CCF Transactions on Pervasive Computing and Interaction (2021) 3:76–93
https://doi.org/10.1007/s42486-020-00054-y

REGULAR PAPER

Data-sharing markets for integrating IoT data processing functionalities

Nasr Kasrin[1] · Aboubakr Benabbas[1] · Golnaz Elmamooz[1] · Daniela Nicklas[1] · Simon Steuer[1] · Michael Sünkel[1]

Received: 16 July 2020 / Accepted: 24 December 2020 / Published online: 26 February 2021

Abstract

The recent evolution of the Internet of Things into a cyber-physical reality has spawned various challenges from a data management perspective. In addition, IoT platform designers are faced with another set of questions. How can platforms be extended to smoothly integrate new data management functionalities? Currently, data processing related tasks are typically realized by manually developed code and functions which creates difficulties in maintenance and growth. Hence we need to explore other approaches to integration for IoT platforms. In this paper we cover both these aspects: (1) we explore several emerging data management challenges, and (2) we propose an IoT platform integration model that can combine disparate functionalities under one roof. For the first, we focus on the following challenges: sensor data quality, privacy in data streams, machine learning model management, and resource-aware data management. For the second, we propose an information-integration model for IoT platforms. The model revolves around the concept of a *Data-Sharing Market* where data management functionalities can share and exchange information about their data with other functionalities. In addition, data-sharing markets themselves can be combined into networks of markets where information flows from one market to another, which creates a web of information exchange about data resources. To motivate this work we present a use-case application in smart cities.

Keywords Internet of Things · Infrastructure Design · Information Integration · Data Management Systems · Resource-Awareness · Data Quality · Privacy · Machine Learning Model Management

1 Introduction

Over the last years, the *Internet of Things* has evolved from a high-level vision of always-connected devices to a real cyber-physical system class that appears in many application domains, from healthcare Pike et al. (2019) over smart cities Zanella et al. (2014) to smart farming and precision agriculture Kamilaris et al. (2016). IoT has introduced radical changes in the way data are processed. The amount of IoT data, the velocity of change, and variety of sources/formats implies new challenges to process and inter-operate between heterogeneous data sources and formats Elsaleh et al. (2020).

In addition, we are faced with another set of challenges on platform design level. Assuming these emerging data management challenges are solved, how can we extend IoT platforms to smoothly integrate new data management functionalities? This is an especially pressing problem since various available IoT platforms are not easily extendable beyond the original use-cases assumed by their designers. We have

✉ Nasr Kasrin
nasr.kasrin@uni-bamberg.de
http://www.uni-bamberg.de/mobi

Aboubakr Benabbas
aboubakr.benabbas@uni-bamberg.de
http://www.uni-bamberg.de/mobi

Golnaz Elmamooz
golnaz.elmamooz@uni-bamberg.de
http://www.uni-bamberg.de/mobi

Daniela Nicklas
daniela.nicklas@uni-bamberg.de
http://www.uni-bamberg.de/mobi

Simon Steuer
simon.steuer@uni-bamberg.de
http://www.uni-bamberg.de/mobi

Michael Sünkel
michael.sunkel@uni-bamberg.de
http://www.uni-bamberg.de/mobi

[1] Chair of Mobile Systems, University of Bamberg, An der Weberei 5, 96047 Bamberg, Germany

had firsthand experience with several IoT platforms[1][2] and have faced various issues related to extending platforms with data management functionalities. Currently, data processing related tasks are typically realized by manually developed code and functions which are deployed smart devices (AKA edge nodes), in the cloud, or on in-between on so-called fog nodes. This creates future difficulties in maintenance and growth of the platforms. Hence it is needed to explore other approaches to designing IoT platforms that can be extendable to integrate emerging functionalities. Overall these challenge relate to devising approaches, techniques, and system platforms that ease the development and maintenance for the developer and operator of systems.

In this paper we cover both these aspects: (1) we explore several emerging data management challenges, and (2) we propose an IoT platform integration model that can combine disparate functionalities under one roof.

We call the IoT integration model we develop the *Information Basin (IB)* model. The IB model is information-driven in the sense that data is shared by exchanging information that describes the data. We model each data management functionality/component as an agent that describes its data and shares its descriptions with other agents. The IB model revolves around the concept of a *data-sharing market* which is a shared information space where multiple agents contribute by curating and 'advertising' their data resources and discovering those of others. So for the IoT platform use case, various IoT functionalities integrate by exchanging the data descriptions of their inputs and outputs within data-sharing markets. To allow more sophisticated organizations of data exchange and integration, the IB model has a distributed network structure, where vertices are data-sharing markets and directed edges are information flow between them where data descriptions travel from one market to another. This enables more complex hierarchies of data-sharing markets to form. We will discuss the IB model in more details in the approach section in this paper.

Regarding the data management challenges we explore, let us first look at the landscape of IoT data management challenges. Abbasi et al. (2017) present a set of data management challenges In the IoT, their list includes: standardization, data storage management, confidentiality and privacy, integrity, energy constraints, device mobility and heterogeneity, device security and backup, availability, and internal adversaries. From the big data side we have the four Vs of volume, velocity, variety, and veracity. The challenge we discuss in this paper intersect with several of those topics. The topics we do *not* cover include: security, data volume, integrity, and standardization. Next we introduce the set of data management challenge covered in this paper.

1. **CQuality**. Many IoT applications are based on sensor data which is never perfect; we have to deal with data quality issues which might even change depending on the context of the sensor. Hence, the need for methodologies that model sensor data quality for processing and decision making open the door to the challenge we call the CQuality challenge.
2. **CPrivacy**. IoT devices are ubiquitous, sensors are present in more devices that produce continuous streams of data about their environment. This leads to the situation where people are not aware of the information they share with others, which paves the way for the challenge of privacy preserving data stream processing, which we refer to as the CPrivacy challenge.
3. **CModel**. Recently there has been work in utilizing machine learning to train models that then replace manually programmed or modeled functions, e.g., for activity recognition. We consider the life-cycle management of these models as part of a (higher-level) data management. We refer to this challenge as the CModel challenge.
4. **CResource**. Data management is distributed geographically between the sensor/actuators, gateways up to the cloud. This distribution leads to a high heterogeneity between the processing nodes in terms of computing resources, system security and connectivity; this creates the opportunity for building solutions that tackle the management problem from a resource-aware approach, which we call the CResource challenge.

The rest of the paper is structured as follows: in Sect. 2 we present a motivating use-case for the paper, and in Sect. 3 we discuss our approach to IoT platform design. Next we present the challenges: in Sect. 4 the *CQuality* challenge, in Sect. 5 we highlight issues related to the *CPrivacy* challenge, and in Sect. 6 we discuss the *CModel* challenge. In Sect. 7 we cover the *CResource* challenge, in Sect. 8 we discuss related work, and wrap up in Sect. 9 with conclusions and future work.

2 Use case

The establishment of smart cities can be supported by the Internet of Things (IoT) platforms with sensors collecting data and improve the life quality and resource efficiency in future cities. Many smart city applications use their gathered data to measure city-wide processes like mobility, energy, or environmental factors like air quality. Various challenges arise in managing this huge amount of data consistently.

