notice a slight temperature rise every evening and interpret this occurrence as the balcony door being open at those times. In another example, the user wears a fitness tracker and shares the collected data with a requesting service. The service provider then knows when the user is going on runs and, therefore, is not at home. In both cases, the information is sensitive for a user to share. Also, it might not be necessary information for the service to have. Maybe the first service aims to average the temperature in a given neighborhood over a given period, and the second one illustrates heart rate changes while working out. Hence, it would be appropriate to process the user's data before sending it to the service. In cloud computing, processing data near the edge of a network instead of in the cloud is called edge computing [229]. A survey on edge computing by Khan et al. finds that the primary goals in the current research on edge computing are to lower costs, latency, and energy consumption [128]. They analyzed 48 different papers and compared them by their objectives. While they found 14 sources for minimizing latency, 15 for optimizing cost, and 8 for minimizing energy consumption, only three papers are concerned about maximizing privacy and two about strengthening security. Therefore, it is no surprise that one of the open challenges of edge computing they state is "user's trust on edge computing system".

The papers categorized as maximizing privacy and strengthening security are placed in cloud storage, help to minimize latency by preserving the privacy of the meta-data needed for this purpose, are conceptualized in a concrete context, or try to hide personal data from other parties altogether.

The scheme proposed in [271] splits the user's encrypted data into three different-size parts and separately saves them on the cloud server, the edge server, and the user's local machine. That way, even if an attacker gets all the data from one server, the attacker cannot recover the user's original data. Therefore, data is save from outside and from attackers inside the cloud or edge server. Contrary to high computational and storage capabilities in the cloud, single-edge nodes typically lack both. However, nodes cooperate to optimize their utilization in resource allocation schemes to overcome this problem. A gateway manages this cooperation and sends different data types to the suitable host(s): the cloud or one or multiple edge nodes. When deciding where to forward the data, the gateway must consider different factors, which must be defined and attached. Zhang and Li [285] focus on preserving the privacy of this meta-data not only from outside eavesdroppers but also from potentially corrupted gateways.

The cybersecurity framework proposed by Sohal et al. aims to prevent attacks by unauthenticated and unauthorized edge devices [236]. It works in three phases: First, an intrusion detection system categorizes a device as legitimate or malicious. Second, a two-stage Markov model eliminates false alarms by reevaluating devices recognized as malicious before. Third, devices that fail the first two phases are shifted to a virtual honeypot cloud, a decoy of the real system. The logs of an attacker's activities in an attack database repository are used to prevent unknown attacks in the future. In "Fog Computing: Mitigating Insider Data Theft Attacks in the Cloud" security is ensured by user profiling and decoy data [240]. The algorithm monitors when, how, and how much a user accesses the data and sends decoy data to a probable intruder if it detects unusual behavior.

Other approaches focus on the Internet of Things use-cases, where users share collected sensor data with a set of services [13, 14, 238]. Users then set access rights to the data they collect for these services, sharing only the data they are willing to. Approaches mainly differ in their applied access control schemes. While these let users and citizens decide which service is allowed to use (parts of) their data, citizens still have to trust the central authority managing the permission database. Besides, services must account for different data representations since users can specify the data the service can see. This applies in some scenarios and makes it easy to use for a wide range of citizens and services, but services may receive a large amount of raw data that is irrelevant to them. In other cases, services may only receive a specific subset where relevant information is missing, making the whole dataset unusable. The proposed approach in this thesis is more service-oriented and lets services define preprocessing steps for the required data. This results in already predefined datasets where no additional quality control (in the form of completeness) is required since either services get exactly the data they need or no data at all.

7.3 DISTRIBUTED PREPROCESSING ARCHITECTURE

The distributed preprocessing architecture is built upon the proposed architecture of SkABNet as a distributed discovery network. In the SkABNet architecture (Figure 13), an abstract *subscribe* and *publish sensor data* are mentioned after a successful search. While SkABNet only ensures the efficient search for relevant data sources and the exchange of addresses, this contribution focuses on how citizens can make their data streams available to interested services.

For this purpose, the interested service sends two pieces of information with its subscribe request: First, a description of itself (provided by the service operator during initialization), including the purpose of the service, its interfaces, and what data it needs for which purposes. Second, the service sends a sequence of preprocessing steps executed locally on the Connector managing the data sources. This local preprocessing offers three advantages. First, only relevant information is sent to the service, and potential private information is removed locally. Second, network traffic is reduced, as not all raw data needs to be sent, especially if it does not contain relevant information. Lastly, the service receives all data in a homogeneous format (the result of the preprocessing steps), simplifying processing.

One of the main goals of this thesis is to use existing data sources while respecting the owners' data sovereignty. Therefore, it is crucial to enable citizens to make an informed decision about whether they want to share their data and how it is used. This means that when citizens consent to provide their data, they must understand each preprocessing step and the resulting data transferred to the service. However, within a citizen-focused sensor network, it is assumed that the average citizen knows little about computer science and, thus, about any form of data processing (see Research Question 1.3). This means that the definition of preprocessing steps and the resulting data format must be understandable for all citizens. Furthermore, it must be ensured that malicious services cannot disguise their collected data. It must be impossible to deceive the data owners by displaying an explanation that does not represent the executed preprocessing step.

Additionally, service use cases are expected to be diverse, and a citizen-centric network needs to support a wide range of services in accordance with prior requirements. Therefore, preprocessing steps should be defined modularly so that each service can combine them to define the required data.

To fulfill these requirements, a selection of preprocessing steps are needed, each performing exactly one small task. These steps can be divided into three different categories:

- *Filters* take a list of data objects based on specified conditions and remove data objects from the list.

- *Modifiers* take individual data objects and remove or add properties.
- *Transformers* that alter the structure or order of objects.

By chaining multiple steps a step sequence is created reflecting the entire process of preprocessing requested data. Each step transforms the data and passes it further to the next step. The steps can be concatenated in any desired order and in any desired quantity, giving service developers maximum flexibility.

Since individual steps are kept simple and specific in scope, it is possible to create an automatically generated explanation for each step within a preprocessing pipeline. These individual annotations per step can then be combined with a flowchart depicting the chronology of the process to help citizens understand what is happening in each step individually and in the overall process.

To request preprocessed data, the sequence of steps is appended to a data request message and sent to all data providers of the required data type. The receiving citizens now go through the explanations for the steps and decide whether to accept or decline the request. If they decide to provide their data, the data is preprocessed according to the sequence of steps they have accepted.

7.4 PRIVACY PRESERVING PREPROCESSING

As described, developers define required data as a series of preprocessing steps. The JSON format is the most suitable representation for each step and the transferred preprocessing steps. It is independent of any programming language, is easy to understand and write for humans, and is supported by virtually all modern programming languages. A JSON schema containing all the allowed properties for every step type has been defined to help service developers. All participating citizens within the network must know the steps provided. Descriptions and functionalities are, therefore, shared beforehand and are part of the Connector software.

`https://www.json.org/json-en.html`

Structure of Preprocessing Steps

To keep the steps simple, the structure of each step is very similar. Each step contains at least its type and some input as part of the `properties`-array. For the developed prototype, a collection of eleven different preprocessing steps have been defined, which can be categorized as follows: “filtering”, “modifiying”, and “transforming”.

Filter

Steps that filter data objects remove entire objects from a list if they do not meet a certain condition.

WHERE This step removes data points that do not fit a queried condition. Additionally to the properties array, it contains a `predicate` property, which defines the condition that data objects need to fulfill to be chosen to stay in the list of results. The condition can be one of the following: `EQUAL`, `NOT_EQUAL`, `LESS_THAN`, `LESS_THAN_OR_EQUAL`, `GREATER_THAN`, `GREATER_THAN_OR_EQUAL`, `CONTAINS`, `NOT_CONTAINS`, `STARTS_WITH`, or `ENDS_WITH`. In the `properties`-array, the two values evaluated against each other under the condition are stated. At least one has to be a field of the data object, whereas the other can be a number, a string, or an array.

LIMIT Most of the time, the intermediate result contains multiple items. For use cases in which only a specific number of items is needed, the list can be limited.

Modifier

Modifiers are steps that either remove or add properties from the data objects. To remove properties, the modifier selects properties to keep and discards all other properties. Modifiers that add properties to data objects do a simple aggregation or calculation and store the result in a new data field.

SELECT With a selection step, the developer can choose which parts of the data object they want to forward as input to the next step and which are irrelevant to the current use case and can be removed for the subsequent processing. They do so by providing the names of the fields that should stay included in the `properties`-array of the step definition.

Many representations of sensor reading values may not be simple flat data objects but contain nested objects and arrays. To select fields deeper inside this nested structure, the developer prefixes the fields they want to select with the names of the objects and arrays they are in. Particular for this step is the `flatten` flag. If set to `true`, the step not only selects the specified fields but also alters the structure of the data objects by moving each property up one layer.

SUM adds up all values of a given object property of the objects in an array.

MULTIPLY is used to multiply values in two object properties together.

DAY_OF_WEEK converts a field containing a date into a number between 1 and 7, each representing a day of the week.

SIZE counts the elements in an array and stores the result in a new field. The first element in the `properties`-array defines the new field's name, whereas the second element specifies which array's elements should be counted.

ARRAY_ELEM_AT selects and stores an element of an array in a new data field. A subsequent SELECT step can reduce an array of objects to a specific single data object.

Transformer

Transforming steps alter the structure of data objects or the order of objects in a list.

GROUP groups every input element that has the same value in the field defined by the first element in the `properties`-array into one array.

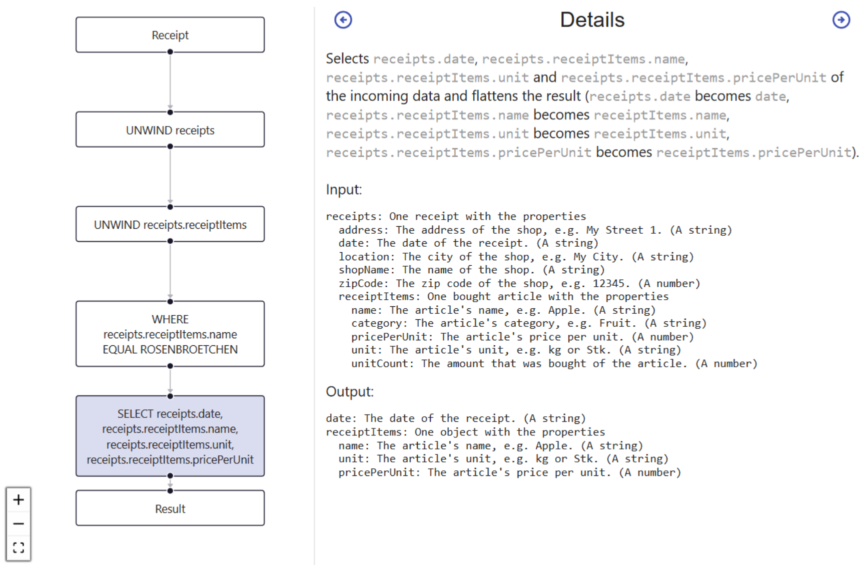
SORT uses the first element in the provided `properties`-array is the field name over which the input should be sorted. The second element states whether the sorting order should be ascending or descending.

UNWIND deconstructs an array contained in a data object. A duplicate of the whole data object is created for each element of the array. Instead of the array, the duplicates each contain only one element of the previous array. So, if the data object contains an array of length 20, 20 copies of the object are created, each containing one element of the array instead of the entire array.

Citizen Interface

As described in Section 7.3, every citizen must understand the pipeline of preprocessing steps that a service requires to decide whether data should be shared. A flowchart is generated using the steps the service requires to achieve this. Figure 21 shows an example where a service compares prices for a specified item (*Rosenbroetchen*) in different shops. Users can zoom in and out within the flow chart and select individual nodes to read more information about the step type on the right side. Each of the flow chart nodes contains essential keywords for the represented step. As soon as a user clicks on a node in the chart, the corresponding step explanation is displayed on the right side. The explanations are generated by filling the standard explanation phrases with the dynamic content from the step definitions. For step types with input and output displayed, a YAML format is created using the receipt type's JSON schema definition.

`https://yaml.org/`


Receipt
UNWIND receipts
UNWIND receipts.receiptItems
WHERE receipts.receiptItems.name EQUAL ROSENBROETCHEN
SELECT receipts.date, receipts.receiptItems.name, receipts.receiptItems.unit, receipts.receiptItems.pricePerUnit
Result
+
−

Details

Selects `receipts.date`, `receipts.receiptItems.name`, `receipts.receiptItems.unit` and `receipts.receiptItems.pricePerUnit` of the incoming data and flattens the result (`receipts.date` becomes `date`, `receipts.receiptItems.name` becomes `receiptItems.name`, `receipts.receiptItems.unit` becomes `receiptItems.unit`, `receipts.receiptItems.pricePerUnit` becomes `receiptItems.pricePerUnit`).

Input:

```
receipts: One receipt with the properties
  address: The address of the shop, e.g. My Street 1. (A string)
  date: The date of the receipt. (A string)
  location: The city of the shop, e.g. My City. (A string)
  shopName: The name of the shop. (A string)
  zipCode: The zip code of the shop, e.g. 12345. (A number)
  receiptItems: One bought article with the properties
    name: The article's name, e.g. Apple. (A string)
    category: The article's category, e.g. Fruit. (A string)
    pricePerUnit: The article's price per unit. (A number)
    unit: The article's unit, e.g. kg or Stk. (A string)
    unitCount: The amount that was bought of the article. (A number)
```

Output:

```
date: The date of the receipt. (A string)
receiptItems: One object with the properties
  name: The article's name, e.g. Apple. (A string)
  unit: The article's unit, e.g. kg or Stk. (A string)
  pricePerUnit: The article's price per unit. (A number)
```

Figure 21: Flowchart of an example preprocessing procedure. On the left, a chart with all individual steps is displayed. Details are shown on the right by selecting a single step, describing how the data has been changed.

7.4.1 *Service Interface*

Service developers must create a JSON containing all the steps providers must execute before sending their data to the created service. This can happen in the integrated development environment of the developers' choice, but it needs to be provided to the Service Interface when the service is initiated. This JSON is then sent to all discovered data providers with a subscription request.

7.5 PROTOTYPE AND EVALUATION

A prototype with the eleven steps mentioned before is implemented to evaluate the proposed approach. With these steps, preprocessing steps for five different services, each analyzing receipt data collected from grocery stores, are defined. Each data point within the examplary citizen sensor network is therefore a single receipt, containing the list of items, each with a price per item, number of bought items, and a category such as *vegetables*, *frozen food*, and *household goods*. Further, the store's location and the purchase date and time are available for each receipt. To analyze the usability at first, the extent to which participants with different knowledge levels of programming understand the meaning of a given series of steps was assessed. Second, the user study examined whether participants with at least some programming knowledge can define a series of steps for a given task.

7.5.1 *Example Services*

Following five example services that use the eleven preprocessing steps are introduced. These services show that a wide range of services can already be created with a relatively small number of preprocessing steps.

Price Trend of Articles

One of the most interesting analyses for customers is how the price for a given article changed over time. While this information is easily scrapable for online shops, getting a price overview for grocery items can be challenging. By collecting data over time and from different sources, a service could provide a historical graph showing the price movement for different items, and customers could decide whether to buy them at the current price.

This service needs a list of all purchased items, including the price per item and the timestamp of the purchase. Therefore, all items need to be formed into a single list; this can be done with the *UNWIND* step. Finally, the required information is selected and transferred as a list to the service.

Top 10 Most Bought Articles

Another service contains the list of the ten most frequently purchased items. This list can be accessed with the following preprocessing steps. Following the UNWIND step to create a single list with all bought items, a *SELECT* step selects the item name and the number of purchased items. A *LIMIT* step selects the ten most frequently occurring article names by sorting from most to least occurrences. In the result, the article names paired with the number of their occurrences on receipts are given.

Expenses per Category

A service that calculates the expenses per category could help identify where the most money is spent. In the preprocessing, the price per unit and the category of the articles are selected from the receipts, which are represented by a list of items after using a UNWIND step. A *MULTIPLY* step multiplies in pairs the price per unit with the unit count to get out the actual expenses. Within a group where all articles of the same category can be found, the just calculated expenses are *SUMMED* up and finally returned together with the corresponding category.

Purchases per Weekday

By grouping purchases per weekday, the days with the most traffic within grocery stores can be defined. With more data, an additional grouping could be done, leading to information about the number of customers per weekday and shop. With a *DAY_OF_WEEK* step, the date of each receipt is converted into a number representing a weekday. A group is formed each weekday, and the number of items per group is counted. The days of the week and the corresponding counts are returned.

Last Price of a single item per Location

To find the location that sells a specific item for the lowest price, a service requires all items, including the price per item and the purchase location. After an *UNWIND* to receive a list of all bought items, a *SELECT* is used to find only the item searched for. After that, the results are ordered in descending order so that the most recent purchase is at the front. Then receipts with the same address are put into a group, and the first item of each group is stored at `latestReceipt`. Finally, the result is flattened by several SELECT steps so that for each address, one object contains the address, the price per unit, and the date.

7.5.2 *Usability Study*

For the usability study, 13 participants were asked to complete a survey and perform multiple tasks. At the beginning of the survey, participants were asked basic questions about themselves. Eight of the participants were male, and five were female. Nine participants were between the ages of 20 and 29, three were between 30 and 39, and one was between 60 and 69. One of the participants rated their English skills on a scale of 1 (*not at all*) to 5 (*very well*) as a 2, three as a 4, and nine as a 5. Only one of the participants rated their computer skills on a scale of 1 (*very poor*) to 5 (*very well*) as a 2, one as a 3, six as a 4, and five as a 5. Four of the participants rate their experience in computer science, especially software development, on a scale from 1 (*not at all*) to 5 (*very well*) as a 1, one as a 3, four as a 4, and four as a 5. One participant uses a computer for less than 1 hour a day, two for 3 to 5 hours a day, eight for 5 to 8 hours a day, and two for more than 8 hours a day. Seven of the participants stated they use their computer at least once a week for gaming, twelve for writing, twelve for writing and reading e-mails, all for surfing the internet, ten for watching videos, and eight for programming. Two added *studying* to the list, *work*, *online shopping*, and *design* were added by one person.

Tasks

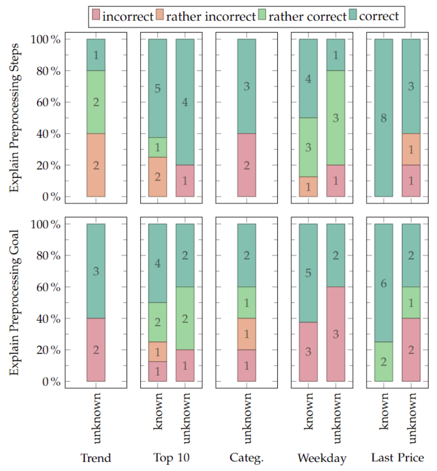
After answering the basic questions, participants were divided into two groups based on their knowledge of the JSON format. This was based on the question "Do you know the JSON format, and are you able to write a JSON-object?" inside the survey. If they answered "no", they were given five different sequences of steps to explain. If the answer was "yes", they were given three step sequences to explain and two tasks to implement a step sequence. Each of the tasks addressed one type of preprocessing of receipt data.

Tasks in which participants were asked to explain the step sequences used the five examples from above and were structured as follows: First, they were shown a visualization of a step sequence as shown in Fig. 21. They were motivated to inspect each step and read the details presented. In the survey, they were then asked to describe in their own words what happens in each step and what they think could be learned from the resulting data should this sequence of steps be executed. They were then asked to rate how well they thought they understood the sequence of steps on a scale from 1 (*not at all*) to 5 (*very well*). In the next question, they were given four possible goal descriptions, from which they had to choose one as the goal of the sequence steps they had seen before.

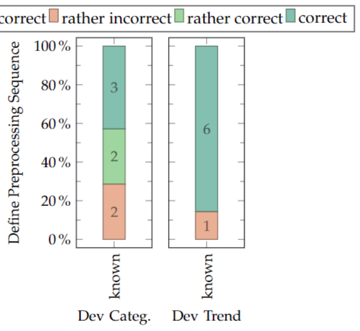
As mentioned before, participants with JSON knowledge had three tasks to explain the behavior and had to write the missing two sequences (price trend and expenses per weekday) themselves. These participants were provided with two different tools to complete this task. First, in the IDE of their choice (the recommendation was Visual Studio Code), they could use the JSON schema created for the set of step types to have code completion, error markers, and populated comments available. Second, a web application was created where users could paste a step sequence definition into a text field. As long as the definition was in a valid format, the flow chart and associated explanations were generated. In the survey, they should insert their two final results.