[1] ThingsBoard, https://thingsboard.io/.

[2] Cumulocity, https://www.softwareag.cloud/site/product/cumulocity-iot.html.

To define the challenges, explain and provide appropriate solutions first, we describe our use-case, a smart city platform, the so-called Living Lab Bamberg. This testbed provides a user-centric environment to test and use different sensing systems. The Living Lab is open for industry partners and citizens who help us receive new sensor systems and find installations locations. Furthermore, the University of Bamberg is not a campus university with the advantage that we can place lots of sensors in different buildings inside the city.

In general, we use different kinds of stationary and mobile sensors systems. An example of a stationary sensor is a people-counting camera, and an example for a mobile sensor is a sensor box in a city bus to measure air quality. These sensing systems produce data that we use in our different application scenarios. We describe some of these application scenarios below.

One application scenario is indoor and outdoor localization management. Every year many people visit a lot of street festivals in the world's cultural heritage, Bamberg. Bamberg is a medieval city with many narrow alleys and small lanes that make it difficult to plan events with a lot of visitors. The goal was to flow-track the movement of visitors in the area of the festival, as well as learn movement models of the civilians and groups of people. This can help on a short term basis to predict escape routes on the fly, or in retrospect to plan for future planning of street festivals.

For the measurement, we used a combination of different sensing technologies like manual counting, camera-based counting, and Wi-Fi tracking of mobile phones. In addition, the people tracking camera system helps us to collect trajectories from people inside buildings (an example of indoor localization would be an iBeacon network). In this environment, we can simulate tourist information panels inside the university or guides for city museums.

Already in such a scenario we have several data management challenges arising. We also highlight the information-drivne aspect of each challenge since this is the key property we use in the proposed information integration model, the Information Basin (IB) model.

1. **Festival Sensor Data Quality**. The above sensors, like most other sensors, produce readings that are never perfect. However for it should be possible to build quality models for data–using sensor models and environmental context–to enrich them with quality information. This can greatly help in correcting some kinds of faultiness or at least in describing the nature of the faultiness of the sensor data. Such quality information can be invaluable for developing machine learning models for example (see below). The *CQuality* challenge we present in this paper forms an instance of studying this problem. It must be noted that in the above approach sensor quality models, as well as quality information attached to data sets (ex. metadata) are both structured information, which can be represented, exchanged and processed by others in the IB network.
2. **Visitors' Privacy**. As data is collected about festival attendees, vulnerability of the visitors' privacy is the Wi-Fi tracking of mobile phones arises. How can we collect data from the festival attendees while protecting their identities? The *CPrivacy* covers this aspect of the problem. Although not as information-driven as the sensor data quality challenge, information representing which parts of the data are privacy sensitive, or what algorithms and parameters to run on them, are structured information that drives privacy data processing.
3. **Festival Attendees Movement Models.** Developing movement models helps to predict, mange emergencies and support the planning of future festivals. Sensor data and its quality information can be used to develop these models which in part are also structured information that can be processed by different systems. This forms the *CModel* investigation in this paper. Examples of strucutre information driving this data management functionality include, training datasets used, learning parameters, and the learned models themselves are all structured information that are key to driving machine learning model management.
4. **Resource-aware Computation of Festival Data.** So far we have assumed that computation is managed locally or by some high-performance machine. But the IoT reality has given us many options to run computations. For privacy, it can greatly benefit if data is for example pseudo anatomized on the fog node, or the nearest gateway, if it has such computational capabilities. Also for developing machine learning models, resource-aware computation can provide various alternatives to pre-process, clean-up, or run jobs across nodes. This challenges, as opposed to the three above, is a service challenge that supports other data management functionalities.
5. **Integration**. Various data management functionalities arise in this use-cases, and IoT platforms suffer from a lack of fluidity when new features or functions are added to the platform. So with discussing all the above challenges, a natural question arises, how can we integrate such functionalities together? Towards this end we present an information-driven paradigm for IoT platform design. We model the data management functionalities referenced above as agents, which processes data and consumes and produces information about this data. Then integrating the different functionalities into a common framework by sharing and exchanging all this information. We present this model in the next section.

Mobility sensing is just one use case for smart cities. With such a small use-case, we already see various motivating requirement for the challenges and approach presented in this paper. Similar challenges can be found in other applications as well in such as environmental sensing, using distributed air quality measurements and analysis.

3 Platform integration: the information basin model

We have introduced several IoT platform functionalities so far. From a platform design perspective we model each functionality as a *functional dimension* of the IoT platform, as defined below.

Definition 1 (Functional Dimension) A functional dimension is a component of an IoT platform that: (1) adds independent functionality or value of its own (i.e. processing, input to output dynamics), and (2) is expected to integrate with the platform (ex. by producing or consuming data/information).

Sensor data quality management (CQuality) is an example of functional dimension of an IoT platform since it adds an independent functionality (processing sensor data and sensor information and producing data quality models, etc.) and it must integrate into a larger platform so that its results can be used by other systems. Machine learning model management (CModel) shares a similar structure to CQuality as a functional dimension, and so on.

Distributed resource-aware computing (CResource) can also be understood as a functional dimension in the sense that it adds functionality and produces results/data back to the platform. A similar argument can be made about CPrivacy in the sense of adding functionality and integrating into the larger system.

Faced with expanding requirements, innovations, and functionalities of IoT platforms, we must find ways to design platforms so that they can handle multiple functional dimensions as well as be extendable to integrate new ones. To achieve this we use a model we are developing[3] called the *Information Basin (IB) model*, which revolves around the concept of a *data-sharing market*.

Definition 2 (Data-Sharing Market) A data-sharing market is a shared information space where multiple organizations can describe and 'advertise' their data resources as well as discover those of others.

[3] Submitted, awaiting review.

Online open-data repositories are one example of a data-sharing market, however their underlying model for information representation (i.e. list + keywords) might be primitive for more sophisticated applications. The information representation model we currently use is more akin to a lightweight semantic graph model driven by shared domain models.

Definition 3 (The Information Basin (IB) Model) The IB model is a platform design model for enabling data-sharing and exchange networks across multiple groups, users, or systems. The basic building block of the IB model is (1) a set of data-sharing markets and (2) information flow between them. The IB model is 'information-driven' in the sense that data is shared by exchanging *information about* data. An IB network is 'animated' when (1) information about new data resources is added to a market, or (2) information flows from one market to another, thereby creating a kind of living web-of-markets. Since we are dealing with (structured) information here, if we assume that markets can have different domain models, information flow must include a mapping stage from the source domain model to the destination model.

Continuing with the online open-data repository example, implementing an IB network using them will require establishing a mechanism whereas some selection of entries from one repository can be propagated to other repositories, where each repository might have a different set of users, visibility etc.

So how can we apply the IB model to establish an expandable IoT platform with multiple functional dimensions? We can design a simple IB network where each functional dimension has its own internal data-sharing market, and in addition declare an IoT platform-wide market where information flows from the respective feature dimension markets to the main one.