7.5.3 *Evaluation*

Each survey participant was shown five different evaluation tasks, each relating to a different preprocessing sequence. Participants without JSON knowledge were shown five preprocessing sequences and needed to explain which data is send to the service. Participants with JSON knowledge had to explain three preprocessing sequences and develop the remaining two themselves.
First, participants had to select a goal for the preprocessing pipeline shown. The participants selected the correct goal in 29 of 36 cases or 81% of the time. Of the seven incorrect selections, two were from par-



(a) Correctness of responses to open-form text questions asking participants to explain a given preprocessing definition. The responses are split by whether JSON is known or unknown to the participant.



(b) Correctness of the preprocessing definitions written by the participants that know JSON for the services: *Expenses per Category* and *Price Trend of an Article*

Figure 22: Results of the explanation for the given preprocessing definition and the development tasks.

ticipants who knew JSON, and five were from participants who did not. All five participants who did not know JSON selected the correct goal for the first task *Price Trend*. Only 8 of the total 13 responses were correct in the last most complicated task. In the other two tasks, *Top 10* and *Expenses per Category*, only one of the people who did not know JSON could not correctly identify the goal.

In addition to selecting the correct goal out of four different options, participants had to explain each preprocessing step and its goal in their own words. Figure 22a visualizes the explanation correctness for both the steps and the preprocessing goal, graded into *incorrect*, *rather incorrect*, *rather correct*, and *correct*.

Overall, the sequence steps explanation was correct in 29, rather correct in 9, rather incorrect in 6, and incorrect in 5 out of 49, or equivalently in 59%, 18%, 12%, or 10% of cases. Among the participants who knew JSON, 21 of 24 or 88% of the explanations were either correct or rather correct. Among the participants who did not know JSON, this number is at 17 of 25 or 68%. None of the participants who knew JSON gave a preprocessing steps explanation that was graded as incorrect. For all tasks, both groups gave an explanation that, on average, was either rather correct or correct.

The preprocessing goal explanations were correct in 26, rather correct in 8, rather incorrect in 2, and incorrect in 13 out of 49, or 53%, 16%, 4%, or 27% of cases respectively. The participants who did not know JSON gave rather or fully correct goal explanations about as often as those who knew JSON, with 15 of 25 or 60% and 16 of 24 or

67% respectively. Similar to the step explanations, two of the goal explanations given by the group that did not know JSON were incorrect as the participant only entered "as before" as the answer.

Figure 22b shows the results of the two preprocessing sequence development tasks shown to the participants who knew JSON, also graded on a four-point scale from incorrect to correct. One of the participants who knew JSON did not submit any preprocessing sequence definitions as they filled out the survey on their smartphone and thus could not open a code editor or view the step type reference. This participant was excluded from the results, leaving only seven instead of the eight participants who knew JSON.

7.5.4 *Discussion*

The single-choice questions relating to the sequence's goal show that the participants understood the goal of the sequence correctly in most cases. Overall, the explanations could convey the sequence goal both to citizens with and without a computer science background. The results of the free-form explanation tasks (see Figure 22a) showed that all citizens, whether they knew JSON or not, were able to correctly explain the sequence steps in many cases or at least rather correctly in most cases. This shows that most users can understand the individual steps of a sequence. When describing the overall goal of a preprocessing step sequence, the correctness mostly follows that of the step explanations. On average, the goal description is less correct than the step description. This indicates that it was harder for the participants to understand the more complex dependencies along a step sequence. With a slight misunderstanding in one of the steps along the sequence, the independent explanations of the steps would still be mostly correct. However, regarding the goal, a misunderstanding along the sequence leads more easily to an incorrect understanding of the overall goal. Therefore, each step must have a simple and easily understandable description.

Further results indicate that the participants without JSON knowledge were especially unsure of the price trend of a single item in the first task. Participants scored better on the following tasks, which indicates that they gained a better understanding after seeing one or two examples. It also indicates that explanations should be improved further for the first contact.

The results of developing a preprocessing sequence for a given service show that most developers understood the tasks well and could create a correct preprocessing sequence. These results could be optimized with additional resources to learn preprocessing steps by giving a better introduction. In general, the results indicate that potential service providers would be able to define their preprocessing steps within the limits of the currently implemented preprocessing steps.

7.6 SUMMARY

This chapter analyzed whether citizens benefit from distributed preprocessing for their locally collected data. Therefore, a set of modular preprocessing steps are defined that services could use to create a more complicated preprocessing pipeline that receives only the required data. Results show that, with an intuitive user interface and a comprehensive explanation for each step, even citizens with no computer science background understood most of the preprocessing steps requested by a service. Moreover, developers with no prior introduction can quickly start to define their own preprocessing pipeline using the provided steps to retrieve only relevant data for their services.

One of the main goals in the future is to improve the user interface by highlighting changes and combining multiple smaller steps into recipes with a more meaningful explanation, making it even easier to understand data preprocessing and usage. Further, with additional changes and simplifications within the user interface, a more extensive user study is planned to support current findings. Additionally, to make the system more adaptive, how the community can add new preprocessing steps at runtime needs to be investigated. Currently, every participating Connector must know all preprocessing steps beforehand. Similar to the new attribute compositions within the discovery overlay, new preprocessing steps and recipes could be published and agreed upon within the network. By utilizing a distributed consensus algorithm, the system would provide more flexibility and enable even more services.

After introducing a framework for describing a preprocessing pipeline in this chapter, participating services can now obtain relevant data from the citywide data space. However, an essential question for service providers is to what extent they can trust the data from self-installed sensors. The following chapter examines how the placement of data sources can be classified based on their collected data. This classification allows services to decide whether to use these data sources.

8 ANALYSIS AND EVALUATION OF DATA SOURCES

After services find relevant data sources through the distributed discovery overlay and request the data streams with a pipeline of preprocessing steps, the service continuously receives new data. A challenge, also reflected in Research Question 2.2, is assessing the data quality. The increasing size of a citywide data space raises the risk of misconfigured or faulty sensors impacting data accuracy. This, in turn, influences the conclusions that can be drawn from collected data. With decreasing data quality, erroneous conclusions could be drawn, impacting decisions based on these. To mitigate this, sensor deployment is often standardized to ensure data quality. However, this level of control and expertise is only feasible in networks overseen by a single operator with sufficient resources for monitoring and maintenance. In a citizen-centric data space, this kind of maintenance is not possible. Not only are citywide data spaces usually too large to maintain manually, but requiring experts to set up sensors would increase the hurdle for new participants to join and could thereby limit the number of sensors contributing to the data space.

Since it is unfeasible to check the deployment configuration of each individual sensor in a citywide data space, a software solution is required to process measurements and separate high-quality measurements from low-quality ones. As soon as an incorrectly deployed sensor is detected, the measurements of this sensor need to be ignored when processing the dataset. By comparing measurements from all participating sensors, the same software could extract characteristic deviations within collected data within groups of sensors that share a common context. Such deviations can be expected or unexpected, depending on what is being monitored, possibly requiring observers to take corrective action to avoid a further skewing of measurements [153]. Integrating this kind of software as a service into citywide data spaces provides an opportunity to extract additional knowledge from the network, such as exploiting spatial-temporal dependencies between nodes to enrich measurements with insights not available to a single sensory node [49]. These kinds of dependencies are implicitly encoded in the data. They can be identified as anomalies if they only affect a single sensor node or a common deployment context that affects measurements of sensor nodes that might be physically separated from each other but monitor the same phenomena [80, 187].

The remainder of this chapter is structured as follows: Section 8.1 gives a short overview of the fundamentals, defining anomalies, events, and preprocessing steps that need to be taken to analyze collected data. The fundamentals are followed by an overview of related work in Section 8.2, with existing machine learning approaches. Required components for the developed software pipeline are described in Section 8.3. Finally, in Section 8.4, evaluates the proposed software, and Section 8.5 concludes this chapter by discussing the results.

8.1 FUNDAMENTALS

The following section discusses the fundamentals of various anomalies that can occur within time series data generated by sensors. Each time series contains measurements of a single type (temperature, humidity, e.g.) taken at a specific place at a specific time. Anomalies can range from a single measurement deviating to a whole group of successive measurements that differ strongly from surrounding measurements collected by other sensors. Subsequently, different types of correlation between anomalies are defined, which can lead to interconnected anomalies defined as *characteristics* of a group of sensors. Since an isolated anomaly can be classified as an outlier and filtered out, anomalies that occur within measurements of multiple sensors or are repetitive over a given time are most likely caused by a common context that triggered these anomalies, indicating a characteristic of the common context of affected sensors.

8.1.1 *Anomaly types*

To understand how anomaly detection across multiple sensor nodes can benefit the data quality and characteristic detection within a city-wide data space, a definition for an anomaly is provided by Grubbs (1969), quoted by Shahid et al. [227, p.195]:

Definition 1 *An outlying observation, or outlier (**anomaly**), appears to deviate markedly from other members of the sample in which it occurs.*

How an outlier or anomaly presents itself in a stream of measurements varies depending on its cause. The simplest type of anomaly within a stream of measurements is a *Point Anomaly* [42, 103]. Point anomaly is an individual data point deviating from the "normal" distribution of values within the data stream as displayed in Figure 23a. Since these anomalies are not correlated to other data points within the stream, they are often considered noise and should be removed before the data is processed further. As soon as a correlation exists, this anomaly is defined as a *Collective Anomaly*. A collective anomaly can be identified when a collection of related data instances is anomalous concerning the entire data set. In this case, an individual instance of data measured during the anomaly is not anomalous, but their occurrence together as a collection is [42], as seen in Figure 23b.

Finally, the context in which some data points occur determines whether they are anomalous or not. For this reason, these kinds of anomalies are called *Contextual Anomalies*. When analyzing contextual anomalies, two attributes need to be taken into consideration, namely [42, 103]:

1. *Contextual attributes*, which are used to determine the context for a data instance,
2. *Behavioral attributes* define a data instance's non-contextual characteristics.

To illustrate how a context anomaly presents itself in a data distribution, consider Figure 23. The *contextual* attribute would be the time scale, and the *behavioral* attribute would be the cosinusoidal fluctuations, defining the "normal" temperature behavior during certain times of the year. Using the behavioral attribute values within a specific context identifies a contextual anomaly at t_2. Even though the temperature is the same at t_1 and t_2, given the context that t_1 is measured in winter and t_2 in summer, the measurement at t_2 would be considered an anomaly [42].

8.1.2 *Correlation of data*

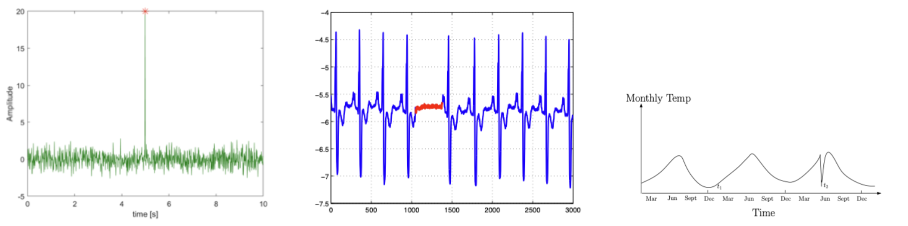
While the reason behind an anomaly on a single sensor node is hard to detect, when anomalies across multiple sensor nodes are combined, they can be grouped and are easier to analyze. Therefore, a context characteristic can be defined as:

Definition 2 *A **context characteristic** is a sequence of data anomalies that occurs across multiple data sources over extended periods, which allows identifying historical patterns [80, 227].*

Since sensor nodes are distributed within a city-wide sensor network, sensors affected by the same context have similar anomalies, and therefore, a spatial correlation is present between their measurements [49, 108, 187].

Definition 3 *A **spatial correlation** exists between nodes in a sensor network when they are deployed spatially dense to each other, resulting in multiple nodes sampling a similar data distribution [187].*

Similarly, context characteristics affecting multiple spatially independent sensor nodes simultaneously or in predictable matter encode a temporal correlation in their collected measurements [187]. Furthermore, measurements collected at one instant on a single sensor node are related to previous measurements on the same node. Therefore, a temporal correlation can be observed between readings produced by a single or a group of sensor nodes [108].



(a) A *Point Anomaly* presenting itself in a distribution with a value five times higher than the next highest value [103]. No other value shows a similar value, and no pattern can be detected.

(b) In the above graph, an individual value within the red marked section would not be an anomaly, but the presence of a collection of these values is considered an *Collective Anomaly* [42].

(c) A time series containing a *Contextual Anomaly* that has the same value as other measurements but is isolated and not matching the expected value [42].

Figure 23: Three different types of anomalies increasing in complexity.

Definition 4 *A* ***temporal correlation*** *arises when a predictable relationship exists between sequential measurements on a single or a group of sensor nodes [187].*

Understanding the relationship these correlations have with each other and the information contributed by their existence is imperative. It ultimately forms the basis of the approach for detecting contextual characteristics investigated in this chapter.

8.1.3 *Time Series Invariances*

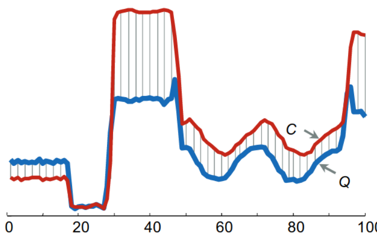
Before the similarity between two time series can be measured, it is important to understand which kinds of invariances can occur and their effect on the different similarity metrics.

Amplitude Invariance

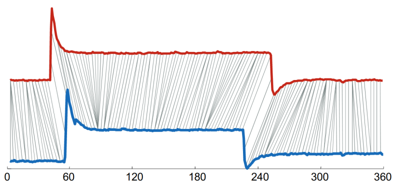
If two sensors monitor the same phenomenon, it cannot be assumed that the same values are being measured. Internal differences in sensor configuration or the intensity with which a context affects sensors may result in curves with similar shapes that offset each other. This kind of invariance is termed *Amplitude Invariance* [25] and can be identified in Figure 24a.

Warping Invariance

Given the continuous nature of context characteristics, such as exposure to the sun at a given time of day, it is unlikely that sensors affected by these are influenced simultaneously. Often, there is a shift in the time between sensors being affected. These kinds of shifts in



(a) Amplitude Invariance between two time series affected by the same context C and Q [25].



(b) Warping Invariance presenting itself when comparing two time series affected by the same context at slightly different times [25].

Figure 24: Two different invariances showing time series data affected by the context.

the curves are referred to as *Warping Invariance* [25] and present themselves as seen in Figure 24b.

Complexity Invariances

In addition to considering the shifting of time series, as done in the previous invariances, the complexity of a time series can also influence the effectiveness of a distance metric [25]. Comparing the number of peaks and values in a time series and calculating the similarity can help one intuitively understand its complexity invariances. It has been shown that when complexity invariance is not considered, the distance between a pair of complex time series is often greater than the distance between a pair of simple time series [25] and that complex time series are usually found to be more similar to a simple time series than another complex time series [25]. While the effect of complexity invariance can be somewhat mitigated by using an approach that corrects for warping invariance [25], it is advantageous to not only rely on these methods.

Multiple Invariance

It is also possible that two time series might differ because of a combination of the invariances mentioned above. This is most often the case since each Connector in a citywide data space has its own sensors connected, thereby introducing potential amplitude invariance. While the sensors might be sampling the same distribution, they are not deployed in the exact same physical location, thereby introducing potential warping invariance. The complexity of time series can also differ easily, even when nodes have the same hardware and run the same software. Communication problems can result in missing values that must be considered when computing the similarity. Therefore, the metric that describes the similarity between time series needs to address these invariances.

8.2 RELATED WORK

When attempting to detect anomalies in time series, most approaches can be categorized as belonging to one of two groups: either an anomaly is identified as discrepancies from a model of expected values, or time series are analyzed for containing patterns learned from models of a known anomaly [87]. The first category can identify anomalies that might represent unknown contextual characteristics. If measurements deviate far enough from the expectation, these measurements are considered a series of anomalies. If a series of anomalies is detected within a group of sensors, they are classified as a new unlabeled characteristic. The second group of approaches is more suited to accurately detecting already known characteristics, but it needs existing models representing the series of anomalies that represent this characteristic. Multiple models are required to detect multiple characteristics, increasing the detection process's complexity.

8.2.1 *Predictive Models*

Predictive models for anomaly detection belong to the first category introduced above and rely on using previous observations to anticipate future values a sensor node produces [42]. Any new, unseen data is compared to the predicted value from the model to determine to which class it belongs. The input is accepted as normal if the measured value is similar to the predicted value. However, if the actual value differs too much from the predicted value, it would be classified as an anomaly [42, 103]. If multiple subsequent anomalies are observed, they can then be classified as a characteristic.

8.2.2 *Autoencoders*

A common method used for generating a time series model is using autoencoders [93, 205]. Autoencoders are specialized neural network topologies that are trained in an attempt to copy their input to their output [93]. Due to this unique characteristic, autoencoders are often called semi-supervised learning methods since they partially rely on techniques more commonly found in supervised learning. Consisting of an input layer, 1 to n hidden layers, and an output layer, an autoencoder can be divided into an encoder function $h = f(x)$ and a reconstructing decoder function $r = g(h)$.

8.2.3 *Support Vector Machines*

Another model-based approach for anomaly detection in time series has been developed using Support Vector Machines (SVMs), specifically in the form of one-class SVMs [283]. This approach utilizes the excellent generalization provided by SVM-based models and extends it using a specialized kernel. Using this kernel improves the performance of identifying anomalous time series [42]. Given that SVM is a supervised method, labeled data is required to train these models.

8.2.4 *Deep Temporal Clustering*

Deep Temporal Clustering was developed to extract informative features on various time scales, which makes it ideal for handling longer data sequences [157]. This is achieved using a specialized temporal autoencoder, which can achieve greater dimensionality reduction by collapsing input sequences in all dimensions except temporal [157]. Once the dimensions have been reduced, the internally encoded sequence is fed into the clustering layer.

The novel temporal clustering layer, built specifically to handle unlabeled spatial-temporal data [157], was agnostic of the similarity metric used for the actual clustering. This makes the framework flexible, allowing it to apply to data gathered from various domains by using a similarity metric that is best suited for the data. Furthermore, this allows simplified prototyping with different metrics to determine which forms the most meaningful clusters.

8.2.5 *Swift Event*

SwiftEvent [87] is an anomaly detection algorithm that detects already known characteristics. Relying on labeled data, SwiftEvent learns the characteristics of a specific, user-defined anomaly using supervised learning methods. Once the detection of these anomalies has been learned, the model can be applied to real-time data to detect the occurrence of learned anomalies. Training of the models is achieved by projecting marked anomalies into a representation befitting the features. Similar anomalies have similar projected representations, forming clusters. A model is trained for each of the formed clusters, which can then be used to detect the presence of the characteristic.

8.3 CONSTRUCTING A PIPELINE

In this section, a service is created utilizing previously presented techniques. This service is deployed in the citywide data space and enables other services to classify participating sensors based on their collected data. The proposed service contains a pipeline of steps that identify and classify anomalies in data streams, and clusters data sources based on the corresponding characterizations triggering these anomalies. For this purpose, each step that the pipeline must fulfill to cluster sensors based on their contextual characteristics is described. Time series data from a large set of sensors measuring a specified type of data is needed as an input for this pipeline. Therefore, a large set of participating citizens need to contribute their data streams to the service. Nonetheless, participation is voluntary to adhere to the citizens' data sovereignty. To increase participation, citizens could benefit from the service themselves by identifying misconfigurations or falsely placed sensors, helping citizens to get a deeper understanding of their collected data. The developed pipeline outputs a categorization of the sensors and their data into different categories that are not labeled. Rather, the differences between the clusters are highlighted so that in an additional manual step, it can be determined which context affects sensors in these clusters.

8.3.1 *Similarity Metrices*

When performing any clustering on data, choosing an appropriate similarity measure is important to ensure the formation of logical clusters. Choosing a similarity metric for time series data is not as simple as it is for geometric distances. As discussed in Section 8.1.3, different invariances influence the distance between two time series affected by the same real-world phenomena. Different approaches exist to measure the similarity between two time series data streams considering different invariances. Most common distance metrics are lock-step measures such as the Euclidean Distance [79], while others use elastic measures such as Dynamic Time Warping [28].

Euclidean Distance

The most common distance metric used for clustering is the Euclidean Distance [113]. Due to its simple implementation, it is often the first metric to be considered when forming clusters.

The Euclidean Distance requires two time series Q and C of the same length n,

$$\begin{aligned} Q &= q_1, q_2, \dots, q_n \\ C &= c_1, c_2, \dots, c_n \end{aligned}$$

and can be calculated as follows:

$$ED(Q, C) \equiv \sqrt{\sum_{i=1}^{n} (q_i - c_i)^2}$$

Euclidean Distance, therefore, compares all measurements at the same index; this makes it vulnerable to all invariances. Potential sensor configuration and placement differences lead to invariant curves when considering time series data from different deployed sensor nodes.

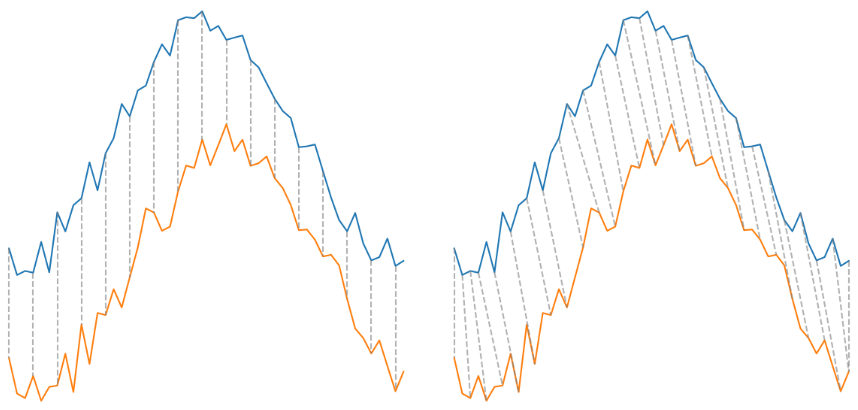


Figure 25: A comparison of using Euclidean distance (left) and DTW (right) to measure similarity between time series[286].

Dynamic Time Warping

One solution to consider invariances is Dynamic Time Warping (DTW). DTW is a method used on time-dependent data to find an optimal alignment between two time-dependent sequences [25, 181, 247]. Often used in speech recognition, this method has successfully matched similar speech patterns spoken at different tempos [181].

When considering Figure 25, a comparison between using a lock-step distance such as the Euclidean Distance to determine the similarity between time series data (left) and using DTW (right) can be seen. In causal relationships, there is often a lag in a phenomenon's presentation to individual observers. As an illustrative example, consider multiple sensors deployed densely together but in cardinal directions of a building monitoring temperature. While sensors deployed east of the building would measure higher temperatures in the morning and lower temperatures in the evening. The opposite happens with sensors deployed west. Meanwhile, sensors deployed south have temperatures that are nearly consistently higher. Despite this, all the affected nodes would observe the same characteristic and should, therefore, be clustered together. If the Euclidean Distance is used when comparing time series, data points measured simultaneously would be compared. Therefore, it is necessary to use a metric impervious to this effect when comparing time series. DTW is an attempt to consider the warping invariance of the data when calculating the similarity.

Formally DTW can be defined as an optimization problem. Given two time series $X = (x_0, \dots, x_{n-1})$ and $Y = (y_0, \dots, y_{m-1})$:

$$\mathrm{DTW}(X, Y) = \min_{\pi} \sqrt{\sum_{(i,j)\in\pi} \left\|X_i - Y_j\right\|^2}$$

where $\pi = [\pi_0, \ldots, \pi_K]$ is a temporal alignment of time series such that the Euclidean Distance between aligned time series is minimal [247].

While the alignment of peaks and valleys also indirectly improves dissimilarities introduced by complexity invariance, this method does not account for the varying number of peaks and valleys that might be present in the series and is, therefore, potentially unable to find the best alignment.

One of the main disadvantages of DTW is the quadratic time and space requirements of the algorithm [220]. This limits its use to cases with a relatively small amount of time series data since using this algorithm would otherwise be infeasible. An optimized implementation of DTW, called FastDTW [220], was developed with a linear time and space requirement, thereby significantly increasing the cases in which this metric can be used.

8.3.2 *Clusters in a Dataset*

After examining methods to determine the similarity of time series data, the service needs to cluster time series data sources based on the similarity in their collected data. Since the number of distinguishable characteristics is unknown, the number of clusters must first be found. Therefore, a method of empirically determining the number of clusters that could be present in the data needs to be defined. The following introduces two heuristic methods that are commonly used when no additional knowledge is available concerning the number of clusters in a dataset.

The Elbow Method

One of the most commonly used methods is the *Elbow Method*. This method works by iteratively increasing the number of clusters to be formed and examining the effect the formation of additional clusters has on the *inertia*.

Definition 5 ***Inertia*** *is the sum of squared distances of samples to their nearest cluster center, also known as an intra-cluster variance.*

$$\sum_{i=1}^{N} (x_i - C_k)^2$$

where N *is the number of samples within the dataset, and* C *is the center of a cluster.*

Initially, an increase in the number of clusters significantly affects the intra-cluster variance. If this is not the case, it would indicate that the data could not be divided into multiple clusters and would signal the end of the analyses. However, as the number of clusters increases, a

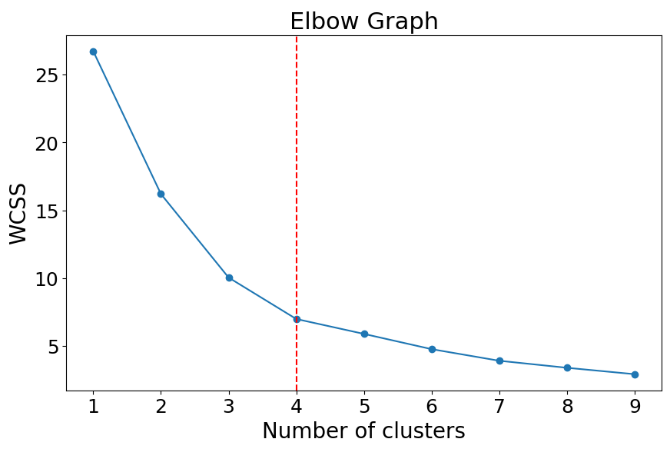


Figure 26: An example of an elbow graph with the optimal value for k marked in red

point is reached after which the effect on the inertia is drastically reduced, visually resulting in an *elbow* when plotted. This effect results from the clustering algorithm's inability to decide to which cluster a sample should belong, and it is, therefore, an indication that too many clusters have been formed. Since the inertia does not reduce further after a certain number of clusters, increasing the number of clusters offers no advantage. This number of clusters is the optimum number, after which the inertia decreases linearly when more clusters are added. To demonstrate what such an elbow curve might look like, consider the curve formed when clustering an example dataset, as seen in Figure 26.

From Figure 26, it is clear to notice that after three clusters have been formed, additional clusters have less influence on the variance. This is an indication, therefore, that the model is overfitted for the data, and meaningful clusters are no longer being formed; instead, the data is split more randomly.

Silhouette Score

Another heuristic commonly used for determining the correctness of formed clusters is to calculate the so-called *Silhouette Score* [216]. This score measures the inter- and intra-cluster variation to determine the effectiveness of formed clusters. Similar to the Elbow Method mentioned previously, determining the Silhouette Score relies on iteratively increasing the number of clusters into which the data should be divided. After each iteration, the Silhouette Score is calculated for each sample assigned to a cluster.

After data has been assigned to their respective clusters, two measures need to be calculated before the Silhouette Score can be determined. For a sample i assigned to cluster A, determine

$$a(i) = \text{average dissimilarity of } i \text{ to all other samples in } A$$

as well as

$$d(i, C) = \text{average dissimilarity between } i \text{ and all other clusters}$$
$$\text{which is not } A$$

Once $d(i, C)$ has been determined, which results in a list of dissimilarities to other clusters, the distance to the nearest other cluster is of interest. This value denotes the cluster to which i would belong if not assigned to A. This can be calculated as

$$b(i) = \min_{C \neq A} d(i, C)$$

Upon further consideration of the equations above, it becomes clear that at least two clusters need to be formed to calculate the Silhouette score; otherwise, it would not be possible to determine the distance to the nearest other cluster. As a result, the score is undefined when only forming a single cluster. After determining these measures, the Silhouette Score can be calculated as:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

The resulting score $s(i)$ lays in the range

$$-1 \leqslant s(i) \leqslant 1$$

A great advantage of the Silhouette score is its simple interpretation. A calculated score of 1 indicates a well-formed cluster. For a particular sample i where $s(i) = 1$, the closest possible alternative cluster is maximally dissimilar, thereby indicating a large inter-cluster variance, as well as i being maximally similar to other samples contained in the same clusters, or the cluster having a low intra-cluster variance. However, if i is calculated with a score of 0, it is a 50/50 decision to which cluster i should belong. Therefore, it is equally similar to the other data in its assigned cluster and to the data contained in the closest other cluster. The worst-case scenario would be a score of -1. In this case, i would be more similar to the nearest neighboring cluster data than its current cluster. This indicates that i has been assigned to the incorrect cluster. When evaluating the results produced by calculating the Silhouette Score, the number of clusters with the highest score would, therefore, result in the best clusters.

8.3.3 *Clustering of time series data*

To identify contextual characteristics, the service first needs to identify sensors deployed in a similar context. To do so, sensors are clustered based on the similarity of their collected data. Each cluster contains all sensors placed in a similar context; therefore, their collected data show similar characteristics. The following introduces different algorithms to find the most expressive clusters.

K-means clustering

The *K-means* clustering algorithm is possibly the most well-known method in the *partitional* clustering category.

Definition 6 *In Partitional Clustering, clusters are characterized by a central vector, and data points close to these vectors are assigned to the respective clusters.*

This method attempts to minimize the intra-cluster variance while maximizing the inter-cluster variance [3, 274]. This leads to maximally dense clusters (i.e., the points in a cluster are maximally similar) while clusters are maximally separated by some distance metric. Formally, this optimization can be expressed as:

Definition 7 *Given a set of* d*-dimensional observations* $(x_1, x_2, \ldots, x_n)$*, partition* n *observations into* k *sets* $S = \{S_1, S_2, \ldots, S_k\}$ *such that*

$$\sum_{i=1}^{k} \sum_{x,y \in S_i} \|x - y\|^2$$

returns the minimal sum of variances within the clusters.

This method works by randomly partitioning objects into nonempty subsets, constantly adding new objects, and adjusting the centroids. These steps are repeated until a local minimum is met by optimizing the sum of the squared distance between each object and the centroid [274]. Even though *K-means* is an unsupervised method, there are still hyperparameters that need to be determined for each application, such as a predetermined *k* for the number of clusters that are to be populated [274]. Therefore, this clustering needs to be repeated, and the correct number of clusters needs to be determined using methods from Section 8.3.2.

Traditionally, the Euclidean distance is used to determine to which cluster a data point belongs. However, this metric is unsuitable as a distance metric for time series data since the time dimension is ignored. Shifted time series have a large Euclidean distance and, therefore, do not belong to the same cluster. Furthermore, it has been demonstrated that applying traditional clustering methods, such as *K-means*, directly to spatial-temporal data usually results in severe overfitting and poor performance [157].

Density Peaks

Density Peaks (DP) is the most popular in its class of *density-based* clustering algorithms for time series data [119]. These methods allow for constructing non-spherical clusters, which is impossible in partitional clustering methods, such as *K-means* [214].

Definition 8 *Density-based Clustering considers the density of data points when forming clusters instead of distances. Therefore, cluster centers are defined to be the densest region in a data space [214].*

Cluster centers are determined by calculating the local density ρ_i of each point i as:

$$\rho_i = \sum_j \chi(d_{ij} - d_c) \quad \text{where} \quad \chi(d) = \begin{cases} 1, & d \geqslant 0 \\ 0, & d < 0 \end{cases}$$

d_c denotes the predefined neighbourhood distance and d_{ij} is the distance between i and j, based on some similarity measure [214].

After identifying the cluster centers as the points with the highest local density, neighboring points are assigned to clusters based on the neighborhood distance d_{ij}.

$$\delta_i = \begin{cases} \min(d_{ij}) & \text{if } \exists j \rho_j > \rho_i \\ \min(d_{ik}) & k \in \text{all node otherwise} \end{cases}$$

Other than *K-Means*, DP does not rely on the number of clusters as an input parameter. Since the given parameters define what would be considered a meaningful cluster, this method can independently determine the number of clusters to form from the given parameters. This is advantageous in cases where the number of clusters in the data is unknown, but the idea of what should be considered a meaningful cluster is known. Therefore, finding the correct parameters for DP also relies on domain knowledge and empirical evidence. Since appropriate domain knowledge in a citywide data space might depend on individual sensor data types, this method is not applicable in the current state but is discussed in the conclusion.

Deep Temporal Clustering

As described in Section 8.2 *Deep Temporal Clustering* is an architecture that combines well-known neural network methods with a novel temporal clustering layer, thereby forming a single end-to-end learning framework [157]. This unsupervised learning method uses an autoencoder topology in combination with Long-Short Term Memory (LSTM) and convolutional layers to reduce the dimensionality of the input data and cluster the reduced data [157].

8.3.4 *Model Generation using Artificial Neural Networks*

With the created clusters, sensors deployed in a similar context are now grouped. The service utilizes Artificial Neural Networks (ANN) to highlight characteristics that define these clusters by generating an abstraction model for each cluster. This model is then compared to a generated model based on all sensors, showing the main differences and, therefore, the meaningful characteristics of this cluster.

Artificial Neural Network

ANNs are inspired by human brains, consisting of neurons communicating with each other over a network of electrochemical activity. These networks attempt to emulate the understanding of how the brain works by copying its structure and communication methods to form networks that can be trained to solve complex problems. Similar to the neuron being the most basic unit in the brain, a mathematical model of a neuron (called a perceptron) is the most basic unit in an ANN.

Definition 9 *A **Neural Network** is a computational model consisting of a network of simple mathematical functions called "neurons" [93]. The properties of such a network are determined by its topology and the properties of the neurons it consists of [166].*

In feed-forward neural networks, information flows in one direction. Suppose the neurons in such a network consist of simple perceptrons, which cannot build up a memory of previous inputs. In that case, it can be expected that they do not perform well on time series data, where temporal dependencies are present and crucial to consider. To address this problem, various adaptations of the network are possible.

Long-Short Term Memory

When working with time series, it is important to consider the past when generating a more general model of a set of given time series data. LSTM cells are an example of artificial cells that specialize in processing longer input sequences. Instead of a simple feed-forward activation function, as is present in the perceptron, LSTM cells have an internal recurrence that produces paths where the gradient can flow for long durations, thereby mitigating the vanishing and exploding gradient problems present in Recurrent Neural Networks [93, 109]. The so-called "LSTM cell" enables networks to learn long-term dependencies more easily by incorporating gates controlling the weights fed into the network [93, 109].

Autoencoders

Autoencoders are semi-supervised neural networks that can learn the patterns of multiple time series and can be used to identify deviations [93, 205]. Autoencoders of particular interest for this investigation are the so-called *undercomplete* autoencoders [93]. This kind of network has the unique characteristic that the hidden layer has a lower dimensionality than the input and output layers, forming a bottleneck. During the training process, input data is compressed into ever-shrinking hidden layers, thereby reconstructing an internal representation from which the output is reconstructed by expanding the internal representation to the input dimensionality. The model is thereby forced to prioritize input data and, in so doing, isolate the important properties [93]. This network topology results in anomalies not being present in the internal representation, as they do not conform to the general observed distribution. The success of the training is then measured by calculating the loss between the reconstructed output and the initial input. During this phase, anomalies can be identified since they would not be present in the reconstructed internal representation.

Due to the prioritization of input features, a certain loss is expected in the reconstructed model. The training process can be described as minimizing this loss:

$$L(x, g(f(x)))$$

where L is a loss function penalising $g(f(x))$ for being dissimilar from x [93]. Should the produced model match the input data exactly, the model is overfitted for the input.

Definition 10 ***Overfitting*** *is "the production of an analysis that corresponds too closely or exactly to a particular data set, and may therefore fail to fit additional data or predict future observations reliably" [188].*

An overfitted model has been over-specialized to describe the training data set and does not generalize well to new data that was not part of the training set.

8.4 EVALUATION

While multiple methods have been identified as useful for exploiting encoded spatial-temporal dependencies in time series data to extract contextual characteristics from deployed sensors, this section investigates their effectiveness by analyzing real-world data. Sensor data generated by approximately 600 weather stations spread across Germany was used to evaluate different methods and compare their effectiveness. These weather stations are deployed standardized, resulting in fewer contextual characteristics encoded in the sensors. One of the remaining characteristics is the corresponding climate zones in which the sensors are deployed. This classification is also provided by *German Weather Service* (Deutscher Wetterdienst), the German weather and climate authority, which is then used to control the correct classification.

8.4.1 *Dataset*

A time series dataset containing such dependencies is required to evaluate spatial and temporal dependencies' exploitation effectively. The physical separation between nodes encodes the spatial dependencies, while the concurrent observation of the same phenomenon introduces the temporal dependencies.

Data gathered by approximately 600 weather stations spread across Germany, the locations of which can be seen in Figure 27b, was accumulated to construct a large enough dataset to better determine the effectiveness of various stages of the proposed approach. The resulting 600 time series contained in the dataset consists of the average temperature measured at the location over a week. The wide geographic spread of the sensors covers many different environments, which leads to varying measurements in the dataset. This poses the ideal circumstances for analyzing the usefulness of the approach. Going through the data of each station would be time-consuming and tedious. However, automating this task could result in discovering interesting characteristics that could easily be missed otherwise.

Figure 27a visualizes the measurements from all weather stations over a week. It would be a laborious task to extract possible contextual characteristics that could be present from this overwhelming graph, not to mention the possibility of mistakes due to the human factor. This is the main motivation for investigating modern approaches to extracting this information in an automated fashion.

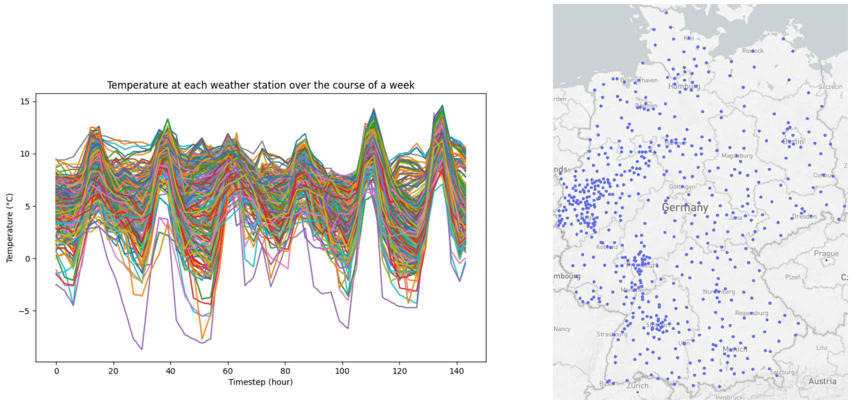
8.4.2 *Outlier Removal*

In 1, anomalies are defined as being sparse in the data and not conforming to the usual pattern of the data in which they are found. Anomalies affecting only a single sensor are unnecessary information and usually result from sensor or network errors. These kinds of anomalies should be removed before constructing a generic model since they could negatively influence the accuracy of clustering and trained models based on the data. However, it would be a mistake to attempt to remove all anomalies. 2 describes contextual characteristics as consisting of a series of anomalies. It would, therefore, be unfortunate if contextual characteristics were removed during the process of outlier removal. In the following section, the removal of point anomalies, which are introduced in Section 8.1.1, are analyzed using Principal Component Analysis (PCA) [93]. In the following, the effectiveness of this method is analyzed when applied to the dataset.

While more than 40 principal components contribute to the variance in the dataset, each component's contribution became significantly smaller. From the results summary shown in Table 7, it can be seen that the initial dimensionality of the input dataset $(144, 588)$

Number of components	Explained variance (%)
2	80
4	90
20	99
39	99.9

Table 7: The percentage of variance which is described by an increasing number of components.



(a) A visualization of temperature measurements over a week at all weather stations belonging to the dataset.