A primitive IB network would be a single IoT platform-wide market shared by all functional dimensions, and although this can work, this assumes that all the dimensions must adhere to the same domain model of information, as well as being forced to share all produced data to this market. Usually functional dimensions would have some local/private data that is not meant to be shared, in addition to having domain models that are specialized differently. So for these reasons we will opt for a multi-market IB network. Figure 1 depicts this IB network, where *IoT-Mark* is the name of the IoT-wide shared market. Regarding the contributors in the different markets, we can assume that each functional dimension market is accessible only by users/systems of that functional dimension and the IoT-Mark market is accessible and writable by all the users/systems of all the functional dimension.

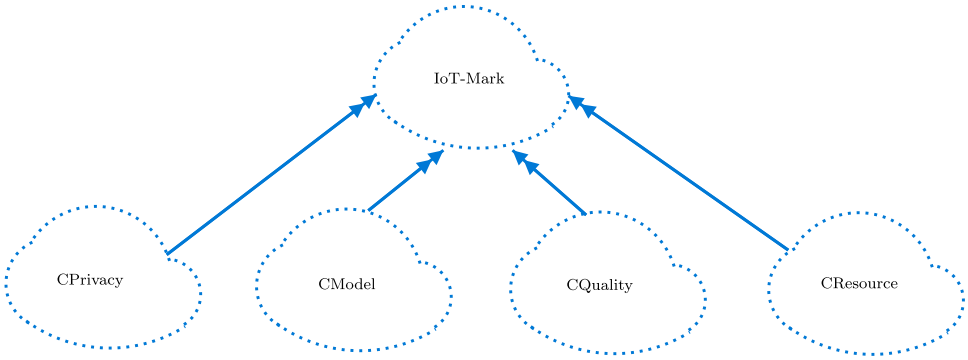


Fig. 1 The IoT platform IB network with clouds denoting markets and arrows denoting information flow

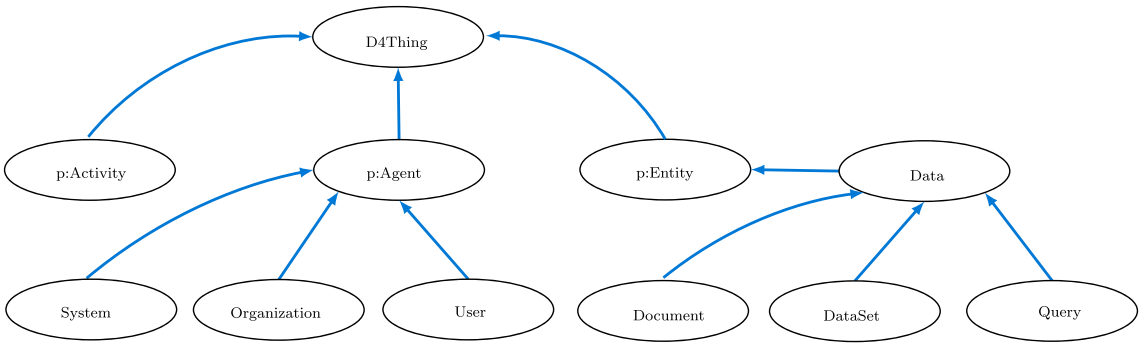


Fig. 2 Abstract shared model, with arrows denoting the *type-of* relation

Next we will looking at the example of how the above IB network can support an IoT platform with the multiple functional dimensions covered in this paper. First regarding domain models across markets, we can give the IoT-Mark market an abstract domain model that is specialized by each market differently. By specialized we mean the abstract model is extended by adding new types that must be a subtype of some element in the abstract model. Figure 2 depicts one such model, where the '*p:*' prefix denotes that the type comes from the PROV W3C provenance recommendation[4].

The CQuality market can further specialize the above model by defining a rich sub-hierarchy of sensors and sensor quality models, the CModel market on the other hand can define a rich sub-hierarchy of type DataSet above, and so on. Let us look at a concrete example of how the functional dimensions can integrated within the IoT platform.

Example 1 The CQuality solution processes a DataSet and its context and produces a reliability model. Let's assume this reliability model is defined in the abstract domain model on the DataSet type, hence it is understood by the other groups. The CQuality market would have a forwarding rule to the IoT-Mark market for all new reliability information, and hence this piece of information about this particular DataSet instance will be propagated. In the IoT-Mark market the CModel group can query for all DataSets with reliability information and use that information as parameters to their learning algorithms and produce a learning model where they declare that the DataSet above has been used as training data. Let's say that hypothetically, a problem occurs with the learning model and someone would like to diagnose the possible reasons, with the information about which DataSets where used in the training they can trace this same DataSet back to the CQuality group and can explore more information about how the reliability information was derived, and so on.

In essence, a functional dimension = (structured) information + operations (that use and produce more information). And an IB network = information models / mappings + markets + information flow.

We will not go into more details regarding the approach due to scope. The goal of this section was to show an IoT platform model where a set of IoT functionalities can be combined under a common roof collaborate and exchange information and data to achieve the goals of an IoT platform. The remainder of this paper will dive into the various challenges introduced earlier.

[4] https://www.w3.org/TR/prov-overview/.

4 CQuality: data quality challenges in IoT environments

This challenge revolves around the quality of data in the context of IoT. Most applications deal with data coming from different sources. These sources can be human or non-human. Non-human data can come from other similar devices or from sensors that capture the phenomena happening around and quantifying them.

To show the importance of data quality, we can relate to the use cases in Sect. 2, where machine learning models are created on the basis of sensors. If these datasets are not checked for quality, the models created could give a skewed view on the actual observation/behavior. Any predictions based on the learned models from raw data will result in wrong predictions. In this section, we focus on the main challenges of data quality in IoT environments with regards to the stream processing applications. Since most of the data in IoT context is generated from sensors, we know how faulty can the data be. Besides, we note that decision-making is made on the fly. Those two aspects make the data quality an important issue in the process of data evaluation and usage. First, we start by enumerating the different challenges that face the developers of IoT applications and then we describe them individually in this section.

When we talk about data quality, we have to precisely define what quality means. This definitions can be defined by introducing the dimensions that make up the data quality. This challenge is concerned with the definition of data quality. After having a clear idea of the quality dimensions and their metrics, we have to determine the values of those metrics. This poses the second challenge of data quality computation. We also want to make the use of quality-aware data processing as easy as possible by having an easy integration for application developers.

To summarize, this challenge focuses on the following research question: How can we integrate quality-aware data processing in IoT applications through semi-automatic and automatic methods?

4.1 Related work

In this part, we give a background on the relevant research that addresses the different aspects of data quality in IoT environments.

4.1.1 Data quality definition

The first challenge is to clearly define what data quality is. According to Buchholz and Schiffers (2003), quality is "Any information that describes the quality of information that is used as context information". Batini et al. define the data quality in terms of specific dimensions Batini et al. (2006) like accuracy, completeness, volatility and timeliness. Klein and Lehner (2009) define data quality as the accuracy and resolution of the data as it goes though the steps of a data processing chain. With these definitions of data quality, we have to distinguish between inherent quality dimensions and domain-specific dimensions.