(b) The locations of approximately 600 weather stations spread across Germany

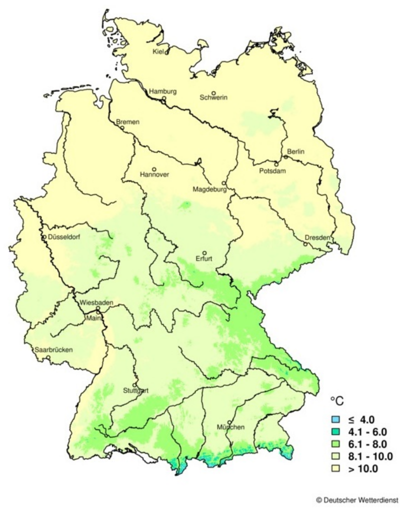
Figure 27: Data set of approximately 600 weather stations deployed in Germany. Average Temperature within a week.

can be reduced to a mere two components while still being able to describe 80% of the variance present in the data, thereby resulting in a $(144, 2)$ dimensional dataset of principal components. However, losing 20% of the variance is unacceptable since this would most probably include almost all the anomalies in the data.

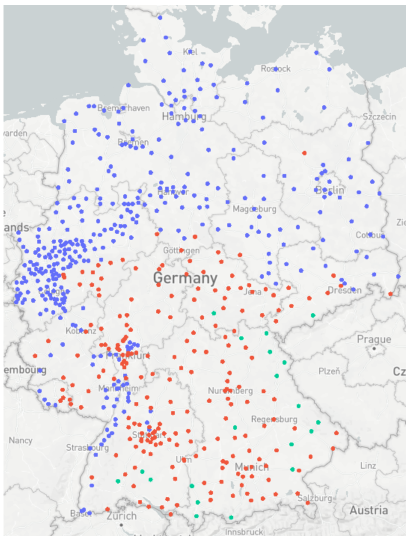
Since a third party produces the dataset, it is unknown whether the data has been processed. For this reason, outlier removal is applied rather conservatively, opting to remove only 0.1% of the variance. When working with raw sensor data, the degree of outlier removal is expected to be much greater. The dataset resulting from the above process is used for the remainder of the investigation.

8.4.3 *Comparing different clustering methods*

Knowing the correct clusters to which each sample should belong is essential for accurately validating the performance of clustering methods. However, since the data is unlabelled and the exact configuration or context in which each station is deployed is unknown, a



(a) A visualization of the average temperatures raster across Germany in October 2021.

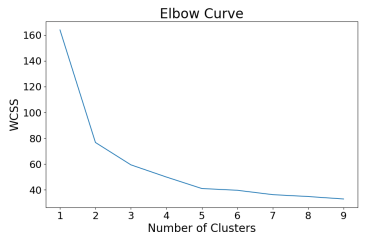


(b) Location of all weather stations colored in their appropriate climate zone.

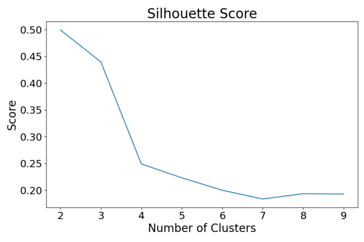
Figure 28: Reference data from the German Weather Service shows that weather stations should be divided into three climate zones.

proxy ground truth is required to perform the evaluation. In this case, one such substitute is the use of the different climate zones across Germany to identify potential clusters that are expected to form. Various factors, including elevation and proximity to large bodies of water, influence each region's climate. It can, therefore, be expected that measurements taken from regions with similar climates should be clustered together since nodes from the same region are sampling similar distributions. Thus, the distances between the time series should be small. A raster grid helps map average temperature measurements across the country over a month to define the expected clusters.

The raster grid is made available in the *ESRI ASCII Grid* format at a resolution of 1km × 1km by the German Weather Service [273]. This format allows for converting from real-world latitudinal and longitudinal coordinates to pixels in the raster grid, thereby finding the average temperatures for each weather station in the dataset. The average temperatures are then divided into belonging to one of five classes, which can be seen in Figure 28a. Figure 28b shows the locations of all weather stations spread across the country, colored according to the climate region they belong to once the mapping was done between the raster grid and the regions shown in Figure 28a. Upon closer inspection, it is noticed that not all five regions are represented in Figure 28b since there is no station in each region. Because the weather stations are spread across only a subset of the climate regions, three clusters are assumed to be present in the data. When clustering the temperature time series, it can be expected that measurements taken



(a) The Elbow Curve shows that three to four clusters are the optimal amount of clusters.



(b) The Silhouette Score highlights that two clusters have the optimal separation, with three clusters close to it.

Figure 29: Comparing both methods, three clusters are selected as the optimal number.

in a region are clustered as belonging to that region. Therefore, the clustering process is expected to produce three clusters, each containing a time series produced by sampling the temperature in a specific climate region. The following compares the performance of the clustering methods discussed in Section 8.3. By performing the clustering based on the various similarity measures mentioned in the same section, the ability of each method to form the expected clusters shown in Figure 28b are compared. The evaluation is split into two tables to better compare the proposed similarity measures. In Table 8, the Euclidean Distance is used as a similarity measure, while DTW is used in Table 9. The three clustering methods are evaluated using each similarity measure. The clustering process is applied multiple times to each method, selecting the best-performing iteration for comparison.

	Precision	Recall	Rand Index	F1-Score	MCC
K-Means	0.76	0.65	0.73	0.71	0.47
DPC	0.71	0.78	0.73	0.74	0.46
DTC	0.73	0.83	0.76	0.78	0.54

Table 8: The performance of different clustering methods with Euclidean Distance as the similarity measurements.

	Precision	Recall	Rand Index	F1-Score	MCC
K-Means	0.75	0.77	0.76	0.76	0.52
DPC	0.73	0.77	0.75	0.75	0.49
DTC	0.80	0.62	0.74	0.70	0.48

Table 9: The performance of different clustering methods with DTW as the similarity measure.

In general, when comparing Table 8 and Table 9, it is clear that the clustering methods performed fairly equally. During the sole consideration of Table 8, where the Euclidean Distance is utilized as a distance metric, Deep Temporal Clustering (DTC) outperformed the other contenders on clustering the dataset in almost all metrics and is, therefore, the only candidate that comes into consideration. However, when including Table 9 to the consideration, where DTW is used as a metric, a reduction in the performance of DTC can be noticed. In comparison, the performance of Density Peak Clustering (DPC) and K-Means has improved and is comparable to the results achieved by DTC using the Euclidean Distance as the distance metric. Table 10 summarises the comparison between the remaining candidates:

	Precision	Recall	Rand Index	F1-Score	MCC
K-Means (DTW)	0.75	0.77	0.76	0.76	0.52
DPC (DTW)	0.73	0.77	0.75	0.75	0.49
DTC (eucl)	0.73	0.83	0.76	0.78	0.54

Table 10: A side-by-side comparison of the performance of the best clustering candidates, irrespective of distance metric.

As can be seen in Table 10, although the achieved results are very similar, DPC performed slightly worse and is therefore eliminated as a candidate, leaving DTC and K-Means as the remaining options. While evaluating these remaining candidates, it was noticed that DPC had much higher computational requirements than K-Means while producing similar results. Since the dataset in this investigation consists of univariate time series, the unnecessarily complex processing done by DTC becomes a burden to itself. In future iterations of this process, where multivariate time series might be investigated, DTC should be reevaluated as a possible clustering candidate. Furthermore, from the research done in Section 8.3, DTW is currently the state of the art when it comes to time series comparison and would, therefore, potentially apply to more kinds of time series than using the Euclidean Distance. For these reasons, K-Means was selected as the clustering method for the remainder of the investigation.

8.4.4 *Cluster Analysis*

The previous section used the three predefined climate zones to represent the "correct" number of clusters present in the data. This was a best-guess assumption, which enabled the analysis of the performance of various clustering algorithms. It is, however, not necessarily the correct number of clusters. In the following, the heuristics discussed in Section 8.3.2 are applied to find the correct number of clus-

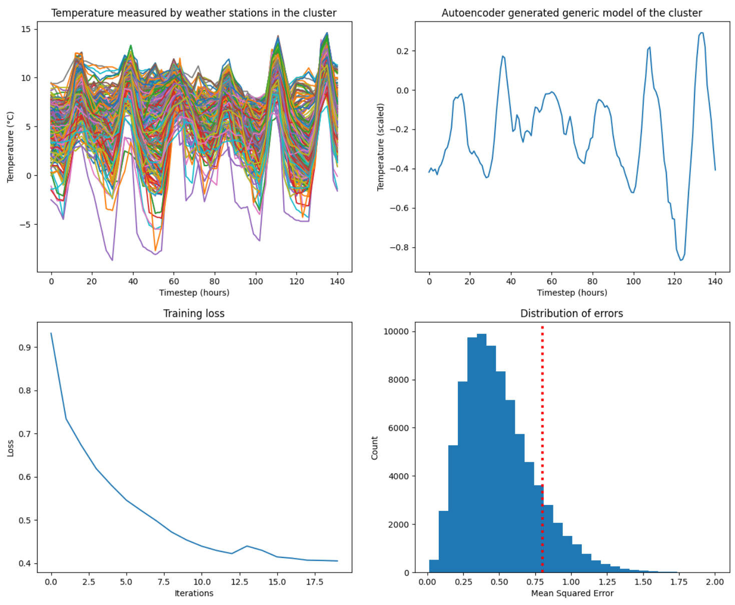


Figure 30: The training process for a generic model of all data in the dataset.

ters. Figure 29a displays the Elbow Curve created by clustering with K-Means with a number from one to nine clusters.

To identify the correct number of clusters using this method, as described in Section 8.3.2, three clusters are identified as the number of clusters present according to this method. From Figure 29a, it can be seen that forming more than three clusters results in a linear reduction in the variance for each additional cluster being added. This is an indication of overfitting taking place.

The Silhouette Score is calculated as a comparison based on the same dataset to support the Elbow Curve's finding. Having understood what is expressed by the Silhouette Score in Figure 8.3.2, the score can be calculated for an increasing number of clusters in the data, similarly to how the Elbow curve was obtained. Since it would be impractical to show the scores of all samples in all clusters to make a decision, instead, the mean Silhouette score over all samples is calculated, as can be seen in Figure 29b.

Figure 29b shows a decreasing mean Silhouette Score as more clusters are added. The highest score is achieved when two clusters are formed. Even though the Silhouette Score suggests two clusters for optimal separation between clusters, the score for three clusters is not much lower. In addition to the Elbow method finding three to be the most suitable number of clusters, as well as this being the number of clusters naively found in Section 8.4.3, forming three clusters would allow for more interesting analyses to be done. For these reasons, three clusters are used for the remainder of this evaluation.

8.4.5 *Autoencoder*

Based on the various architecture components described in 8.3.4, an autoencoder based on LSTM cells can be constructed, which can learn a generic model based on the input time series. A generic model allows for summarising multiple time series into one, simplifying the comparison between time series belonging to a cluster. Since the cluster from which the autoencoder is formed consists of multiple time series, any sequence of anomalies (or contextual characteristics) that occur in multiple time series is encoded into the generic model. This functions as a filter, removing contextual characteristics unique to a single time series and including contextual characteristics present in multiples. A description of the various layers can be seen below:

```
Model: "autoencoder"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 3, 163)]          0
input_layer (LSTM)           (None, 3, 144)            177408
hidden1 (LSTM)               (None, 3, 32)             22656
hidden2 (LSTM)               (None, 1)                 136
bridge (RepeatVector)        (None, 3, 1)              0
hidden3 (LSTM)               (None, 3, 1)              12
hidden4 (LSTM)               (None, 3, 32)             4352
hidden5 (LSTM)               (None, 3, 144)            101952
time_distributed_1 (TimeDis  (None, 3, 163)            23635
tributed)

=================================================================
```

Clear similarities are apparent when the above-listed layers are compared to the generic architecture presented by [157]. The input is compressed into more complex embedded representations as it flows through the network. Once the training has been completed, a lower-dimensional model has been trained, which can be used as a generic model for all the input data into the autoencoder. The dataset was used to construct a generic model of all weather stations across Germany to illustrate the generalization produced by applying an autoencoder.

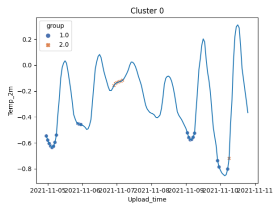
In Figure 30, the performance of the autoencoder training can be seen. The first graph shows the measurements of all weather stations for a week. This is the input to the autoencoder and what is ultimately compressed. The second graph is the generic model that was constructed using the autoencoder. The keen reader would have seen that the temperature measurements have been scaled. The main focus of the investigation is to determine discrepancies between the patterns of cluster models. To determine this, the actual values are irrelevant and have, therefore, been normalized. Additionally, normalizing the values in the time series mitigates the effects of amplitude invariance.

Visually determining the autoencoder's performance is not simple, so two further graphs are added to aid the evaluation. The following graph shows the loss performance during each iteration of the training process. This indicates whether the training was successful and how much loss is present in the generated mode. It also helps determine the number of iterations necessary to train a well-performing model. It is essential to stop the training process before overfitting occurs. When considering the graph, a point can be identified where the rate at which the loss decreases becomes almost linear. This indicates the model is starting to overfit the data, and training should be stopped. Additionally, a second graph visualizes the loss distribution over the input. This distribution helps decide the threshold after which a sample would be considered an anomaly. Since anomalies are sparse and absent in multiple time series, they are not present in the cluster's generic model. Therefore, they can be identified as the samples with a large loss compared to the generic model. As an example, a red line was added to the loss distribution graph to indicate a possible threshold. In this example, all samples with an error larger than 0.8 are identified as anomalies.

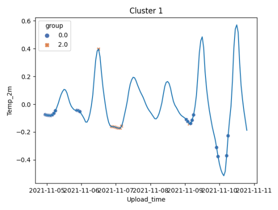
8.4.6 *Identifying Contextual Characteristics*

Having identified the best methods to cluster and create generic models of the time series, these methods can be applied to forming three generic models, one for each cluster formed using K-Means with DTW as the similarity metric. Using the generated models, the focus can be shifted to identifying contextual characteristics. Since each cluster model was generated based on multiple time series belonging to the cluster, the models have encoded contextual characteristics present in multiple time series. The identification of contextual characteristics has thus become trivial due to the time scale being correlated across all clusters. Discrepancies can be identified by comparing the models of each cluster with each other.

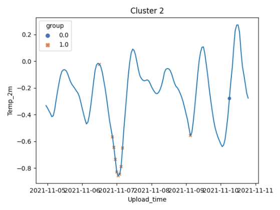
Similar to previous steps, when identifying the contextual characteristics, it is necessary to determine an "error threshold" after which a sequence should be considered a contextual characteristic. Depending on the use case, this likely differs and must be determined empirically. No contextual characteristics are identified if a very large threshold is chosen. Conversely, if a threshold is chosen that is too low, every time step is considered an anomaly or belongs to a contextual characteristic. In the case of the used dataset, a threshold of 0.4 is determined and used. Therefore, any difference between the



(a) Highlighted contextual characteristics within the generic model of the first cluster.



(b) Highlighted contextual characteristics within the generic model of the second cluster.



(c) Highlighted contextual characteristics within the generic model of the third cluster.

Figure 31: Contextual characteristics discovered in each of the clusters.

generic models larger than 0.4 is considered abnormal and belongs to a contextual characteristic. The contextual characteristics identified between the clusters in the dataset are visualized with markers in Figure 31. When calculating the difference between the models, it is unknown which of the models being compared is the source of the contextual characteristic. The automatic detection of this might be analyzed in a future iteration. Therefore, the contextual characteristic is marked in both models. As an additional advantage, marking corresponding contextual characteristics simplifies the comparison between the models since the graph of a single model contains information from others. This approach, however, doesn't scale well and could become problematic if the data contains many clusters.

Several anomalies (or contextual characteristics) could be identified, as shown in Figure 31. All contextual characteristics are recognized due to a discrepancy larger than 0.4 between the cluster graphs. When only considering the discrepancies between Figure 31a and Figure 31b, it can be deduced that during four periods in the week, the difference between Figure 31a and Figure 31b was large enough to be considered anomalous. A clear anomalous event can be seen when comparing Figure 31c to the other two clusters. The third temperature drop is much larger than in the other two clusters. During one period, the temperatures of nodes in cluster two behaved very differently than those in the other clusters.

8.5 SUMMARY

This chapter analyzed various methods applicable to different steps of detecting contextual characteristics based on spatial-temporal dependencies in time series. As a result, a pipeline consisting of the steps identified during this investigation can be constructed, as depicted in Figure 32. Once a time series dataset has been collected containing data with spatial-temporal dependencies, such as data collected by a wireless sensor network, Principal Component Analysis

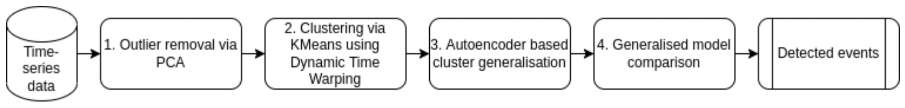

Time-series data
1. Outlier removal via PCA
2. Clustering via KMeans using Dynamic Time Warping
3. Autoencoder based cluster generalisation
4. Generalised model comparison
Detected events

Figure 32: A flow diagram showing the necessary steps identified to detect contextual characteristics based on spatial-temporal dependencies.

(PCA) is applied to the data to remove point anomalies (1). This preprocessing step improves the performance of the clustering step that follows. K-means clustering based on Dynamic Time Warping as a similarity measure was shown to form the best clusters and should, therefore, be applied once the anomalies have been removed (2). The third step exploits the spatial dependencies in the data due to the assumption that sensors deployed in similar contexts are clustered together since they are sampling from a similar distribution. Once the clusters have been formed, they can be generalized using an autoencoder in step four. Since contextual characteristics would have simultaneously affected multiple sensor nodes in that region, the characteristics are encoded into the generalization for that region's cluster. Generalized models are based on samples taken during the same period; therefore, discovering cluster characteristics involves comparing the correlating measurements encoded in the models and returning the identified characteristics.

The evaluation showed that the proposed pipeline can detect an expressive number of clusters within a significant amount of time series data. By analyzing these clusters, the pipeline can identify multiple characteristics within each cluster that separate it from others. While the evaluation used temperature data, the pipeline can be easily applied to different contextual data since no additional context information specific to temporal data was used.

Within the proposed citywide data space, the pipeline is provided as a service that helps data and service providers gain additional information about the characteristics encoded due to the sensor placement or its surroundings. Participating citizens provide their collected data streams to get feedback on their placement (e.g., whether it is exposed to the sun at certain times or shows similar characteristics to others in the surroundings). The service then allows others to request additional characteristic information about a given sensor. When the sensor participates, the requester receives information about characteristics found within the provided data stream. This can help the requesting service decide if a sensor is a viable data source for its functionality.

While the pipeline currently only identifies and highlights characteristics that separate the existing clusters, it provides no information on what caused these characteristics. In the future, a feedback loop for the service participants is planned. This loop lets them add information about the sensors' placement and surroundings, which can then be generalized for the cluster. This information can then be used to learn which characteristics are caused by which event, giving services and participants highlighted characteristics, generalized (expected) value curves, and the corresponding information about the causing event.

Another open question is how many characteristics can be identified within a Smart City context with a much denser sensor distribution. The current evaluation used data streams collected over a week. Data streams collected daily could be investigated to find more fine-granular characteristics. These can then be grouped by the surrounding weather conditions (e.g., sunny, rainy, or freezing) since different behaviors depend on weather conditions. Therefore, different characteristics might be visible within the collected data.

Finally, as mentioned in Section 8.4.3, different methods must be reevaluated when multivariant data is used to identify characteristics.

The following chapter summarizes the presented contributions. It then discusses the results and answers the research questions. The thesis is closed with a discussion of future work and a conclusion.

9 DISCUSSION AND CONCLUSION

After presenting, analyzing, and evaluating the individual contributions in the previous three chapters, this chapter begins with a summary of these contributions embedded in the overarching architecture. It continues addressing and discussing the research questions from the introduction chapter. The thesis concludes with an outlook on future research possibilities.

9.1 CONTRIBUTION SUMMARY

This thesis introduced a comprehensive architecture for a decentralized citizen-centric data space. It is based on distributed, citizen-managed *Connectors*, which act as gateways for the citizens' locally deployed sensors and services. By connecting these Connectors, deployed services can access not only the local reachable sensors but citywide collected data. The following paragraphs summarize the three main contributions.

The first identified research gap focuses on discovering existing data sources within a decentralized data space. To overcome this challenge, this thesis proposes SkABNet (see Chapter 6) as a fully decentral discovery overlay. It allows services to utilize existing data streams by describing relevant criteria. Through a semantic search, SkABNet returns all matching data sources, allowing services to request their data. To evaluate the proposed approach, a simulation of a decentralized network consisting of 40.000, 60.000, and 80.000 nodes was conducted. Results show that the semantic searches require up to 90% less message overhead compared to SkipNet searches. To further improve the efficiency of common search queries, a single sensor is represented multiple times within the network with different semantic descriptors. By simulating a network consisting of 50.000 sensors represented by one to three semantic descriptions (increasing the network size 150.000 nodes) it was shown that the message overhead could be further decreased by 50%.

Second, this thesis addresses the privacy concerns of citizens who act as data providers. Sensor owners are provided with the possibility to set individual access rights for each of their connected sensors. *Private* sensors are not shared with the data space and are only available locally. Collected data from sensors with a *public* setting can be used without limitations. Nonetheless, the owner can see which services use the provided data. As proposed in this thesis, sensor owners can now set their access setting to *protected*. Owners must manually approve each subscription request if a remote service wants to use the provided data. For this, the service provides a description of its function and goal. Further, the service sends a preprocessing pipeline consisting of network-wide predefined steps, which is executed locally on the Connector of the data provider (see Chapter 7). A user study was conducted to evaluate the expressiveness of the preprocessing pipeline and its steps. Within this study, participants were given five example services of increasing complexity. To assess if participants understand the preprocessing and the purpose of the service only based on the preprocessing pipeline, the service description was not provided. Results illustrate that 77% could *correctly* or *rather correctly* describe the data processing undertaken on their data based on the provided step sequence.

Lastly, standardized data measurement cannot be expected since the citizens set up their sensors. An analysis method was introduced to allow service operators to still assess the data quality of available data sources in the network (see Chapter 8), which analyzes the data sources and detects sensor-specific characteristics visible as anomalies within the data streams. The evaluation conducted in this thesis focused on anomaly detection and cluster mechanisms to group data sources affected by the same characteristics. A real-world data set provided by the *German Weather Service* was used to compare and select existing algorithms and create a pipeline suited for environmental data.

Following the summary of contributions, the subsequent section discusses the evaluation results and their interpretation in relation to the questions presented at the beginning.

9.2 DISCUSSION

After summarizing the contributions presented in this thesis, the following section addresses and discusses the research questions posed in the introduction, using the contributions and the resulting findings. Additionally, the scientific and practical applicability of the achieved results and their limitations are highlighted.

The first research question is: **How can modern Smart Cities actively integrate their citizens to become more sustainable?** Sustainability within a Smart City can be viewed from an environmental and social standpoint [110]. While environmental sustainability describes the ecological aspect of the Smart City by optimizing resource consumption and reducing waste, social sustainability focuses on social cohesion and the inclusion of its citizens [40].

Regarding environmental sustainability, this thesis focused on using and integrating existing hardware. The goal is to reduce the ecological footprint and save money economically through the long-term reuse of existing hardware. By providing Connectors as hardware-independent software, participating citizens can use existing hardware to connect their sensors and provide them as data sources in a citywide data space.

While setting up the Connector software in a simple and user-friendly manner remains a continuous challenge that needs constant improvement, participating citizens can engage deeply with their data, its use, and processing through the Connector and the created data space. This data space creates a citywide community that continuously learns and participates in the city's urban growth. Therefore, the proposed approach improves social sustainability. Also, by giving citizens the tools to connect and understand their collected data and create services themselves, a citywide Citizen Science community is built, allowing citizens to take active roles within their surroundings.

To explore the various aspects of the research question, three sub-questions were posed, focusing on integrating existing sensor hardware, ensuring data sovereignty, and including non-technical citizens.

For services and other users to utilize various sensors, it is essential to address *TC1 - Interoperability*. This is also reflected in *RQ 1.1*. To overcome the heterogeneity of existing data sources provided by citizens, a data model was introduced in Chapter 5 that builds on established ontologies. Collected data can then (if desired by the owners) be stored locally for visualization and interpretation or provided as a data stream for the data space. With the help of the provided semantic description of the data sources and their data, the introduced discovery overlay *SkABNet* enables the integration of existing data sources into a decentralized managed data space. While this thesis focused on integrating the collected data from the sensor hardware and making data easily accessible for services, it did not address the connection of individual sensors to the Connector. However, many open standards, interfaces, and communities already delve deeper into this topic, e.g., openHAB [292], HomeAssistant [112], and ioBroker [291].

RQ 1.1: How can data from existing sensor hardware be effectively integrated into a citywide data space?

RQ1.2 also addresses citizens' trust in order to enhance active participation and the resulting social sustainability. In particular, it focuses on data sovereignty as described in *SC1 - Ensuring Data Sovereignty and Privacy of Citizens*, which refers to citizens having control over their data and information shared with other participants, as well as being able to decide how services may use provided data.

RQ1.2: What mechanisms are required to ensure the data sovereignty of citizens and to create an understanding of the use of their data?

First, the proposed Connector enables connection to a citywide data space without automatically and unintentionally sharing collected data or existing metadata with others. For each connected sensor, citizens can decide if information about collected data should be shared with the data space. Due to its fully decentralized architecture *SkABNet*, no other participant can obtain information about the home network or local sensors unless explicitly authorized by the owner. Only by voluntarily providing information through the data space can other citizens and services obtain information, strengthening the citizen's data sovereignty concerning data sharing.

Second, services must individually request data streams and disclose their data usage when sensors are configured to collect protected data. Therefore, a distributed preprocessing framework was developed to show data owners how services use their provided data transparently. The user study results show that for simple services 92% of the participants could identify and describe the correct processing goal. For more complex services this number drops to 61% showing that further improvements need to be made to help citizens identify relevant information. In contrast to identifying the services' goal, preprocessing step description was performed at least *rather correctly* by 84% of participants for the most complex service. This indicates an already understandable visualization of each preprocessing step. Hence, future improvements should focus on a more precise visualization of the preprocessing pipeline results, which helps the citizens understand data usage by more complex services.

Focusing on technical-trained citizens, the evaluation proved they can create custom services by using the existing modular preprocessing steps. This allows them to search for relevant data sources within the data space and generate new knowledge through Citizen Science.

In addition, results from the user study can be further divided into non-technical and technical citizens. The results show that technically skilled citizens still better identify and explain service goals and individual processing steps than non-technical citizens. When focusing on non-technical citizens, 80% could correctly describe the preprocessing goal for the simplest service, dropping down to 40% for the most complex. Hence, non-technical citizens need to be accounted for by a socially sustainable Smart City as addressed in *SC3 - Inclusion of Non-Technically Trained Citizens* and *RQ1.3*. Since only a minimal introduction was given at the beginning of the study, adding help functions and feedback loops to the user interface when investigating the preprocessing pipeline can assist non-technical citizens.

***RQ1.3:** How can non-technically trained citizens be effectively involved in using and managing the citywide data space?*

Further, when working with the Connector, the design focuses on semantic rather than technical descriptions of the sensors to actively integrate technically inexperienced citizens into the data space. While this thesis focused on the usability for data usage, the Connector and its user interfaces must also be user-friendly and require minimal technical experience. Therefore, the usability of the Connector's user interface and data integration should also be investigated in the future. By leveraging existing communities and building a local citywide community, reusable integration modules for the most common interfaces can be created and provided to participants for easier integration. This aspect directly ties into social sustainability. Only through the active participation and promotion of citizens and communities can they be empowered, learn new essential skills, and build trust in the system discussed in *SC2 - Trust and Transparency*.

In summary, utilizing a gateway such as the Connector introduced in this thesis integrates citizens and makes existing sensor hardware accessible to a citywide data space. Using this Connector, citizens can manage their sensors and services that use the data they provide, increasing the city's economic and social sustainability. Moreover, other citizens can access the data provided in this data space and use them as data sources for their services. Through granular privacy settings and transparent usage descriptions by the services, citizens retain data sovereignty. With a straightforward and user-friendly interface, they can understand and evaluate the use of their data even without technical knowledge.

While the first research question focuses on the perspective of data sources, the second research question examines the data space from the perspective of data consumers, i.e., services: **How can services access a citywide data space and utilize the available data while considering data sovereignty?** Here, the Connector again functions as a gateway between the external software, the service, and the data space. Through the service interface (described in Figure 5.2.2), citizens can connect self-managed services. Citizens provide a semantic description of the required data, which is then searched for by the discovery overlay *SkABNet* in the citywide data space.

RQ2.1: How can services efficiently find relevant data sources in the distributed data space and utilize their data?

The first sub-question *RQ 2.1* focuses on data discovery as depicted in *TC2 - Discovery of Sensor Data* and data utilization from a service standpoint, which is discussed in the following. As described in Section 6.4.1, data sources are published in SkABNet with selected attributes from their semantic description as individual nodes. These nodes connect and communicate in the form of a structured peer-to-peer network, enabling semantic searches started by a service to discover all relevant data sources and request their data efficiently. The evaluation conducted using a simulated city with 50,000 sensors illustrates that, depending on the complexity of the search query, the required messages could be reduced by up to 90% compared to SkiPNet. Even in the worst case, when matching nodes are distributed evenly within the overlay network, no more than $O(\log(n))$ messages per searched node are needed, where n is the number of represented sensors in the network.

Given the evaluation, SkABNet proves to be scalable and extensible. It also improves interoperability due to its semantic descriptions of sensors. Additionally, its decentral architecture increases the privacy of individual citizens since no central entity can gain insights into a citizen's network with its connected sensors and given access rights to remote services. This thesis intentionally did not address *Security* within the overlay layer. Fundamentally, there is no access from outside the local network to the local Connector. Settings cannot be issued over the distributed network; these are only possible through the local user interface. However, security should be examined more deeply from an overarching network perspective. It is known that the discovery overlay network can be manipulated by malicious actors, who can deliberately stop messages and searches. While the proposed Network interface uses the transport overlay *drasyl* [33], which ensures end-to-end encryption of messages between Connectors, preventing eavesdropping or alteration, intentional non-forwarding can degrade search performance and skew results. Future research should investigate methods to identify and exclude these harmful participants from the network.

After services have discovered relevant data sources, they must request protected data from other citizens. Within this request, services provide a preprocessing pipeline consisting of modular preprocessing steps that are known within the network. Due to predefined preprocessing steps, Connectors can ensure that the data is preprocessed as described and that no services gain additional information due to malicious preprocessing pipelines, preserving data sovereignty and privacy of the data sources. While these predefined steps ensure that collected data is unified and easier to process for the service, this approach comes with two drawbacks.

First, these preprocessing steps need to be modular and reusable to enable all sorts of services. Currently, no mechanism is included that allows for the proposal of new preprocessing steps to be added to an established data space. Second, while the local execution prevents malicious actions from services, it does not protect services from malicious data sources. This thesis did not investigate the possibility of a malicious data source deliberately manipulating or ignoring local preprocessing steps to send false results to the service. Since a service expects data in a specific format due to the defined preprocessing pipeline, manipulating the data could alter the service's results and crash the service (depending on its implementation) due to incompatible data formats. Although appropriate logs and debugging analysis can identify and exclude malicious data sources in the future, this represents a significant additional effort for the service provider. Therefore, future research should investigate methods to ensure the correct execution of the distributed preprocessing pipeline and guarantee at least syntactically correct data.

RQ2.2: What mechanisms are necessary to evaluate data quality in a decentralized system without violating the data sovereignty of citizens?

Another critical challenge for services is to determine the data quality of its data sources, since using citizen-operated sensors, service providers can not rely on standardized placed sensors [16, 21, 122]. This challenge is targeted in the final research question *RQ2.2* While existing work often focuses on outlier removal in the examined data, this thesis considers data quality based on influencing factors. Using the method presented in Chapter 8, collected data streams are compared, and contained anomalies are uncovered. If an anomaly occurs not only in a single sensor but in a group of sensors, it no longer represents a local anomaly. Instead, it describes a characteristic of the influenced sensors. Evidently, this anomaly is caused by an external phenomenon measured by several sensors, meaning these share a common characteristic in their placement and data measurement.

As no central instance in the presented data space can access all sensors and their data, the method is implemented as a separate service that must adhere to the same rules as all other services. This data quality service requires the complete data streams, i.e., the raw data of the sensors. Meaning that the decentralized preprocessing pipeline needs unfiltered data from the examined sensor. To maintain the data sovereignty of the citizens, data source owners can decide whether the service can analyze their data sources. To motivate data owners to share their data, quality services also need to benefit them, e.g., by making them aware of possible incorrect placement of the sensors. Nevertheless, participation in the quality service is voluntary and remains at the discretion of the data owners. For other services, the quality service provides the opportunity to view the characteristics of the data source and decide if it is suitable for their provided service. With the help of this service, *TC3 - Data Quality* is addressed.

For example, a weather station might only be shaded at certain times, leading to significant temperature differences during those periods.

As described in Section 8.5, the developed data quality service consists of a sequence of steps that extract and highlight the characteristics. Data quality cannot be directly assessed since there is no ground truth through standardized sensors. Instead, a set of data streams is compared, and their differences are analyzed. This means multiple data streams must be interpreted to generate a result. Differences between the individual data streams are then highlighted as characteristics. If a characteristic is recognized in several data streams, it is no longer assumed to be an outlier but an external influence causing this characteristic. This work did not address the semantic description of the individual characteristics, but it can be achieved through a feedback loop with the data quality service users. As highlighted, the data quality service focuses on maintaining data sovereignty, achieved through voluntary participation motivated by the service's benefits.

In summary, the network interface from the Connector, based on the SkABNet network, allows services to discover available data sources. Since each citizen defines which data sources are discoverable, SkABNet adheres to the citizen's data sovereignty. Further, by defining a transparent preprocessing pipeline run on the citizen's Connector, raw data from private or protected sensors can never be accessed. While data quality is still one of the most significant challenges for privately owned sensors, in a decentral data space, no central entity can guarantee or measure the data quality for individual sensors. Here, data quality services should also benefit citizens collecting data so that they are motivated to share their data and let a remote service analyze collected data.

9.3 FUTURE WORK

After discussing the research questions, this section presents future research topics addressing identified limitations. Improvements are grouped into two main categories: first, open topics that contribute to a finished product usable by citizens, and second, technical improvements that enhance the system's extensibility and enable continuous adaptation to new circumstances.

One important open question is how the proposed data space can be actively integrated into citizen's daily lives. While a user study was already conducted in Chapter 7 to examine the extent to which citizens without technical knowledge can understand preprocessing steps and thus assess their data usage, other areas must also be analyzed with the help of a more in-depth user study. The connection between the Connector and local sensors was assumed to be given in this thesis, but this must be simplified for participating citizens. Only in this way can enough citizens be motivated to contribute their data to a citywide data space. It must be investigated how existing sensors can be easily integrated through existing standards.

Another premise of this work was the voluntary and willing participation of the citizens. According to Matschke et al., various environmental (e.g., platform usability, time and effort requirements, quantity of existing content), personal (internal motivation, tool competence), interpersonal, and socio-cultural factors are the primary motivations for information exchange [165]. However, this motivation usually remains only in an open, egalitarian social network. Once participating services and thus individual participants earn money from the data provided by others, the willingness to share voluntarily shifts, as the value of one's own data is now assessed [24, 53]. Therefore, an in-depth examination of the willingness to share data, especially regarding the value of one's own data in an urban and citizen science context, would be particularly interesting. It would also be worth investigating how this willingness shifts once local industry professionalizes and monetizes such a network.

Regarding the distributed preprocessing framework proposed in Chapter 7, the future work contains improving the user interface by highlighting changes and combining smaller steps into recipes with meaningful explanations, enhancing understanding of data preprocessing. An extensive user study may lead to further user interface changes and simplifications. To make the system more adaptive, adding new preprocessing steps at runtime must be investigated, requiring all Connectors to know these steps. Using a distributed consensus algorithm, new preprocessing steps and recipes could be published and agreed upon within the network, providing more flexibility and enabling additional services.

Lastly, an overarching improvement in the citizen-operated data space is the integration of user feedback loops. Through a wide range of feedback loops, the system can continuously adapt to new requirements and provide citizens with a better experience handling their data and services. The introduced service for categorizing data quality could, for example, use the provided raw data and the resulting characteristics to inform owners about specific details of sensor placement. This knowledge can then provide feedback to other sensors with similar characteristics, thereby optimizing sensor placement over time and improving the overall data quality.

Regarding technical improvements, the focus is on making the data space more adaptable to changing usage behavior and increasing quality by decreasing human errors when providing data.

In the future, research should investigate the mechanisms by which the decentralized data space can adapt to new circumstances. This includes, on the one hand, changes in the most common searches within the discovery network, requiring different identifier compositions to respond to these queries efficiently. On the other hand, new preprocessing steps are anticipated to be needed, which must then be known to all Connectors. Due to the absence of a central authority, changes cannot be enforced across all Connectors. Instead, the consortium of Connectors must agree on new compositions and preprocessing steps, which are then applied to all participating Connectors. Existing consensus algorithms can be studied and their applicability evaluated for this purpose.

Overall, technical tools should be employed to verify citizen inputs in the Connector and assess their validity based on local knowledge. This can help to avoid a wide range of human errors or misconfigurations, ultimately improving data quality.

9.4 CONCLUSION

In conclusion, this thesis contributes to the vision of a sustainable and citizen-centered Smart City by addressing both the technical and social challenges of implementing a decentralized data space. By shifting from a technology-driven to a citizen-centered approach, this work enhances participatory rights and promotes active engagement with urban spaces.

The proposed SkABNet overlay network significantly improves the efficiency of discovering relevant sensor data in a decentralized environment, demonstrating up to 90% reduction in message overhead. This solution not only optimizes data retrieval but also promotes the reuse of existing citizen-operated sensors, minimizing the need for additional sensor deployment.

Further, the data sovereignty framework introduced in this work allows citizens to retain control over the preprocessing and distribution of their sensor data, ensuring that even non-technical users can make informed decisions about how their data is utilized. The accompanying user study validated the effectiveness of this approach, with the majority of participants demonstrating an understanding of the data processing steps involved.

Lastly, developing a data categorization method facilitates the assessment of data quality in an environment where sensor placement and configuration may vary. By detecting anomalies and clustering data streams with shared characteristics, this method enables service providers to evaluate available data sources' reliability better while helping citizens improve sensor placement for more accurate data collection.

Together, these contributions represent a significant step toward creating a more inclusive and transparent Smart City framework where citizens actively participate in collecting and managing urban data. Future research may build upon these findings by exploring further technical improvements and refining the social aspects of citizen participation, ensuring that Smart Cities are not only technologically advanced but also deeply aligned with the needs and rights of their inhabitants.

PUBLICATIONS

Some ideas and figures have appeared previously in the following publications:

[31] Heiko Bornholdt, David Jost, Philipp Kisters, Michel Rottleuthner, Dirk Bade, Winfried Lamersdorf, Thomas C Schmidt, and Mathias Fischer. "SANE: Smart networks for urban citizen participation." In: *26th International Conference on Telecommunications (ICT)*. 2019.

[32] Heiko Bornholdt, David Jost, Philipp Kisters, Michel Rottleuthner, Sehrish Shafeeq, Winfried Lamersdorf, Thomas Schmidt, and Mathias Fischer. "Smart Urban Data Space for Citizen Science." In: *Electronic Communications of the EASST* (2021).

[136] Philipp Kisters, Dirk Bade, and Julius Wulk. "Dynamic routing using precipitation data." In: *4th International Conference on Fog and Mobile Edge Computing, FMEC* (2019).

[137] Philipp Kisters, Heiko Bornholdt, and Janick Edinger. "SkAB-Net: A Data Structure for Efficient Discovery of Streaming Data for IoT." In: *32nd International Conference on Computer Communications and Networks (ICCCN)*. 2023.

[138] Philipp Kisters, Vinh Ngu, and Janick Edinger. "Urban Heat Island Detection Utilizing Citizen Science." In: *European Conference on Service-Oriented and Cloud Computing*. 2022.

[139] Philipp Kisters, Hanno Schreiber, and Janick Edinger. "Categorization of crowd-sensing streaming data for contextual characteristic detection." In: *Journal of Smart Cities and Society* (2023).

[140] Philipp Kisters, Leonie v. d. Veen, and Janick Edinger. "Privacy-Preserving Edge Processing in Decentralized Citizen-Centric Sensor Networks." In: *International Symposium on Intelligent and Distributed Computing*. 2023.