The inherent quality dimensions can be automatically computed like those given by Batini. However, the domain-specific dimensions must be defined by the application developers. This makes the task of data quality definition using a specific tool a little bit tricky.

In order to deal with data quality representation, we can use semantic tools like ontologies to define both types of quality dimensions. The inherent quality dimensions can have their own terms in an ontology like the SSN ontology Compton et al. (2012). The domain-specific dimensions can be expressed through extensions of ontologies or by enabling the inclusion of user-defined terms to include these. However, the definition of any data quality dimension or process must be simple and clear to encourage any users to adopt it.

4.1.2 Data quality processing

The challenges in the area of data processing are numerous. Should the quality dimensions be computed online or offline? Do we output data with quality annotations or metadata about the quality?

The online approaches are mostly present in applications that process the data on the fly as an incoming stream. Geisler et al. (2016) introduces a Data Quality ontology-based framework for data stream applications. The ontology gives quality metrics for content, queries and applications. The framework is based on an ontology for the description of sensor and quality dimensions. Kuka and Nicklas (2014) give an approach for quality-aware sensor data processing based on the SSN ontology Compton et al. (2012). The ontology is used to describe context information about the sensors deployed. The Gaussian Mixture Models are used to assess the probability of a data element being an outlier.

Schmidt et al. (2004) adopt a deterministic data stream processing approach with a system called QStream. Klein and Lehner (2009) propose a flexible model of data quality processing and propagation in a stream processing network for sensor data in a smart environment. We also applied online data stream processing for quality estimation of sensor data in Benabbas et al. (2018, 2019, 2020); Aboubakr et al. (2017a).

The mentioned approaches process the data on the fly and can be also applied offline. These approaches include processes for the computation of the said quality dimensions. The user-defined quality dimensions come with user-defined

procedures to determine the values of the quality dimensions. Besides, we need to consider the trade-off between the accuracy of the computed quality and the costs in terms of computation overhead and delay. Some of the approaches need a training-set to develop a model for their quality-aware processing Wu et al. (2007), while some do not need any training to start their process. These two variants give the user a choice between a delay-free quality-aware processing and one with an up-start time. The challenge is to find a hybrid approach that enables the choice between the different models and the activation/deactivation of quality computation processes.

4.1.3 Data quality integration

This challenge builds on the first two challenges. Given that all the challenges above are solved, how do we make the integration of quality aware processing as easy and as seamless as possible? From the above discussion, we note that most systems deal with data quality as a parallel process or a pre-processing step before passing the data to the actual application. This challenge implies that for any quality-aware solution to be adopted, it must be simple enough and intuitive enough to be wide-spread over all the IoT applications. This gives rise to the following two challenges:

- Simplify the use of semantic models to describe the quality-aware processes.
- Automatically generate the processes from the semantic descriptions.

4.2 Our approach

The first challenge must be addressed through the introduction of processing patterns, which can be deduced from the basic structures of the participating data sources in the IoT applications. The second challenge can be solved by having processes in the background to perform the translation from the semantic model to the actual quality-aware processes. A first step towards these goals is done in a previous contribution Benabbas et al. (2020), where we define the first processing patterns and their use with an automatic generation of the quality-aware processes. The target is to be able to provide templates of sensor models to be used by the IoT application developers to deal with the quality issues wherever applicable. Besides, we want to leverage large scale models that contain multiple sensors to find the target groups for data quality checks.

Figure 3 shows the cycle of data quality integration into IoT platforms. Developers design their applications by having models of the data sources and sensors they have. To write the models, we can use the aforementioned *SSN ontology*. The models are fed to the *Data Quality Management Tool* that extracts the data quality processing patterns from the model. The patterns can be identified through the semantic relationships between the sensors and other spatially collocated sensors or with other data sources. The automatic recognition of such quality patterns and the generation of the queries should solve the second challenge. The processing patterns indicate the method of checking the data quality. Then, the process of quality-aware processing queries generation.

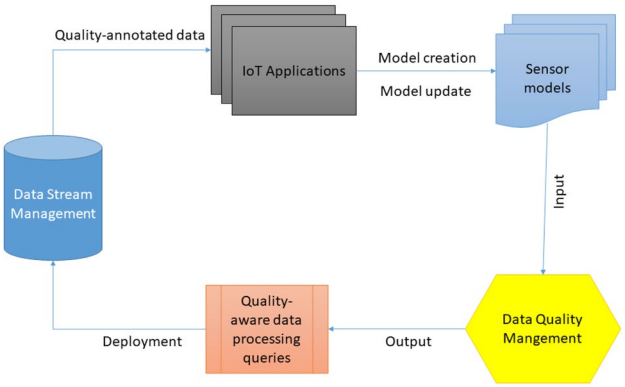


Fig. 3 Cycle of data quality integration in IoT applications

The output of the process are quality-aware queries, which can be deployed on data stream management systems (DSMS) to perform the data quality metrics computation. The results of those queries are quality-annotated data, which are ready to be used by the IoT applications. Any changes in the application model can be updated in the model and this triggers a chain of updates on the queries to reflect the change in the model.

The third challenge of *Data Quality Integration* is the long term goal for IoT applications, where the above steps of model creation and query generation should be standardized and made as a part of any IoT development platform. The developers should have all the necessary tools to make quality-aware processing a permanent part of the development of applications.

5 CPrivacy: privacy-preserving data stream processing

IoT devices are ubiquitous. Sensors are present in more and more devices not just in smart phones or wearables. This leads to the problem that the people are not aware what information about their daily lives they share with others. An example like a fitness tracking app that gives away location of secret US army bases Hern (2020) seems funny, but there are too many privacy fails that this problem could be ignored.

Recently laws are being published to ensure the security and privacy features of IoT devices. For example, the federal state of California has prohibited the use of standard passwords (2020). This is a good first step to define standards for private IoT devices.

The problem is that nowadays you can be monitored by sensors without knowing that. Especially if companies like supermarkets use systems to track customer to analyze their shops. In 2018 the European GDPR was launched and provided a good foundation for privacy. Since that time supermarkets were not allowed anymore to store private information like the mac addresses of the customer's mobile devices in plain text.

We want to analyze stream of stakeholders of a street festival to create new security and safety concepts. How can we set up long-running city-wide sensor campaigns and share the data without compromising the citizen's privacy? We integrate different state-of-the art privacy methods to reduce the risk of leaks in data publication to a minimum. In a next step we adopt our prototype to data streams. We will refer this challenge as the **privacy-preserving data stream processing** challenge.

To summarize, this challenge focuses on the following research question: How can we process privacy-preserving data streams without publishing sensitive information's?

5.1 Related work

We already defined our privacy-preserving architecture for our smart city testbed in Steuer et al. (2016). One of the main challenges is the publication anonymous data. Privacy-Preserving Data Publishing (PPDP) is a good fundamental for our research in this domain. Fung et al. treated also moving object data in their survey Fung et al. (2010) as privacy in location-based services (LBS). "Location privacy is defined as the ability to prevent untrusted third parties to reveal current or past location[s] of an individual" Pelekis and Theodoridis (2014). There are two general approaches to prevent re-identification in trajectories: spatial cloaking and perturbation Pelekis and Theodoridis (2014).