BIBLIOGRAPHY

[1] Mohammad Aazam, Marc St-Hilaire, Chung-Horng Lung, and Ioannis Lambadaris. “Cloud-based smart waste management for smart cities.” In: *IEEE 21st International Workshop on Computer Aided Modelling and Design of Communication Links and Networks (CAMAD)*. 2016.

[2] Nibras Abdullah, Ola A Alwesabi, and Rosni Abdullah. “IoT-based smart waste management system in a smart city.” In: *Recent Trends in Data Science and Soft Computing: Proceedings of the 3rd International Conference of Reliable Information and Communication Technology*. 2019.

[3] Mohammad Abu Alsheikh, Shaowei Lin, Dusit Niyato, and Hwee Pink Tan. “Machine Learning in Wireless Sensor Networks: Algorithms, Strategies, and Applications.” In: *IEEE Communications Surveys and Tutorials* (2014).

[4] Kofi Sarpong Adu-Manu, Cristiano Tapparello, Wendi Heinzelman, Ferdinand Apietu Katsriku, and Jamal-Deen Abdulai. “Water Quality Monitoring Using Wireless Sensor Networks: Current Trends and Future Research Directions.” In: *ACM Transactions on Sensor Networks* (2017).

[5] Charu C. Aggarwal. *Managing and Mining Sensor Data*. Ed. by Charu C. Aggarwal. Springer, 2013.

[6] Jose Aguilar, Marxjhony Jerez, Maribel Mendonça, and Manuel Sánchez. “Performance analysis of the ubiquitous and emergent properties of an autonomic reflective middleware for smart cities.” In: *Computing* (2020).

[7] Nurzhan Zhumabekuly Aitzhan and Davor Svetinovic. “Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams.” In: *IEEE Transactions on Dependable and Secure Computing* (2016).

[8] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and E. Cayirci. “A Survey on Sensor Networks.” In: *IEEE Wireless Communications* (2002).

[9] A.R. Al-Ali, Imran A. Zualkernan, Mohammed Rashid, Ragini Gupta, and Mazin Alikarar. “A smart home energy management system using IoT and big data analytics approach.” In: *IEEE Transactions on Consumer Electronics* (2017).

[10] Eiman Al Nuaimi, Hind Al Neyadi, Nader Mohamed, and Jameela Al-Jaroodi. "Applications of big data to smart cities." In: *Journal of Internet Services and Applications* (2015).

[11] Vito Albino, Umberto Berardi, and Rosa Maria Dangelico. "Smart cities: Definitions, dimensions, performance, and initiatives." In: *Journal of Urban Technology* (2015).

[12] Mohammed Aljoufie and Alok Tiwari. "Citizen sensors for smart city planning and traffic management: crowdsourcing geospatial data through smartphones in Jeddah, Saudi Arabia." In: *GeoJournal* (2022).

[13] Sascha Alpers, Stefanie Betz, Andreas Fritsch, Andreas Oberweis, Gunther Schiefer, and Manuela Wagner. "Citizen empowerment by a technical approach for privacy enforcement." In: *Proceedings of the 8th International Conference on Cloud Computing and Services Science*. 2018.

[14] Ruhul Amin, Neeraj Kumar, GP Biswas, Rahat Iqbal, and Victor Chang. "A light weight authentication protocol for IoT-enabled devices in distributed Cloud Computing environment." In: *Future Generation Computer Systems* (2018).

[15] David L Anderson, Frank F Britt, and Donavon J Favre. "The 7 principles of supply chain management." In: *Supply Chain Management Review* (2007).

[16] Christine Anhalt-Depies, Jennifer L Stenglein, Benjamin Zuckerberg, Philip A Townsend, and Adena R Rissman. "Tradeoffs and tools for data quality, privacy, transparency, and trust in citizen science." In: *Biological Conservation* (2019).

[17] Leonidas G Anthopoulos. *Understanding smart cities: a tool for smart government or an industrial trick?* Springer, 2017.

[18] Wolfgang Apolinarski, Umer Iqbal, and Josiane Xavier Parreira. "The GAMBAS middleware and SDK for smart city applications." In: *IEEE International conference on pervasive computing and communication workshops (PERCOM WORKSHOPS)*. 2014.

[19] James Aspnes and Gauri Shah. "Skip graphs." In: *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms* (2003).

[20] Claudio Badii, Pierfrancesco Bellini, Angelo Difino, and Paolo Nesi. "Smart City IoT Platform Respecting GDPR Privacy and Security Aspects." In: *IEEE Access* (2020).

[21] Bálint Balázs, Peter Mooney, Eva Nováková, Lucy Bastin, Jamal Jokar Arsanjani, and others. "Data quality in citizen science." In: *The science of citizen science* (2021).

[22] Ryohei Banno and Kazuyuki Shudo. “An Efficient Routing Method for Range Queries in Skip Graph.” In: *IEICE Transactions on Information and Systems* (2020).

[23] Viviana Bastidas, Markus Helfert, and Marija Bezbradica. “A Requirements Framework for the Design of Smart City Reference Architectures.” In: *Proceedings of the 51st Hawaii international conference on system sciences*. 2018.

[24] Ahmed Saleh Bataineh, Rabeb Mizouni, May El Barachi, and Jamal Bentahar. “Monetizing personal data: a two-sided market approach.” In: *Procedia Computer Science* (2016).

[25] Gustavo Batista, Xiaoyue Wang, and Eamonn Keogh. “A Complexity-Invariant Distance Measure for Time Series.” In: *Proceedings of the 11th SIAM International Conference on Data Mining*. 2011.

[26] Martin Bauer, Erno Kovacs, Anett Schulke, Naoko Ito, Carmen Criminisi, Laurent-Walter Goix, and Massimo Valla. “The Context API in the OMA Next Generation Service Interface.” In: *14th International Conference on Intelligence in Next Generation Networks*. 2010.

[27] Lisa Benton-Short and John Rennie Short. *Cities and Nature*. Routledge, 2013.

[28] Donald J Berndt and James Clifford. “Using dynamic time warping to find patterns in time series.” In: *Proceedings of the 3rd international conference on knowledge discovery and data mining*. 1994.

[29] Franco Bianchini. “Remaking European cities: the role of cultural policies.” In: *Cultural policy and urban regeneration: The West European experience* (1993).

[30] Dario Bonino, Maria Teresa Delgado Alizo, Claudio Pastrone, and Maurizio Spirito. “WasteApp: Smarter waste recycling for smart citizens.” In: *International Multidisciplinary Conference on Computer and Energy Science (SpliTech)*. 2016.

[31] Heiko Bornholdt, David Jost, Philipp Kisters, Michel Rottleuthner, Dirk Bade, Winfried Lamersdorf, Thomas C Schmidt, and Mathias Fischer. “SANE: Smart networks for urban citizen participation.” In: *26th International Conference on Telecommunications (ICT)*. 2019.

[32] Heiko Bornholdt, David Jost, Philipp Kisters, Michel Rottleuthner, Sehrish Shafeeq, Winfried Lamersdorf, Thomas Schmidt, and Mathias Fischer. “Smart Urban Data Space for Citizen Science.” In: *Electronic Communications of the EASST* (2021).

[33] Heiko Bornholdt, Kevin Röbert, and Philipp Kisters. "Accessing smart city services in untrustworthy environments via decentralized privacy-preserving overlay networks." In: *IEEE International Conference on Service-Oriented System Engineering (SOSE)*. 2021.

[34] H. L. Brantley, G. S. W. Hagler, E. S. Kimbrough, R. W. Williams, S. Mukerjee, and L. M. Neas. "Mobile air monitoring data-processing strategies and effects on spatial air pollution trends." In: *Atmospheric Measurement Techniques* (2014).

[35] Lutz Breuer, Noreen Hiery, Philipp Kraft, Martin Bach, Alice H Aubert, and Hans-Georg Frede. "HydroCrowd: a citizen science snapshot to assess the spatial control of nitrogen solutes in surface waters." In: *Scientific reports* (2015).

[36] Alessandro Brighente, Mauro Conti, Gabriele Di Renzone, Giacomo Peruzzi, and Alessandro Pozzebon. "Security and Privacy of Smart Waste Management Systems: A Cyber-Physical System Perspective." In: *IEEE Internet of Things Journal* (2023).

[37] Tim Campbell. "Learning cities: Knowledge, capacity and competitiveness." In: *Habitat International* (2009).

[38] Keyan Cao, Yefan Liu, Gongjie Meng, and Qimeng Sun. "An Overview on Edge Computing Research." In: *IEEE Access* (2020).

[39] Andrea Caragliu and Chiara Del Bo. "Smartness and European urban performance: assessing the local impacts of smart urban attributes." In: *Innovation: The European Journal of Social Science Research* (2012).

[40] Michael Carley, Paul Jenkins, and Harry Smith, eds. *Urban Development and Civil Society: The Role of Communities in Sustainable Cities*. Earthscan, 2001.

[41] Pablo Chamoso, Alfonso González-Briones, Fernando De La Prieta, Ganesh Kumar Venyagamoorthy, and Juan M Corchado. "Smart city as a distributed platform: Toward a system for citizen-oriented management." In: *Computer communications* (2020).

[42] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly Detection: A Survey." In: *ACM Comput. Surv.* (2009).

[43] Heejun Chang, Sarah Praskievicz, and Hossein Parandvash. "Sensitivity of urban water consumption to weather and climate variability at multiple temporal scales: The case of Portland, Oregon." In: *International Journal of Geospatial and Environmental Research* (2014).

[44] Chee-Yee Chong and S.P. Kumar. "Sensor networks: Evolution, opportunities, and challenges." In: *Proceedings of the IEEE* (2003).

[45] Min Chen, Shiwen Mao, and Yunhao Liu. "Big data: A survey." In: *Mobile networks and applications* (2014).

[46] Yiheng Chen and Dawei Han. "Water quality monitoring in smart city: A pilot project." In: *Automation in Construction* (2018).

[47] Bin Cheng, Salvatore Longo, Flavio Cirillo, Martin Bauer, and Ernoe Kovacs. "Building a Big Data Platform for Smart Cities: Experience and Lessons from Santander." In: *IEEE International Congress on Big Data*. 2015.

[48] Bin Cheng, Gurkan Solmaz, Flavio Cirillo, Erno Kovacs, Kazuyuki Terasawa, and Atsushi Kitazawa. "FogFlow: Easy Programming of IoT Services Over Cloud and Edges for Smart Cities." In: *IEEE Internet of Things Journal* (2018).

[49] Hongju Cheng, Zhe Xie, Leihuo Wu, Zhiyong Yu, and Ruixing Li. "Data prediction model in wireless sensor networks based on bidirectional LSTM." In: *EURASIP Journal on Wireless Communications and Networking* (2019).

[50] Mohamed Cherradi and Anass EL Haddadi. "Data lakes: A survey paper." In: *The Proceedings of the International Conference on Smart City Applications*. 2021.

[51] Carla-Fabiana Chiasserini, Imrich Chlamtac, Paolo Monti, and Antonio Nucci. "Energy efficient design of wireless ad hoc networks." In: *Second International IFIP-TC6 Networking Conference*. 2002.

[52] Elisa Chioatto, Muhammad Attiq Khan, and Paolo Sospiro. "Sustainable solid waste management in the European Union: Four countries regional analysis." In: *Sustainable Chemistry and Pharmacy* (2023).

[53] Jay Pil Choi, Doh-Shin Jeon, and Byung-Cheol Kim. "Privacy and personal data collection with information externalities." In: *Journal of Public Economics* (2019).

[54] Hafedh Chourabi, Taewoo Nam, Shawn Walker, J. Ramon Gil-Garcia, Sehl Mellouli, Karine Nahon, Theresa A. Pardo, and Hans Jochen Scholl. "Understanding Smart Cities: An Integrative Framework." In: *45th Hawaii International Conference on System Sciences*. 2012.

[55] Simone Cirani, Luca Davoli, Gianluigi Ferrari, Remy Leone, Paolo Medagliani, Marco Picone, and Luca Veltri. "A Scalable and Self-Configuring Architecture for Service Discovery in the Internet of Things." In: *IEEE Internet of Things Journal* (2014).

[56] Simone Cirani and Luca Veltri. "Implementation of a framework for a DHT-based Distributed Location Service." In: *16th International Conference on Software, Telecommunications and Computer Networks*. 2008.

[57] Flavio Cirillo, Gürkan Solmaz, Everton Luís Berz, Martin Bauer, Bin Cheng, and Ernoe Kovacs. "A Standard-based Open Source IoT Platform: FIWARE." In: *IEEE Internet of Things Magazine* (2019).

[58] Annalisa Cocchia. "Smart and Digital City: A Systematic Literature Review." In: *Smart city: How to create public and economic value with high technology in urban space* (2014).

[59] Amanda Coe, Gilles Paquet, and Jeffrey Roy. "E-Governance and Smart Communities: A Social Learning Challenge." In: *Social Science Computer Review* (2001).

[60] Balazs Csanad Csaji, Zsolt Kemeny, Gianfranco Pedone, Andras Kuti, and Jozsef Vancza. "Wireless Multi-Sensor Networks for Smart Cities: A Prototype System With Statistical Data Analysis." In: *IEEE Sensors Journal* (2017).

[61] David Culler, Deborah Estrin, and Mani Srivastava. "Overview of sensor networks." In: *Computer* (2004).

[62] Thiago Pereira Da Silva, Thais Batista, Frederico Lopes, Aluizio Rocha Neto, Flávia C. Delicato, Paulo F. Pires, and Atslands R. Da Rocha. "Fog Computing Platforms for Smart City Applications: A Survey." In: *ACM Transactions on Internet Technology* (2022).

[63] Houda Daki, Asmaa El Hannani, Abdelhak Aqqal, Abdelfattah Haidine, and Aziz Dahbi. "Big Data management in smart grid: concepts, requirements and implementation." In: *Journal of Big Data* (2017).

[64] Maya Daneva and Boyan Lazarov. "Requirements for smart cities: Results from a systematic review of literature." In: *12th International Conference on Research Challenges in Information Science (RCIS)*. 2018.

[65] Sabrina De Capitani Di Vimercati, Sara Foresti, Giovanni Livraga, and Pierangela Samarati. "DATA PRIVACY: DEFINITIONS AND TECHNIQUES." In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* (2012).

[66] Corrado De Fabritiis, Roberto Ragona, and Gaetano Valenti. "Traffic Estimation And Prediction Based On Real Time Floating Car Data." In: *11th International IEEE Conference on Intelligent Transportation Systems*. 2008.

[67] Martin De Jong, Simon Joss, Daan Schraven, Changjie Zhan, and Margot Weijnen. "Sustainable–smart–resilient–low carbon–eco–knowledge cities; making sense of a multitude of concepts promoting sustainable urbanization." In: *Journal of Cleaner Production* (2015).

[68] Yuri Demchenko, Cees De Laat, and Peter Membrey. "Defining architecture components of the Big Data Ecosystem." In: *2014 International conference on collaboration technologies and systems (CTS)*. IEEE. 2014.

[69] Swati Dhingra, Rajasekhara Babu Madda, Amir H. Gandomi, Rizwan Patan, and Mahmoud Daneshmand. "Internet of Things Mobile–Air Pollution Monitoring System (IoT-Mobair)." In: *IEEE Internet of Things Journal* (2019).

[70] Dan Ding, Rory A Cooper, Paul F Pasquina, and Lavinia Fici-Pasquina. "Sensor technology for smart homes." In: *Maturitas* (2011).

[71] James Dixon. *Pentaho, Hadoop, and Data Lakes*. en. Oct. 2010. URL: https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/ (visited on 10/16/2023).

[72] Bassirou Diène, Joel J.P.C. Rodrigues, Ousmane Diallo, El Hadji Malick Ndoye, and Valery V. Korotaev. "Data management techniques for Internet of Things." In: *Mechanical Systems and Signal Processing* (2020).

[73] Manish Dwivedi, Aniruddha Uniyal, and Rajiva Mohan. "New horizons in planning smart cities using LiDAR technology." In: *International Journal of Applied Remote Sensing and GIS* (2015).

[74] European Economic and Committee Of The Regions Social Committee. *European strategy for data*. Tech. rep. European Commision, 2020.

[75] Leif Edvinsson. "Aspects on the city as a knowledge tool." In: *Journal of knowledge management* (2006).

[76] M V Eitzel et al. "Citizen Science Terminology Matters: Exploring Key Terms." In: *Citizen Science: Theory and Practice* (2017).

[77] European Environment Agency. *Europe's air quality status 2024 — European Environment Agency*. en. Briefing. URL: https://www.eea.europa.eu/publications/europes-air-quality-status-2024 (visited on 06/11/2024).

[78] FOAF Project. *Friend of a Friend (FOAF) Ontology*. 2014. URL: http://xmlns.com/foaf/spec/ (visited on 06/11/2024).

[79] Christos Faloutsos, Mudumbai Ranganathan, and Yannis Manolopoulos. "Fast subsequence matching in time-series databases." In: *ACM Sigmod Record* (1994).

[80] Asmaa Fawzy, Hoda M.O. Mokhtar, and Osman Hegazy. "Outliers detection and classification in wireless sensor networks." In: *Egyptian Informatics Journal* (2013).

[81] Mohamed Anis Fekih, Walid Bechkit, Herve Rivano, Manoel Dahan, Florent Renard, Lucille Alonso, and Florent Pineau. "Participatory Air Quality and Urban Heat Islands Monitoring System." In: *IEEE Transactions on Instrumentation and Measurement* (2021).

[82] Luca Filipponi, Andrea Vitaletti, Giada Landi, Vincenzo Memeo, Giorgio Laura, and Paolo Pucci. "Smart City: An Event Driven Architecture for Monitoring Public Spaces with Heterogeneous Sensors." In: *Fourth International Conference on Sensor Technologies and Applications*. 2010.

[83] Michael Franklin, Alon Halevy, and David Maier. "From databases to dataspaces: a new abstraction for information management." In: *ACM SIGMOD Record* (2005).

[84] Jennifer Gabrys. "Programming Environments: Environmentality and Citizen Sensing in the Smart City." In: *Environment and Planning D: Society and Space* (2014).

[85] Arman Ganji, Omid Youssefi, Junshi Xu, Keni Mallinen, Marshall Lloyd, An Wang, Ardevan Bakhtari, Scott Weichenthal, and Marianne Hatzopoulou. "Design, calibration, and testing of a mobile sensor system for air pollution and built environment data collection: The urban scanner platform." In: *Environmental Pollution* (2023).

[86] Stephen R. Gardner. "Building the data warehouse." In: *Communications of the ACM* (1998).

[87] André Gensler and Bernhard Sick. "Performing Event Detection in Time Series with SwiftEvent: An Algorithm with Supervised Learning of Detection Criteria." In: *Pattern Anal. Appl.* (2018).

[88] Ammar Gharaibeh, Mohammad A. Salahuddin, Sayed Jahed Hussini, Abdallah Khreishah, Issa Khalil, Mohsen Guizani, and Ala Al-Fuqaha. "Smart Cities: A Survey on Data Management, Security, and Enabling Technologies." In: *IEEE Communications Surveys & Tutorials* (2017).

[89] Corinna Giebler, Christoph Gröger, Eva Hoos, Holger Schwarz, and Bernhard Mitschang. "Leveraging the Data Lake: Current State and Challenges." In: *Big Data Analytics and Knowledge Discovery*. 2019.

[90] Rudolf Giffinger and Gudrun Haindl. "Smart cities ranking: an effective instrument for the positioning of cities?" In: *5th International Conference Virtual City and Territory, Barcelona, 2,3 and 4 June 2009*. 2009.

[91] Amy Glasmeier and Susan Christopherson. "Thinking about smart cities." In: *Cambridge Journal of Regions, Economy and Society* (2015).

[92] Ran Gong, Ligang Xu, Deguan Wang, Hongyi Li, and Jin Xu. "Water quality modeling for a typical urban lake based on the EFDC model." In: *Environmental Modeling & Assessment* (2016).

[93] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[94] Christos Goumopoulos. "Smart City Middleware: A Survey and a Conceptual Framework." In: *IEEE Access* (2024).

[95] Rudolf Griffinger and Gudrun Haindlmaier. "SMART CITIES RANKING: AN EFFECTIVE INSTRUMENT FOR THE POSITIONING OF CITIES?" In: *City and Environment* (2010).

[96] Verónica Gutiérrez, Jose A. Galache, Luis Sánchez, Luis Muñoz, Jose M. Hernández-Muñoz, Joao Fernandes, and Mirko Presser. "SmartSantander: Internet of Things Research and Innovation through Citizen Participation." In: *The Future Internet*. 2013.