In the cloaking approach the generalization increases the query region until the region contains at least k users. The probability of de-identification is not higher then 1/k. Examples for cloaking are based on k-anonymity like Always-Walk-with-Others. The cloaking approach is just possible if we have a high number of trajectories, if we do not have this then we need perturbation like in the Never-Walk-Alone approach or decide not to publish data sets Pelekis and Theodoridis (2014).

In data publication we focus on k-anonymity because we have just slightly sensitive data that can be anonymized easily without much risk. Our goal is to use k-anonymity also for data streams. There have been several scientific articles over the years, which try to extend the concept of k-anonymity to streaming data. The Continuously Anonymizing Data Streams algorithm (Castle), presented in Cao et al. (2011), has the most similarities to our approach.

Existing privacy-preserving techniques such as k-anonymity are designed for static data sets. The adaption to data streams is challenging because of the differences to classical data bases. In data streams the data input is continues and does not stop. Furthermore the data arrives in real time in an ordered sequence of items Golab and Özsu (2003). In our use case in Sect. 2 entities can appear more than one time if a visitor enters a new region and then immediately returns.

5.2 Our approach

In our approach, we decided to use generalization with k-anonymity. In the first step, we analyze our stored data set from the crowd monitoring use case in Sect. 2. The identifiers in our data set are the mac address and a combination of time stamps and location points, which makes it possible to encrypt stakeholders.

Mac address is the sensitive attribute that we protect via pseudonymisation with hashing. Regarding the quasi identifier time stamp and location, we conclude that we need a location point (e.g. City Hall) and a time stamp categorization (morning, afternoon, evening and overnight) as clusters. The trajectories just consist of these cluster set of elements. A typical trajectory looks like:

City Hall[morning] - Lower Bridge[morning] - Upper Bridge[morning]

Our goal is to see, how k-anonymity can be applied to data streams with the data set from our crowd monitoring use case presented in Sect. 2. Towards enable privacy-preserving data stream processing, we want to extend our data stream management system (DSMS) Odysseus Appelrath et al. (2012). Therefore, we want to implement a standard operator, so that adding anonymization to data streams becomes easy for developers and can be enforced easily.

Hence, very similar to the static data set, the identifying information of stakeholders the mac address is removed. After that, we focus on the following questions: (1) How many tuples are stored temporarily before they get published as clusters? (2) What should be a good window size, so that we can guarantee the privacy in the diversity of trajectories and that we do not lose too much data sets? After answering the questions (1) and (2), we can use a predicate window with a predefined size so that tuples have to fulfill the predicate $cluster\_size \geq k$ to get published.

The anonymization operator is optimized for one data set in one specific use case. In order to make it useful for developers, more crowd sensing use cases have to be supported. For this end, we want to define a set of the most probably scenarios and define the concrete parameters for them.

6 CModel: ML model management

IoT platforms are made of a huge number of different sensory devices that produce a huge volume of data. There are different applications of IoT in different environments, like the smart city described in Sect. 2. Many different services can be defined in such environments as smart mobility monitoring, smart traffic management, and smart buildings. These different services should provide appropriate knowledge and insight into the environment and support domain experts in making critical decisions.

For example, in the smart city scenario, all the people can use a traffic management service, which suggests the shortest path to the destination based on different possible paths' traffic load.

The word machine learning refers to a computer program that is said to optimize a learning criterion using example data and post experiences Alpaydin (2020). Machine learning (ML) and data analytics techniques are powerful tools that provide us the ability to extract knowledge from transmitted data. Based on the requirements, different ML and data analytics techniques should be applied to provide a service in an IoT platform Samie et al. (2019); Cui et al. (2018). To design the best ML model framework that manages all the ML models in our IoT platforms, first, we have to define the challenges related to ML model management. This section reviews the related works for ML model development and management in an IoT platform, and then we define and discuss the challenges related to ML model management. In the end, we describe our approach and explain how we can overcome the challenges and answer the research question.

The research question for ML model management would be: *how we can integrate and manage a variety of ML models with different properties in an IoT platform in a unified and scalable framework?*

6.1 Related work

A variety of ML and data analytics techniques have been introduced to deal with a massive amount of heterogeneous data gathered by the IoT infrastructure Samie et al. (2019); Cui et al. (2018). Different ML models can extract different patterns and knowledge needed by applications. Based on the application demands, the appropriate ML model with the proper properties can be defined.

Data should go through stages of ML pipeline and sometimes combined with context knowledge to gain the appropriate knowledge. ML pipeline consists of different ML techniques for data cleaning, preprocessing, data segmentation, feature selection, processing, and postprocessing. So many related works focused on developing different ML models and combining the ML models from different stages of pipeline to extract the accurate results Chin et al. (2017); Patil and Thorat (2016); Mahdavinejad et al. (2018). Some mature works like Vlacheas et al. (2013) introduced an ML model management framework with some level of automation for selecting the ML models. Besides, combining the context knowledge like what has been done in Sasidharan and Somov (2014) improved the accuracy of ML models. However, all the mentioned works are developed for some limited applications.

In addition, each ML model has its evolution life-cycle. This means that every time an ML model is developed, it must be evaluated and deployed Schelter et al. (2018). The life-cycle of an ML model produces different versions of an ML with other properties.

The rapid speed of growing IoT platforms and maturation of ML techniques address the need to have a unified framework for managing the ML models and corresponding metadata and connections for each model to overcome the aforementioned challenges. The ML model management should allow us to integrate new ML models to the framework, elevate existing models, track and access different ML models with corresponding metadata, and decide which model should be used to extract the desirable knowledge. Such an ML management framework should be able to overcome the challenges of managing ML models that can be divided into three main categories. In the following, we explain each challenge, describe the related works, and discuss the shortcomings and new insights.

6.1.1 ML models for different applications

IoT platforms can cover a vast area like a city Steuer et al. (2016), a building Elmamooz et al. (2017), or a farm Kamilaris et al. (2016). Different users and agents with different demands can be defined in an IoT covered environment. For example, taxi drivers, tourists, and citizens of a city can benefit from smart mobility monitoring service with different applications Zanella et al. (2014). A taxi driver needs the fastest path to the destination, a tourist needs a recommendation for the next interesting place in the city, and a user can use a traffic load app to decide on the hours for shopping. So many different ML techniques have been introduced to extract relevant knowledge for an application. The ML techniques can be categorized into three main categories of supervised learning, unsupervised learning, and reinforcement learning Kavakiotis et al. (2017). Different techniques are introduced to extract hidden knowledge from data. This knowledge can be in the form of classified data, frequent patterns, sequential patterns, and so forth. This knowledge should be presented in an understandable way for the end

user. Moreover, based on the application demands, different ML models should be executed offline or online over different data segments Zorbas et al. (2015). In some cases, it is needed to execute an ML model over various window sizes to extract the hidden patterns and anomalies in different time and space granularities.