[97] Rihan Hai, Christos Koutras, Christoph Quix, and Matthias Jarke. "Data Lakes: A Survey of Functions and Systems." In: *IEEE Transactions on Knowledge and Data Engineering* (2023).

[98] Alon Halevy, Michael Franklin, and David Maier. "Principles of dataspace systems." In: *Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. 2006.

[99] Peter Hall. "Creative Cities and Economic Development." In: *Urban Studies* (2000).

[100] Robert E Hall, B Bowerman, J Braverman, J Taylor, H Todosow, and U Von Wimmersperg. *The vision of a smart city*. Tech. rep. Brookhaven National Lab.(BNL), Upton, NY (United States), 2000.

[101] Marcus Handte, Stefan Foell, Gregor Schiele, Umer Iqbal, Wolfgang Apolinarski, Josiane Xavier Parreira, Pedro Marrón, and Gerd Kortuem. "The GAMBAS Middleware for Smart City Applications." In: *Internet of Things Success Stories* (2014).

[102] Steve Hankey, Greg Lindsey, Xize Wang, Jason Borah, Kristopher Hoff, Brad Utecht, and Zhiyi Xu. "Estimating use of non-motorized infrastructure: Models of bicycle and pedestrian traffic in Minneapolis, MN." In: *Landscape and Urban Planning* (2012).

[103] Peter Hanna and Erik Swartling. "Anomaly Detection in Time Series Data using Unsupervised Machine Learning Methods: A Clustering-Based Approach." PhD thesis. KTH Royal Institute of Technology, 2020. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-273630 (visited on 06/11/2024).

[104] A. K. M. Bahalul Haque, Bharat Bhushan, and Gaurav Dhiman. "Conceptualizing smart city applications: Requirements, architecture, security issues, and emerging trends." In: *Expert Systems* (2022).

[105] C. Harrison, B. Eckman, R. Hamilton, P. Hartswick, J. Kalagnanam, J. Paraszczak, and P. Williams. "Foundations for Smarter Cities." In: *IBM Journal of Research and Development* (2010).

[106] Nicholas JA Harvey, Michael B Jones, Stefan Saroiu, Marvin Theimer, and Alec Wolman. "Skipnet: A Scalable Overlay Network with Practical Locality Properties." In: *Proceedings of the \ldots* (2003).

[107] Ibrahim Abaker Targio Hashem, Victor Chang, Nor Badrul Anuar, Kayode Adewole, Ibrar Yaqoob, Abdullah Gani, Ejaz Ahmed, and Haruna Chiroma. "The role of big data in smart city." In: *International Journal of Information Management* (2016).

[108] Asmaa F Hassan, Hoda MO Mokhtar, and Osman Hegazy. "A heuristic approach for sensor network outlier detection." In: *Int J Res Rev Wirel Sens Netw (IJRRWSN)* (2011).

[109] Sepp Hochreiter. "Untersuchungen zu dynamischen neuronalen Netzen." In: *Diploma, Technische Universität München* (1991).

[110] Robert G. Hollands. "Will the real smart city please stand up? Intelligent, progressive or entrepreneurial?" In: *City* (2008).

[111] Robert G. Hollands. "Critical interventions into the corporate smart city." In: *Cambridge Journal of Regions, Economy and Society* (2015).

[112] *HomeAssistant*. URL: https://www.home-assistant.io/ (visited on 09/24/2024).

[113] Félix Iglesias and Wolfgang Kastner. "Analysis of similarity measures in times series clustering for the discovery of building energy patterns." In: *Energies* (2013).

[114] Yoshimasa Ishi, Yuuichi Teranishi, Mikio Yoshida, Susumu Takeuchi, Shinji Shimojo, and Shojiro Nishio. "Range-Key Extension of the Skip Graph." In: *IEEE Global Telecommunications Conference GLOBECOM 2010*. 2010.

[115] Elvira Ismagilova, Laurie Hughes, Yogesh K Dwivedi, and K Ravi Raman. "Smart cities: Advances in research—An information systems perspective." In: *International journal of information management* (2019).

[116] Matthias Jarke and Christoph Quix. "On Warehouses, Lakes, and Spaces: The Changing Role of Conceptual Modeling for Data Integration." In: *Conceptual Modeling Perspectives* (2017).

[117] Sabeen Javaid, Ali Sufian, Saima Pervaiz, and Mehak Tanveer. "Smart traffic management system using Internet of Things." In: *20th international conference on advanced communication technology (ICACT)*. 2018.

[118] Abdul Rehman Javed, Faisal Shahzad, Saif ur Rehman, Yousaf Bin Zikria, Imran Razzak, Zunera Jalil, and Guandong Xu. "Future smart cities: Requirements, emerging technologies, applications, challenges, and future aspects." In: *Cities* (2022).

[119] Ali Javed, Byung Suk Lee, and Donna M. Rizzo. "A benchmark study on time series clustering." In: *Machine Learning with Applications* (2020).

[120] Hyesoo Jeon and Changjun Lee. "Internet of Things Technology: Balancing privacy concerns with convenience." In: *Telematics and Informatics* (2022).

[121] Peng Jiang, Hongbo Xia, Zhiye He, and Zheming Wang. "Design of a Water Environment Monitoring System Based on Wireless Sensor Networks." In: *Sensors* (2009).

[122] Wan Jiao et al. *Community Air Sensor Network (CAIRSENSE) project: Evaluation of low-cost sensor performance in a suburban environment in the southeastern United States*. Tech. rep. Aerosols/In Situ Measurement/Instruments and Platforms, 2016. URL: https://amt.copernicus.org/preprints/amt-2016-131/amt-2016-131.pdf.

[123] Ashlee Jollymore, Morgan J Haines, Terre Satterfield, and Mark S Johnson. "Citizen science for water quality monitoring: Data implications of citizen perspectives." In: *Journal of environmental management* (2017).

[124] Simon Joss, Robert Cowley, and Daniel Tomozeiu. "Towards the 'ubiquitous eco-city': An analysis of the internationalisation of eco-city policy and practice." In: *Urban Research & Practice* (2013).

[125] George Kakarontzas, Leonidas Anthopoulos, Despoina Chatzakou, and Athena Vakali. "A Conceptual Enterprise Architecture Framework for Smart Cities - A Survey Based Approach:" in: *Proceedings of the 11th International Conference on e-Business*. 2014.

[126] Andrew Karvonen. *Politics of urban runoff: nature, technology, and the sustainable city*. MIT Press, 2011.

[127] B.S. Kerner, C. Demir, R.G. Herrtwich, S.L. Klenov, H. Rehborn, M. Aleksi, and A. Haug. "Traffic state detection with floating car data in road networks." In: *IEEE Intelligent Transportation Systems, 2005*. 2005.

[128] Wazir Zada Khan, Ejaz Ahmed, Saqib Hakak, Ibrar Yaqoob, and Arif Ahmed. "Edge Computing: A Survey." In: *Future Generation Computer Systems* (2019).

[129] Zaheer Khan, Ashiq Anjum, and Saad Liaquat Kiani. "Cloud Based Big Data Analytics for Smart Future Cities." In: *IEEE/ACM 6th International Conference on Utility and Cloud Computing*. 2013.

[130] Zaheer Khan and Saad Liaquat Kiani. "A Cloud-based architecture for citizen services in smart cities." In: *Proceedings - 2012 IEEE/ACM 5th International Conference on Utility and Cloud Computing, UCC 2012* (2012).

[131] Zaheer Khan, David Ludlow, Richard McClatchey, and Ashiq Anjum. "An architecture for integrated intelligence in urban management using cloud computing." In: *Journal of Cloud Computing: Advances, Systems and Applications* (2012).

[132] Rida Khatoun and Sherali Zeadally. "Smart cities: concepts, architectures, research opportunities." In: *Communications of the ACM* (2016).

[133] In-suk Kim, Yong-hyeog Kang, and Young Ik Eom. "An Efficient Contents Discovery Mechanism in Pure P2P Environments." In: *International Conference on Grid and Cooperative Computing*. 2004.

[134] Ralph Kimball and Margy Ross. *The data warehouse toolkit: the complete guide to dimensional modeling*. John Wiley Sons, 2011.

[135] Ayca Kirimtat, Ondrej Krejcar, Attila Kertesz, and M. Fatih Tasgetiren. "Future Trends and Current State of Smart City Concepts: A Survey." In: *IEEE Access* (2020).

[136] Philipp Kisters, Dirk Bade, and Julius Wulk. "Dynamic routing using precipitation data." In: *4th International Conference on Fog and Mobile Edge Computing, FMEC* (2019).

[137] Philipp Kisters, Heiko Bornholdt, and Janick Edinger. "SkAB-Net: A Data Structure for Efficient Discovery of Streaming Data for IoT." In: *32nd International Conference on Computer Communications and Networks (ICCCN)*. 2023.

[138] Philipp Kisters, Vinh Ngu, and Janick Edinger. "Urban Heat Island Detection Utilizing Citizen Science." In: *European Conference on Service-Oriented and Cloud Computing*. 2022.

[139] Philipp Kisters, Hanno Schreiber, and Janick Edinger. "Categorization of crowd-sensing streaming data for contextual characteristic detection." In: *Journal of Smart Cities and Society* (2023).

[140] Philipp Kisters, Leonie v. d. Veen, and Janick Edinger. "Privacy-Preserving Edge Processing in Decentralized Citizen-Centric Sensor Networks." In: *International Symposium on Intelligent and Distributed Computing*. 2023.

[141] Rob Kitchin. "The real-time city? Big data and smart urbanism." In: *GeoJournal* (2014).

[142] Sylvia Knight, Claire Smith, and Michael Roberts. "Mapping Manchester's urban heat island." In: *Weather* (2010).

[143] Jahoon Koo and Young-Gab Kim. "Interoperability requirements for a smart city." In: *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. 2021.

[144] Andreas Krause, Eric Horvitz, Aman Kansal, and Feng Zhao. "Toward Community Sensing." In: *International Conference on Information Processing in Sensor Networks (ipsn 2008)*. 2008.

[145] Alexandr Krylovskiy, Marco Jahn, and Edoardo Patti. "Designing a smart city internet of things platform with microservice architecture." In: *3rd international conference on future internet of things and cloud*. 2015.

[146] Md Abdul Kuddus, Elizabeth Tynan, and Emma McBryde. "Urbanization: a problem for the rich and the poor?" In: *Public Health Reviews* (2020).

[147] Billy Pik Lik Lau, Sumudu Hasala Marakkalage, Yuren Zhou, Naveed Ul Hassan, Chau Yuen, Meng Zhang, and U-Xuan Tan. "A survey of data fusion in smart city applications." In: *Information Fusion* (2019).

[148] Juan Li, Nazia Zaman, and Honghui Li. "A Decentralized Locality-Preserving Context-Aware Service Discovery Framework for Internet of Things." In: *Proceedings - 2015 IEEE International Conference on Services Computing, SCC 2015* (2015).

[149] Xiang-Yang Li, Peng-Jun Wan, and Ophir Frieder. "Coverage in wireless ad hoc sensor networks." In: *IEEE Transactions on computers* (2003).

[150] Sengboon Lim, Jalaluddin Abdul Malek, Mohd Yusof Hussain, and Zurinah Tahir. "Citizen participation in building citizen-centric smart cities." In: *Malaysian Journal of Society and Space* (2018).

[151] Jun Liu, Zhenyu Huang, Ming Fan, Jihui Yang, Jie Xiao, and Yong Wang. "Future energy infrastructure, energy platform and energy storage." In: *Nano Energy* (2022).

[152] Weidang Lu, Yi Gong, Xin Liu, Jiaying Wu, and Hong Peng. "Collaborative Energy and Information Transfer in Green Wireless Sensor Networks for Smart Cities." In: *IEEE Transactions on Industrial Informatics* (2018).

[153] Tie Luo and Sai G. Nagarajan. “Distributed Anomaly Detection Using Autoencoder Neural Networks in WSN for IoT.” In: *IEEE International Conference on Communications (ICC)*. 2018.

[154] Qin Lv, Pei Cao, Edith Cohen, Kai Li, and Scott Shenker. “Search and replication in unstructured peer-to-peer networks.” In: *Proceedings of the 16th international conference on Supercomputing*. 2002.

[155] Arthur M. Del Esposte, Fabio Kon, Fabio M. Costa, and Nelson Lago. “InterSCity: A Scalable Microservice-based Open Source Platform for Smart Cities:” in: *Proceedings of the 6th International Conference on Smart Cities and Green ICT Systems*. Porto, Portugal: SCITEPRESS - Science and Technology Publications, 2017.

[156] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. “l-diversity: Privacy beyond k-anonymity.” In: *Acm transactions on knowledge discovery from data (tkdd)* (2007).

[157] Naveen Sai Madiraju, Seid M. Sadat, Dimitry Fisher, and Homa Karimabadi. “Deep Temporal Clustering : Fully Unsupervised Learning of Time-Domain Features.” In: *CoRR* (2018). arXiv: 1802.01059.

[158] Mobasshir Mahbub. “A smart farming concept based on smart embedded electronics, internet of things and wireless sensor network.” In: *Internet of Things* (2020).

[159] Mozhgan Malekshahi Rad, Amir Masoud Rahmani, Amir Sahafi, and Nooruldeen Nasih Qader. “Social Internet of Things: vision, challenges, and trends.” In: *Human-centric Computing and Information Sciences* (2020).

[160] Lydia Manikonda, Aditya Deotale, and Subbarao Kambhampati. “What’s up with privacy? User preferences and privacy concerns in intelligent personal assistants.” In: *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. 2018.

[161] Samer Mansour, Nidal Nasser, Lutful Karim, and Asmaa Ali. “Wireless Sensor Network-based air quality monitoring system.” In: *International Conference on Computing, Networking and Communications (ICNC)*. 2014.

[162] Hug March. “The Smart City and other ICT-led technoimaginaries: Any room for dialogue with Degrowth?” In: *Journal of Cleaner Production* (2018).

[163] Chris J. Martin, James Evans, and Andrew Karvonen. “Smart and sustainable? Five tensions in the visions and practices of the smart-sustainable city in Europe and North America.” In: *Technological Forecasting and Social Change* (2018).

[164] Kirk Martinez, Jane K Hart, and Royan Ong. “Environmental sensor networks.” In: *Computer* (2004).

[165] Christina Matschke, Johannes Moskaliuk, Franziska Bokhorst, Till Schümmer, and Ulrike Cress. “Motivational factors of information exchange in social information spaces.” In: *Computers in Human Behavior* (2014).

[166] PhD Researcher Matthew Stewart. *Comprehensive introduction to autoencoders*. 2020. URL: https://towardsdatascience.com/generating-images-with-autoencoders-77fd3a8dd368 (visited on 06/11/2024).

[167] Eoghan McKenna, Ian Richardson, and Murray Thomson. “Smart meter data: Balancing consumer privacy concerns with legitimate applications.” In: *Energy Policy* (2012).

[168] Alexey Medvedev, Petr Fedchenkov, Arkady Zaslavsky, Theodoros Anagnostopoulos, and Sergey Khoruzhnikov. “Waste Management as an IoT-Enabled Service in Smart Cities.” In: *Internet of Things, Smart Spaces, and Next Generation Networks and Systems: 15th International Conference*. 2015.

[169] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M.B. Srivastava. “Coverage problems in wireless ad-hoc sensor networks.” In: *IEEE INFOCOM. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society*. 2001.

[170] Seapahn Meguerdichian, Farinaz Koushanfar, Gang Qu, and Miodrag Potkonjak. “Exposure in wireless Ad-Hoc sensor networks.” In: *Proceedings of the 7th annual international conference on Mobile computing and networking*. 2001.

[171] Yemeserach Mekonnen, Lamar Burton, Arif Sarwat, and Shekhar Bhansali. “IoT Sensor Network Approach for Smart Farming: An Application in Food, Energy and Water System.” In: *IEEE Global Humanitarian Technology Conference (GHTC)*. 2018.

[172] Syed Misbahuddin, Junaid Ahmed Zubairi, Abdulrahman Saggaf, Jihad Basuni, Sulaiman A-Wadany, and Ahmed Al-Sofi. “IoT based dynamic road traffic management for smart cities.” In: *12th International Conference on High-capacity Optical Networks and Enabling/Emerging Technologies (HONET)*. 2015.

[173] Subhankar Mishra, Sudhansu Mohan Satpathy, and Abhipsa Mishra. “Energy Efficiency in ad hoc Networks.” In: *International Journal of Ad hoc, Sensor & Ubiquitous Computing* (2011).

[174] Nader Mohamed, Jameela Al-Jaroodi, Imad Jawhar, Sanja Lazarova-Molnar, and Sara Mahmoud. “SmartCityWare: A Service-Oriented Middleware for Cloud and Fog Enabled Smart City Services.” In: *IEEE Access* (2017).

[175] Saraju P Mohanty, Uma Choppali, and Elias Kougianos. "Everything you wanted to know about smart cities: The Internet of things is the backbone." In: *IEEE consumer electronics magazine* (2016).

[176] Andres Monzon. "Smart Cities Concept and Challenges: Bases for the Assessment of Smart City Projects." In: *Smart Cities, Green Technologies, and Intelligent Transport Systems*. 2015.

[177] Rosario Morello, Subhas C. Mukhopadhyay, Zheng Liu, Daniel Slomovitz, and Subhransu Ranjan Samantaray. "Advances on Sensing Technologies for Smart Cities and Power Grids: A Review." In: *IEEE Sensors Journal* (2017).

[178] María V. Moreno, Miguel A. Zamora, and Antonio F. Skarmeta. "User-centric smart buildings for energy sustainable smart cities." In: *Transactions on Emerging Telecommunications Technologies* (2014).

[179] Thomas Morstyn, Niall Farrell, Sarah J Darby, and Malcolm D McCulloch. "Using peer-to-peer energy-trading platforms to incentivize prosumers to form federated power plants." In: *Nature energy* (2018).

[180] Vaia Moustaka, Athena Vakali, and Leonidas G. Anthopoulos. "A Systematic Review for Smart City Data Analytics." In: *ACM Computing Surveys* (2019).

[181] Meinard Müller. "Dynamic Time Warping." In: *Information retrieval for music and motion* (2007).

[182] Taewoo Nam and Theresa A. Pardo. "Conceptualizing smart city with dimensions of technology, people, and institutions." In: *Proceedings of the 12th Annual International Digital Government Research Conference: Digital Government Innovation in Challenging Times*. 2011.

[183] Athira Nambiar and Divyansh Mundra. "An Overview of Data Warehouse and Data Lake in Modern Enterprise Data Management." In: *Big Data and Cognitive Computing* (2022).

[184] Fatemeh Nargesian, Erkang Zhu, Renée J. Miller, Ken Q. Pu, and Patricia C. Arocena. "Data lake management: challenges and opportunities." In: *Proceedings of the VLDB Endowment* (2019).

[185] Peter Newman. "The environmental impact of cities." In: *Environment and Urbanization* (2006).

[186] D. Niculescu. "Positioning in ad hoc sensor networks." In: *IEEE Network* (2004).

[187] Colin O'Reilly, Alexander Gluhak, Muhammad Ali Imran, and Sutharshan Rajasegarar. "Anomaly Detection in Wireless Sensor Networks in a Non-Stationary Environment." In: *IEEE Communications Surveys Tutorials* (2014).

[188] *OVERFITTING: Definition of overfitting by Oxford dictionary on LEXICO.COM also meaning of overfitting*. URL: https://www.lexico.com/definition/overfitting (visited on 06/11/2024).

[189] Hasan Omar. "Intelligent Traffic Information System Based on Integration of Internet of Things and Agent Technology." In: *International Journal of Advanced Computer Science and Applications* (2015).

[190] Ahmed M Shahat Osman. "A novel big data analytics framework for smart cities." In: *Future Generation Computer Systems* (2019).

[191] Tousif Osman, Shahreen Shahjahan Psyche, J. M. Shafi Ferdous, and Hasan U. Zaman. "Intelligent traffic management system for cross section of roads using computer vision." In: *IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*. 2017.

[192] Federica Paganelli and David Parlanti. "A DHT-based discovery service for the internet of things." In: *Journal of Computer Networks and Communications* (2012).

[193] Kellow Pardini, Joel JPC Rodrigues, Ousmane Diallo, Ashok Kumar Das, Victor Hugo C de Albuquerque, and Sergei A Kozlov. "A smart waste management solution geared towards citizens." In: *Sensors* (2020).

[194] Edoardo Patti and Andrea Acquaviva. "IoT platform for Smart Cities: Requirements and implementation case studies." In: *IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*. 2016.

[195] Vincenzo Pavone and Sara Degli Esposti. "Public assessment of new surveillance-oriented security technologies: Beyond the trade-off between privacy and security." In: *Public Understanding of Science* (2012).

[196] Charith Perera, Yongrui Qin, Julio C. Estrella, Stephan Reiff-Marganiec, and Athanasios V. Vasilakos. "Fog Computing for Sustainable Smart Cities: A Survey." In: *ACM Computing Surveys* (2017).

[197] Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. "Sensing as a service model for smart cities supported by Internet of Things." In: *Transactions on Emerging Telecommunications Technologies* (2014).

[198] Riccardo Petrolo, Valeria Loscrì, and Nathalie Mitton. "Towards a smart city based on cloud of things, a survey on the smart city vision and paradigms." In: *Transactions on Emerging Telecommunications Technologies* (2017).

[199] Marco Picone, Michele Amoretti, and Francesco Zanichelli. "GeoKad: A P2P distributed localization protocol." In: *8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*. 2010.

[200] Thomas Piketty. *Capital in the twenty-first century*. Harvard University Press, 2014.

[201] Stephanie Pincetl and Elizabeth Gearin. "The Reinvention of Public Green Space." In: *Urban Geography* (2005).

[202] Antonios Pliatsios, Dimitrios Lymperis, and Christos Goumopoulos. "S2NetM: A Semantic Social Network of Things Middleware for Developing Smart and Collaborative IoT-Based Solutions." In: *Future Internet* (2023).

[203] Michael E Porter and Victor E Millar. "How information gives you competitive advantage: The information revolution is transforming the nature of competition." In: *Knowledge and special libraries*. Routledge, 2009.

[204] O.A. Postolache, J.M.D. Pereira, and P.M.B.S. Girao. "Smart Sensors Network for Air Quality Monitoring Applications." In: *IEEE Transactions on Instrumentation and Measurement* (2009).

[205] Oleksandr I. Provotar, Yaroslav M. Linder, and Maksym M. Veres. "Unsupervised Anomaly Detection in Time Series Using LSTM-Based Autoencoders." In: *IEEE International Conference on Advanced Trends in Information Theory (ATIT)*. 2019.

[206] William Pugh. "Skip lists: A probabilistic alternative to balanced trees." In: *Lecture s in Computer Science (including subseries Lecture s in Artificial Intelligence and Lecture s in Bioinformatics)* (1989).

[207] Dan Puiu et al. "CityPulse: Large Scale Data Analytics Framework for Smart Cities." In: *IEEE Access* (2016).

[208] Mompoloki Pule, Abid Yahya, and Joseph Chuma. "Wireless sensor networks: A survey on monitoring water quality." In: *Journal of applied research and technology* (2017).

[209] QUDT. *Quantities, Units, Dimensions, and Data Types (QUDT)*. 2014. URL: http://www.qudt.org/ (visited on 06/11/2024).

[210] Stefan Rass, Simone Fuchs, Martin Schaffer, and Kyandoghere Kyamakya. "How to protect privacy in floating car data systems." In: *Proceedings of the fifth ACM international workshop on VehiculAr Inter-NETworking*. 2008.

[211] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Schenker. "A scalable content-addressable network." In: *Computer Communication Review* (2001).

[212] Mohammad Abdur Razzaque, Marija Milojevic-Jevric, Andrei Palade, and Siobhan Clarke. "Middleware for Internet of Things: A Survey." In: *IEEE Internet of Things Journal* (2016).

[213] Matei Ripeanu. "Peer-to-peer architecture case study: Gnutella network." In: *Proceedings first international conference on peer-to-peer computing*. 2001.

[214] Alex Rodriguez and Alessandro Laio. "Clustering by fast search and find of density peaks." In: *science* (2014).

[215] MS Roopa, Santosh Pattar, Rajkumar Buyya, Kuppanna Rajuk Venugopal, SS Iyengar, and LM Patnaik. "Social Internet of Things (SIoT): Foundations, thrust areas, systematic review and future directions." In: *Computer Communications* (2019).

[216] Peter J. Rousseeuw. "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis." In: *Journal of Computational and Applied Mathematics* (1987).

[217] Antony Rowstron, Peter Druschel, and Rachid Guerraoui. "Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems." In: *Middleware 2001* (2001).

[218] Philip W. Rundel, Eric A. Graham, Michael F. Allen, Jason C. Fisher, and Thomas C. Harmon. "Environmental sensor networks in ecological research." In: *New Phytologist* (2009).

[219] F Sabry and E Ali. "A survey of real-time data warehouse and ETL." In: *Int. J. Sci. Eng. Res* (2014).

[220] Stan Salvador and Philip Chan. "Toward accurate dynamic time warping in linear time and space." In: *Intelligent Data Analysis* (2007).

[221] Luis Sanchez et al. "SmartSantander: IoT experimentation over a smart city testbed." In: *Computer Networks* (2014).

[222] Eduardo Felipe Zambom Santana, Ana Paula Chaves, Marco Aurelio Gerosa, Fabio Kon, and Dejan S. Milojicic. "Software Platforms for Smart Cities: Concepts, Requirements, Challenges, and a Unified Reference Architecture." In: *ACM Computing Surveys* (2018).

[223] Ricardo Jorge Santos and Jorge Bernardino. "Real-time data warehouse loading methodology." In: *Proceedings of the 2008 international symposium on Database engineering & applications - IDEAS '08*. 2008.

[224] Mahadev Satyanarayanan. "The Emergence of Edge Computing." In: *Computer* (2017).

[225] Hans Schaffers, Nicos Komninos, Marc Pallot, Brigitte Trousse, Michael Nilsson, and Alvaro Oliveira. *Smart cities and the future internet: Towards cooperation frameworks for open innovation*. Springer Berlin Heidelberg, 2011.

[226] Ina Schieferdecker, Nikolay Tcholtchev, and Philipp Lämmel. "Urban Data Platforms: An Overview." In: *Proceedings of the 12th International Symposium on Open Collaboration Companion*. 2016.

[227] Nauman Shahid, Ijaz Naqvi, and Saad Qaisar. "Characteristics and classification of outlier detection techniques for wireless sensor networks in harsh environments: A survey." In: *Artificial Intelligence Review* (2012).

[228] Taylor Shelton, Matthew Zook, and Alan Wiig. "The 'actually existing smart city'." In: *Cambridge Journal of Regions, Economy and Society* (2015).

[229] Weisong Shi and Schahram Dustdar. "The Promise of Edge Computing." In: *Computer* (2016).

[230] Gopal Kirshna Shyam, Sunilkumar S Manvi, and Priyanka Bharti. "Smart Waste Management using Internet-of-Things (IoT)." In: (2017).

[231] Navjot Sidhu, Alberto Pons-Buttazzo, Andrés Muñoz, and Fernando Terroso-Saenz. "A collaborative application for assisting the management of household plastic waste through smart bins: a case of study in the Philippines." In: *Sensors* (2021).

[232] Welington M da Silva, Alexandre Alvaro, Gustavo HRP Tomas, Ricardo A Afonso, Kelvin L Dias, and Vinicius C Garcia. "Smart cities software architectures: a survey." In: *Proceedings of the 28th annual ACM symposium on applied computing*. 2013.

[233] Anthony Simonofski, Estefania Serral Asensio, Johannes De Smedt, and Monique Snoeck. "Citizen Participation in Smart Cities: Evaluation Framework Proposal." In: *IEEE 19th Conference on Business Informatics (CBI)*. 2017.

[234] Anthony Simonofski, Estefanía Serral Asensio, and Yves Wautelet. "Citizen participation in the design of smart cities." In: *Smart Cities: Issues and Challenges*. Elsevier, 2019.

[235] Sofia Zavialova. *Smart Home: market data & analysis*. Market Insights Report. Statista, 2023.

[236] Amandeep Singh Sohal, Rajinder Sandhu, Sandeep K. Sood, and Victor Chang. "A Cybersecurity Framework to Identify Malicious Edge Device in Fog Computing and Cloud-of-Things Environments." In: *Computers & Security* (2018).

[237] John Soldatos et al. "OpenIoT: Open Source Internet-of-Things in the Cloud." In: *Interoperability and Open-Source Solutions for the Internet of Things: International Workshop*. 2015.

[238] Christoph Stach, Clémentine Gritti, and Bernhard Mitschang. "Bringing privacy control back to citizens: DISPEL—a distributed privacy management platform for the internet of things." In: *Proceedings of the 35th Annual ACM Symposium on Applied Computing*. 2020.

[239] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications." In: (2001).

[240] Salvatore J. Stolfo, Malek Ben Salem, and Angelos D. Keromytis. "Fog Computing: Mitigating Insider Data Theft Attacks in the Cloud." In: *IEEE Symposium on Security and Privacy Workshops*. 2012.

[241] Yeran Sun and Amin Mobasheri. "Utilizing crowdsourced data for studies of cycling and air pollution exposure: A case study using strava data." In: *International journal of environmental research and public health* (2017).

[242] Yeran Sun, Yashar Moshfeghi, and Zhang Liu. "Exploiting crowdsourced geographic information and GIS for assessment of air pollution exposure during active travel." In: *Journal of Transport & Health* (2017).

[243] G. Arturo S´nchez-Azofeifa, Cassidy Rankine, Mario Marcos Do Espirito Santo, Rob Fatland, and Milton Garcia. "Wireless Sensing Networks for Environmental Monitoring: Two Case Studies from Tropical Forests." In: *IEEE Seventh International Conference on eScience*. 2011.

[244] Ruben Sánchez-Corcuera, Adrián Nuñez-Marcos, Jesus Sesma-Solance, Aritz Bilbao-Jayo, Rubén Mulero, Unai Zulaika, Gorka Azkune, and Aitor Almeida. "Smart cities survey: Technologies, application domains and challenges for the cities of the future." In: *International Journal of Distributed Sensor Networks* (2019).

[245] Jihane Tadili and Hakima Fasly. "Citizen participation in smart cities: a survey." In: *Proceedings of the 4th International Conference on Smart City Applications*. 2019.

[246] Abeer Iftikhar Tahirkheli, Muhammad Shiraz, Bashir Hayat, Muhammad Idrees, Ahthasham Sajid, Rahat Ullah, Nasir Ayub, and Ki-Il Kim. "A survey on modern cloud computing security over smart city networks: Threats, vulnerabilities, consequences, countermeasures, and challenges." In: *Electronics* (2021).

[247] Romain Tavenard et al. "Tslearn, A Machine Learning Toolkit for Time Series Data." In: *Journal of Machine Learning Research* (2020).

[248] Nick Taylor Buck and Aidan While. "Competitive urbanism and the limits to smart city innovation: The UK Future Cities initiative." In: *Urban studies* (2017).

[249] Yuuichi Teranishi, Ryohei Banno, and Toyokazu Akiyama. "Scalable and locality-aware distributed topic-based pub/sub messaging for IoT." In: *2015 IEEE Global Communications Conference, GLOBECOM 2015* (2015).

[250] Evangelos Theodoridis, Georgios Mylonas, Veronica Gutiérrez, and Luis Muñoz. "Large-Scale Participatory Sensing Experimentation Using Smartphones within a Smart City." In: *Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. 2014.

[251] Alexandre Tisserant, Stefan Pauliuk, Stefano Merciai, Jannick Schmidt, Jacob Fry, Richard Wood, and Arnold Tukker. "Solid Waste and the Circular Economy: A Global Analysis of Waste Treatment and Waste Footprints." In: *Journal of Industrial Ecology* (2017).

[252] Zhao Tong, Feng Ye, Ming Yan, Hong Liu, and Sunitha Basodi. "A survey on algorithms for intelligent computing and smart city applications." In: *Big Data Mining and Analytics* (2021).

[253] Gregory Trencher. "Towards the smart city 2.0: Empirical evidence of using smartness as a tool for tackling social challenges." In: *Technological Forecasting and Social Change* (2019).

[254] M. Tubaishat and S. Madria. "Sensor networks: an overview." In: *IEEE Potentials* (2003).

[255] United Nations. "World Urbanization Prospects The 2018 Revision." In: *UN* (2018).

[256] S Uttara, Nishi Bhuvandas, and Vanita Aggarwal. "Impacts of urbanization on environment." In: *Applied Sciences* (2012).

[257] Liesbet Van Zoonen. "Privacy concerns in smart cities." In: *Government Information Quarterly* (2016).

[258] András Varga and Rudolf Hornig. "An Overview of the OMNeT++ Simulation Environment." In: *Proceedings of the First International ICST Conference on Simulation Tools and Techniques for Communications Networks and Systems*. 2008.

[259] Laura Veldkamp. "Valuing data as an asset." In: *Review of Finance* (2023).

[260] Jenni Viitanen and Richard Kingston. "Smart Cities and Green Growth: Outsourcing Democratic and Environmental Resilience to the Global Technology Sector." In: *Environment and Planning A: Economy and Space* (2014).

[261] Ignasi Vilajosana, Jordi Llosa, Borja Martinez, Marc Domingo-Prieto, Albert Angles, and Xavier Vilajosana. "Bootstrapping smart cities through a self-sustainable model based on big data flows." In: *IEEE Communications Magazine* (2013).

[262] Félix J Villanueva, Maria J Santofimia, David Villa, Jesús Barba, and Juan Carlos Lopez. "Civitas: The smart city middleware, from sensors to big data." In: *Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. 2013.

[263] W3C. *WGS84 Geo Positioning Ontology*. 2003. URL: http://www.w3.org/2003/01/geo/ (visited on 06/11/2024).

[264] W3C. *vCard Ontology*. 2014. URL: http://www.w3.org/2006/vcard/ns# (visited on 06/11/2024).

[265] W3C. *IoT-Lite: A Lightweight Ontology for the Internet of Things*. 2015. URL: https://www.w3.org/submissions/iot-lite/ (visited on 06/11/2024).

[266] W3C. *Semantic Sensor Network Ontology (SSN)*. 2017. URL: https://www.w3.org/TR/vocab-ssn/ (visited on 06/11/2024).

[267] W3C. *Sensor, Observation, Sample, and Actuator (SOSA)*. 2017. URL: https://www.w3.org/TR/vocab-ssn/#sosa (visited on 06/11/2024).

[268] W3C. *Time Ontology*. 2017. URL: https://www.w3.org/TR/owl-time/ (visited on 06/11/2024).

[269] Fan Wang, Guangshun Li, Yilei Wang, Wajid Rafique, Mohammad R. Khosravi, Guanfeng Liu, Yuwen Liu, and Lianyong Qi. "Privacy-Aware Traffic Flow Prediction Based on Multi-Party Sensor Data with Zero Trust in Smart City." In: *ACM Transactions on Internet Technology* (2023).

[270] Meng Wang, Jun Wu, Gaolei Li, Jianhua Li, Qiang Li, and Shen Wang. "Toward mobility support for information-centric IoV in smart city using fog computing." In: *IEEE International Conference on Smart Energy Grid Engineering (SEGE)*. 2017.

[271] Tian Wang, Jiyuan Zhou, Xinlei Chen, Guojun Wang, Anfeng Liu, and Yang Liu. "A Three-Layer Privacy Preserving Cloud Storage Scheme Based on Computational Intelligence in Fog Computing." In: *IEEE Transactions on Emerging Topics in Computational Intelligence* (2018).

[272] Doug Washburn and Usman Sindhu. "Helping CIOs Understand "Smart City" Initiatives." In: *Smart City* (2010).

[273] Deutscher Wetterdienst. *Grids of monthly averaged daily air temperature (2m) over Germany*. 2021. URL: https://opendata.dwd.de/climate_environment/CDC/grids_germany/monthly/air_temperature_mean/DESCRIPTION_gridsgermany_monthly_air_temperature_mean_en.pdf (visited on 06/11/2024).

[274] Zhou (Joe) Xu. *Time series pattern recognition with air quality sensor data*. 2021. URL: https://towardsdatascience.com/time-series-pattern-recognition-with-air-quality-sensor-data-4b94710bb290 (visited on 06/11/2024).

[275] B. Yang and H. Garcia-Molina. "Improving search in peer-to-peer networks." In: *Proceedings 22nd International Conference on Distributed Computing Systems*. 2002.

[276] Tan Yigitcanlar and Md Kamruzzaman. "Does smart city policy lead to sustainability of cities?" In: *Land use policy* (2018).

[277] Tan Yigitcanlar and Sang Ho Lee. "Korean ubiquitous-eco-city: A smart-sustainable urban form or a branding hoax?" In: *Technological Forecasting and Social Change* (2014).

[278] Chansu Yu, Ben Lee, and Hee Yong Youn. "Energy efficient routing protocols for mobile ad hoc networks." In: *Wireless Communications and Mobile Computing* (2003).

[279] Narmeen Zakaria and Jawwad A. "Smart City Architecture: Vision and Challenges." In: *International Journal of Advanced Computer Science and Applications* (2015).

[280] Sohail Zangenehpour, Luis F Miranda-Moreno, and Nicolas Saunier. "Automated classification based on video data at intersections with heavy pedestrian and bicycle traffic: Methodology and application." In: *Transportation research part C: emerging technologies* (2015).

[281] Vanessa Zeller, Edgar Towa, Marc Degrez, and Wouter MJ Achten. "Urban waste flows and their potential for a circular economy model at city-region level." In: *Waste management* (2019).

[282] Changhao Zhang. "Design and application of fog computing and Internet of Things service platform for smart city." In: *Future Generation Computer Systems* (2020).

[283] Dongyu Zhang, Wangmeng Zuo, David Zhang, and Hongzhi Zhang. "Time Series Classification Using Support Vector Machine with Gaussian Elastic Metric Kernel." In: *20th International Conference on Pattern Recognition*. 2010.

[284] Junwei Zhang, Fan Yang, Zhuo Ma, Zhuzhu Wang, Ximeng Liu, and Jianfeng Ma. "A Decentralized Location Privacy-Preserving Spatial Crowdsourcing for Internet of Vehicles." In: *IEEE Transactions on Intelligent Transportation Systems* (2021).

[285] Lei Zhang and Jiangtao Li. “Enabling Robust and Privacy-Preserving Resource Allocation in Fog Computing.” In: *IEEE Access* (2018).

[286] Zheng Zhang, Ping Tang, Lianzhi Huo, and Zengguang Zhou. “MODIS NDVI time series clustering under dynamic time warping.” In: *Int. J. Wavelets Multiresolution Inf. Process.* (2014).

[287] Ben Yanbin Zhao, John Kubiatowicz, and Anthony D. Joseph. “Tapestry : An Infrastructure for Fault-tolerant Wide-area Location and Routing.” In: *Science* (2001).

[288] Bin Zhou, Wentao Li, Ka Wing Chan, Yijia Cao, Yonghong Kuang, Xi Liu, and Xiong Wang. “Smart home energy management systems: Concept, configurations, and scheduling strategies.” In: *Renewable and Sustainable Energy Reviews* (2016).

[289] Kevin Zhu. “The Complementarity of Information Technology Infrastructure and E-Commerce Capability: A Resource-Based Assessment of Their Business Value.” In: *Journal of Management Information Systems* (2004).

[290] Yan Zhuang, Feng Lin, Eun-Hye Yoo, and Wenyao Xu. “AirSense: A Portable Context-sensing Device for Personal Air Quality Monitoring.” In: *Proceedings of the Workshop on Pervasive Wireless Healthcare*. 2015.

[291] *ioBroker*. URL: `https://www.iobroker.net/` (visited on 09/24/2024).

[292] *openHAB*. URL: `https://github.com/openhab` (visited on 09/24/2024).

EIDESSTATTLICHE VERSICHERUNG

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Dissertationsschrift selbst verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Sofern im Zuge der Erstellung der vorliegenden Dissertationsschrift generative Künstliche Intelligenz (gKI) basierte elektronische Hilfsmittel verwendet wurden, versichere ich, dass meine eigene Leistung im Vordergrund stand und dass eine vollständige Dokumentation aller verwendeten Hilfsmittel gemäß der Guten wissenschaftlichen Praxis vorliegt. Ich trage die Verantwortung für eventuell durch die gKI generierte fehlerhafte oder verzerrte Inhalte, fehlerhafte Referenzen, Verstöße gegen das Datenschutz- und Urheberrecht oder Plagiate.

Hamburg, September 2024

Philipp Kisters