Most of the recent works focus on developing the algorithms that fit best to the specific demand like Mahdavinejad et al. (2018); Kavakiotis et al. (2017); Sasidharan and Somov (2014). However, managing ML models should not just focus on developing the most efficient models but also on model management efficiently. This means that an ML model management framework should be developed in a scalable way to make the integration of new models easier. In addition, different components of a framework can be (re) used for other domains and applications.

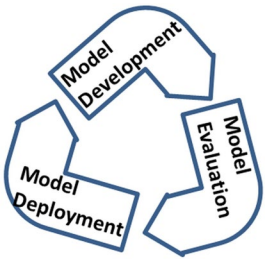
6.1.2 Pipeline of ML models

The data transmitted from sensory devices in big IoT platforms are completely heterogeneous and full of noise and missing values, as mentioned in Sect. 4. Moreover, each bunch of raw data gathered from a specific kind of sensor has its own characteristics like the type of data and number of fields. Gathered data from heterogeneous sensors in different places and time should be processed to return the desired result. The process of converting raw data into knowledge consists of different steps called ML pipeline Schelter et al. (2018). These stages are data cleaning, data preprocessing, data segmentation, feature selection, processing, and postprocessing.

Each step in the pipeline includes one or more ML model that receives an input dataset and converts it to the output dataset. In the cleaning and preprocessing step, some ML models should be used to leverage the quality of data for further usage like noise reduction, data segmentation, and resampling. In data segmentation and feature extraction, the data is divided into batches based on time and space, and different features of a batch can be selected based on their effects on the evaluation of the model. In the processing step, based on the application, different supervised, unsupervised, or reinforcement ML models can be chosen to extract the knowledge out of data. As the last step, in some cases, the result of the second step of the pipeline should be further processed to produce understandable knowledge.

It should be mentioned that combinations of different ML models can change the evaluation results. For example, some preprocessing techniques like discretization can enhance the accuracy of some ML models that cannot work with continuous values Zhu and Collette (2015). Therefore, the ML model management framework should be able to keep and retrieve different ML models in different stages of the pipeline based on the connections between the models. Such a framework should help choose a suitable (combination of) ML model(s) to get the desirable results.

Fig. 4 Life-cycle of ML models



6.1.3 The life-cycle of ML model evolution

Developing an ML model is a continuous task. This means that every time an ML model is developed, the model should be validated with the data and be improved based on the validation results. Figure 4 illustrates the life-cycle of ML models.

The life-cycle of an ML model consists of three stages, including model development, model evaluation, and model deployment Schelter et al. (2018). After defining the training data, selecting the features, and training the model with the training dataset, the model should be evaluated with test datasets. In most cases, an ML model is a combination of feature transformation and a learning algorithm with tuned parameters. An ML model is implemented and integrated to accept domain specific input data and return reliable results. Therefore, the selection of features and tuning the parameters can be made based on the distribution of data. The performance and accuracy of an ML model can change over time and space with changes in the data distribution. To keep the results of ML models reliable, ML models should be validated and improved over time.

The continuous life-cycle of ML models raises the demand to keep the trace and information about different ML over time and space. Introduced ML model management frameworks mostly keep the current version of ML models Mahdavinejad et al. (2018). Keeping the current version of ML models without the history of evolution is not enough for most of the recent IoT platforms. Keeping the trace of the ML model evolution helps us to use and compare different versions on different datasets and produce new versions without losing the previous versions.

6.2 Our approach

Based on the discussed challenges, various ML models with different properties can be developed for an IoT platform. These properties can include information about the input, output, parameters, evaluation results, and connections to other models. Besides, ML models might be

developed in different development environments. To be able to integrate all ML models that are implemented in different environments, we need to define an efficient ML model management framework that can fulfill the following goals:

- The framework should be scalable
- The framework should manage several executions of ML models on different input
- The framework should keep the metadata for different ML models

One of the main capabilities of the ML model management framework is scalability. This means the development and integration of new ML models in the framework should be possible. Such a framework should be easily extended to consider new data sources and new ML models that can deal with the data from new sources. As an example, we can mention continuous ML models in the form of queries for streams of data and incremental ML models for batches of input data.

The second goal is to have a framework that can manage the execution of one or more ML models on the data with different spatio-temporal granularity. The combination of the results of several executions brings new knowledge and insight into the data. To make it understandable, a simple ML model like detection of the top points of interest from the trajectory of tourists in smart city use-case can be considered. The top ten points of interest (POIs) during the week-days might differ from the top ten POIs at the weekend. Weather conditions, festivals, and school time can also change the top ten POIs in a city. By combining the results from several executions of the top POIs detection model, we can discover the mobility patterns of tourists in the city.

In addition, to integrate different ML models and track different versions of an ML model, the framework should be able to keep the metadata of each ML model and connections between ML models. Metadata in the form of connections can be defined between different ML models. For example, an ML model can have different evolutionary versions that should be traceable. Moreover, a combination of some special ML models can have a noticeable effect on accuracy. As the framework supports the scalability and reusability of ML components, different adapted versions of an ML model on heterogeneous data can be developed. Each version of an ML model has its own properties, as mentioned before.

One of the main future goals in the context of ML model management framework is to reuse the framework and components in different IoT platforms with minor changes for adapting ML models to manage and automate all the ML related tasks.

7 CResource: resource-aware data management

The data that needs to be managed in an IoT ecosystem steadily grows in all of its three big data dimensions: volume, velocity and variety. The volume increases due to the elevating amount of data generating devices Atzori et al. (2010); Gubbi et al. (2013) and velocity by advances in communication technologies like 5g Rath and Kumar (2018). Kaur et al. even calls it Internet-of-Big-data (IoBd) Kaur et al. (2020).

The processing of this huge amount of data utilizes many resources. Current IoT platforms are mainly centralized and lack the feature of resource-aware processing in the sense of edge and fog processing Mineraud et al. (2016). Centralized processing is generally sub-optimal since it uses the WAN bandwidth highly inefficiently due to sending all data to the cloud in order to process it there. Furthermore, cloud computing induces high latency, high energy consumption and arises privacy concerns. There exists a rule of thumb that you prefer computation over communication when considering resource-awareness. Properly positioning the processing along the way from the data sources to the sinks is the intended strategy. Enabling edge and fog processing is crucial for being resource efficient and for real-time low latency applications. The data processing in IoT is geographically distributed by the nature of the ecosystem Chandra et al. (2018); Heintz et al. (2015).

To summarize, this challenge focuses on the following research question: How can a global query be distributed in a geographically-distributed data stream management system considering the limitations of the IoT ecosystem?

7.1 Related work

In data stream query optimization a query is optimized to improve runtime performance in the sense of throughput, latency and resource usage. The throughput is wanted to be as high as possible and states how many data points can be processed in a specific time unit. The latency is wanted to be as less as possible and states the time it takes from a data point entering the processing pipeline until the very same data point being reflected in the results. The resource usage is wanted to be as low as possible and states the usage of CPU, RAM, network bandwidth and even battery-/energy-consumption.

Data stream query optimization approaches can be categorized in 11 classes. An optimization technique can either change the data flow/operator-graph or leave it unchanged, can either change the semantics of a query, which means

Table 1 Query optimization categories

#	Optimization	Graph	Semantics	Dynamic
1	Operator reordering	Changed	Unchanged	(Depends)
2	Redundancy elimination	Changed	Unchanged	(Depends)
3	Operator separation	Changed	Unchanged	Static
4	Fusion	Changed	Unchanged	(Depends)
5	Fission	Changed	(Depends)	(Depends)
6	Placement	Unchanged	Unchanged	(Depends)
7	Load balancing	Unchanged	Unchanged	(Depends)
8	State sharing	Unchanged	Unchanged	Static
9	Batching	Unchanged	Unchanged	(Depends)
10	Algorithm selection	Unchanged	(Depends)	(Depends)
11	Load shedding	Unchanged	Changed	Dynamic

that the output will differ from the original one, or leave it unchanged and can be applied statically before runtime or dynamically during runtime. An overview of all the query optimization categories is shown in Table 1.

The data stream query optimization category, which the resource-awareness of data stream processing in IoT fits best, is operator placement. This kind of optimization usually assigns data stream operators to hosts and cores reducing either resource usage due to less communication or better utilizes available resources Schneider et al. (2013); Hirzel et al. (2014, 2018).

According to Sharma et al. (2019) who did a survey on cost-based distributed query optimizers there are two types of query optimization procedures which are either cost-based or heuristic.

Daum et al. proposed a framework called Data Stream Application Manager (DSAM) Lauterwald et al. (2012) to control a network of heterogeneous data stream management systems. A cost model Daum et al. (2011) is used to minimize the overall processing and communication costs where the operator placement query optimization is modelled as a task assignment problem.

Xu et al. optimize data stream queries for distributed/edge processing to minimize latency providing a framework called QueryGuard Xu et al. (2018). They are also using a cost model and a dynamic programming enumeration algorithm in combination with heuristic rules to prune unsatisfied branches in the search space. This approach also guarantees preserved privacy for edge computing.

Pietzuch et al. (2006) propose a virtual stream-based overlay network using a multidimensional metric space to find a good latency bandwidth trade-off for operator placement. A spring relaxation algorithm minimizes the network utilization of a query while keeping the latency low.

Fan et al. (2020) used reinforcement learning to dynamically allocate resources to IoT tasks based on historical data. The sub-optimal decisions made in real time aim to minimize computational and communication delay.

7.2 Our approach

To enable resource-awareness in the IoT ecosystem a distributed data stream management system (DDSMS) is developed. As a base, an existing data stream management system (DSMS) is used which already provides mechanisms and abstractions to gain control over the data flow. Data stream management systems provide semantics for data streams and data steam operators enabling high-level query languages and data stream query optimization.

The network of distributed data stream processing nodes consist of DSMS nodes and smart sensors. Those nodes are controlled by a central unit, which provides holistic control for the whole network and is aware of all node and network properties like bandwidth utilization and latency. The DSMS nodes are fully fledged data stream management systems and capable of processing streaming data using predefined operators. Whereas the smart sensors provide an interface in order to remotely configure basic edge processing on the sensor itself including select, aggregate and filter operators.

In the central unit, different distribution strategies can be implemented which optimize the resource utilization for specific parameters according to a cost model like proposed by Wang et al. (2009). A resource monitor tracks the performance of a global query executed under a certain distribution strategy. The monitoring enables the evaluation of different operator placement strategies for specific use cases to minimize resource utilization for constrained devices or the overall system.

The existing data stream management system Odysseus Appelrath et al. (2012) is extended to implement this approach.

7.2.1 Resource-aware dairy cattle activity monitoring

For demonstration purpose the approach above is applied to the dairy cattle activity monitoring.

The models trained in the training pipeline are used to monitor the cattle behavior in the prediction pipeline. The prediction pipeline consists of the following steps:

1. Measurement via sensor system
2. Segmentation
3. Feature Calculation
4. Prediction

In this pipeline, there are many parameters, which need to be tuned to be resource-efficient. In step 1 the data is generated and it may be best for prediction accuracy to get as many data as possible but for resource-efficiency you want to have a good trade-off between data frequency and accuracy.

In step 2 the sensor data is segmented into windows where a good window size depends on the duration of the activities to recognize. However, the window stride parameter, which configures the amount of window overlap, has a direct impact on the computational costs. Here you want to have a good window stride/accuracy trade-off reducing computational costs arising from window overlap.

In step 3 the data points segmented in windows are aggregated to features. Optimizing the resource-efficiency of the feature calculation relies on a good feature selection strategy. You will get a higher accuracy including more features to the model but there might be a minimal feature subset which has a nice feature/accuracy trade-off reducing computational costs and still fulfilling accuracy needs.

In step 4 the trained models are fed with the features and the activity is predicted. Machine learning algorithms differ in computational prediction costs. Therefore, choosing a ML approach providing less accuracy but reducing resource utilization significantly is a viable optimization strategy in this step. As project complexity grows the amount of training models and their associated information becomes complex and usually re-usability and sharing drops due to lack of framework to manage this ML life-cycle process, paving the way to the *CModel* challenge.

Besides the distribution-agnostic resource optimizations mentioned above there is another important point to think about. It is crucial to decide where which computation will take place. The cloud processing approach which sends all the data from step 1 to the cloud and performs steps 2-4 there is obviously not optimal due to network bandwidth utilization. Computing steps 2-3 on the edge and step 4 in the fog or even steps 2-4 on the edge will perform better according to resource utilization.

The different processing steps in the prediction pipeline are implemented as data stream operators in a DSMS. According to the proposed approach above different distribution strategies can be evaluated in the monitoring system of the distributed data stream management system to find a resource-saving setup. Also the data stream operators implementing steps 1-4 can be parameterized to evaluate parameter settings for the single processing steps.

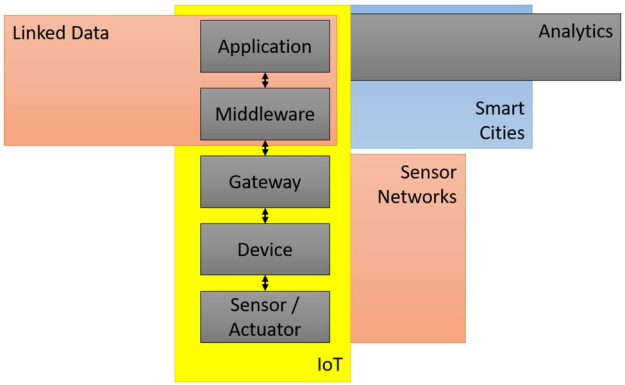


Fig. 5 Related work space

8 Related work

The landscape of the work related to our discussion here is varied and multi-dimensional. It includes topics such as: IoT, smart cities, data management, analytics, sensor networks, communication protocols, and others. One way we can understand this space would be to simplify it into a two-dimensional vertical and horizontal space. Vertically, we will use the IoT architectural structure of: sensor/actuator, device, gateway, middleware and application provided in Guth et al. (2016) as a guide, as well as the definitions they provide for these components.

Sensors and actuators are hardware components, whereas a sensor measures parameters of its physical environment, an actuator acts, controls or manipulates its environment. A device is a hardware component as well that connects to a sensor/actuator, it can process data from sensors or control actuators. Gateways can be used to compensate communication limitations of devices. IoT (Integration) middleware serves as a middle point between applications and devices/gateways, by processing or evaluating received data or sending commands to be executed by actuators Guth et al. (2016). Horizontally we can place orthogonal fields of study that intersect with IoT such as smart cities, big data analytics, distributed data processing, sensor networks, etc. Figure 5 depicts one possible interpretation of the related work space.

Gubbi et al. (2013) present an overall vision of IoT is presented that is influenced by wireless sensor networks. In a similar vain Pike et al. (2019) applies the intersection of IoT and sensor networks to the domain of healthcare. Alavi et al. (2018) apply the IoT approach to the smart cities domain and

focus on the middleware and application side of the problem, while using the motivating scenarios of smart cities to drive the design of their vision.

Moving on in the distributed data processing pipelines and streams, Hernandez et al. (2020) models intelligent data pipelines using an actor-based model for IoT-based applications. The focus here is to support application developers in building intelligent data processing actors in a common ecosystem. To continue the look into data streams in IoT, Elsaleh et al. (2020) present a lightweight ontology to model IoT data streams to support easy data analytics and event detection services.

Turning to existing IoT platforms that are beyond research work, we have recently been experimenting with ThingsBoard[5] and Cumulocity[6] and the possibilities of using them in our stack. Both these solutions implement the basic IoT requirements, and in retrospect we don't see our work as competing with them but as integrating with them to benefit from the rich support for communication protocols, and dashboarding.

Another horizontal dimension of work focuses on the representation of domain entities & assets relevant to the IoT platform, be they sensors, actuators, users, or others. For example, Mormul and Stach (2020) present a context model for holistic monitoring and management of complex IT environments, to be used in conjunction with the larger IoT platform. On the other hand, Sasidharan and Somov (2014) propose a framework where these assets are modeled as either: real world objects, virtual objects or composite virtual objects. In many ways such solutions have a similar aim in representing data about domain entities at different levels of the IoT platform.

The challenges we tackle here can be positioned in the space depicted in Fig. 5. The *CResource* challenge covers the continuous space of device, gateway, sensor networks, and middleware. The *CQuality* challenge touches on sensor, device, gateway, middleware, sensor networks, smart cities as well as linked data. The *CModel* challenge, touches on analytics, smart cities, middleware, applications, and linked data. Our approach toward the *CPrivacy* challenge crosses the IoT stack from sensors up to middleware. And finally, the approach we develop to IoT platform design falls in the linked data, middleware, and applications.

As distinguished from the related work discussed earlier, in this paper, we combine paradigms and focus on a set of data management problems at different levels of the data management stack. For example although an IoT platform is used as a hub for many of the data streams it is not the end goal in itself. Smart city scenarios act as a problem scenario to motivate the choice of problems to solve. Data is processed at different locations in the web of devices. With our focus on enriching data quality information, provenance and data descriptions, we provide new opportunities to work with data. Integrating that with the model management and learning, it becomes and multi-function data management tool-set that is applicable to both research and industry use cases.

The work presented here extends our previous discussion in Steuer et al. (2016) and adds new challenges such as model management, sensor data quality and knowledge management.

9 Conclusion and future work

From the discussion in this paper, we can see that there are many research challenges left for data management in IoT that go beyond the discussion of big data processing.

To optimize data flows across the available edge-fog-cloud infrastructure (*CResource*) to save energy and bandwidth, the system needs to be aware of the operator semantics. Techniques from the well-known relational algebra can be applied to regain control over these heterogeneous environments. However, the cost models used in this optimization step need to be re-defined to cover novel aspects like energy-consumption of operators or privacy constraints (if certain raw data is not allowed to leave a processing node). In addition, the algebra might need to be extended to cover novel operators like ML model based prediction.

Since most IoT systems are based on sensors, the achievable data quality needs to be considered not only during installation, but also online during operation of IoT systems (*CQuality*). If we leave this task to the applications, system-wide and consistent data quality control will hardly be achievable. While certain dimensions of data quality are application-specific, we can model general data quality dimensions and how they depend on installation context for sensors. This model can then be used to auto-generate online quality assessment within large-scale IoT infrastructures, thus dis-burdening the application developers from this tedious step.

Since more and more sensor-based IoT applications are based on machine learning (ML) techniques, ex. for activity recognition, the management of the trained ML models become part of the data management challenges (*CModel*). As we can see from the use case of mobility analytics and the dairy cattle use case, the continuous life-cycle of ML models raises the demand to keep the information about different ML over time and space, so that we can manage and automate all ML related tasks.

A crosscutting concern of large-scale IoT system is privacy (*CPrivacy*), since devices often collect raw data that

[5] https://thingsboard.io/.

[6] https://www.softwareag.cloud/site/product/cumulocity-iot.html.

could be used to derive sensitive information, which is not always needed or even allowed. We discussed that issue within the use case of mobility data: While individual mobility is highly sensitive, aggregated mobility is not. To fully leverage the insights that we could get from such aggregated mobility data, we need to develop trustable online techniques for anonymization. As with data quality, the task of proper data anonymization should be provided as standard functions from an IoT infrastructure so that its application is not dependent on the programming skill of single developer teams.

Finally, an approach to IoT platform design is needed to be able to integrate such heterogeneous data management functionalities under one roof. The approach we follow is influenced by meta-data management approaches to enable a higher level of system support within the IoT infrastructure. In many IoT application domains (ex. smart cities), these infrastructures span different organizations and stakeholders. Hence, we propose the IB model and data-sharing markets to support multiple groups, systems, and functionalities and integrate their data in an information-driven manner. The such structured information can be used both by automation system (ex. distributed query optimizers or data quality assessment) and by human stake-holders, like developers, operators, or even end users (ex. to gain transparency about installed systems in their work environment).

As we can see from this discussion, data management for large-scale IoT systems has still many unsolved challenges. While they all could be tackled by individual software developed and within application-code, it is key for long-term operation, maintenance, and transparency of such systems to get more and more support by frameworks and higher-level programming concepts like query languages. Like a database system that hides many implementation details like data distribution or index usage, future IoT infrastructures might as well provide a high-level interface with pre-build support for resource-aware query optimization, online quality assessment, ML model management, privacy-preserving data aggregation, and a cross-organizational knowledge management.

Funding Open Access funding enabled and organized by Projekt DEAL.

Compliance with ethical standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

Abbasi, M.A., Memon, Z.A., Syed, T.Q., Memon, J., Alshboul, R.: Addressing the Future Data Management Challenges in IoT: A Proposed Framework. Int. J. Adv. Comput. Sci. Appl. (IJACSA) **8**(5), 197–207 (2017)

Aboubakr, B., Steuer, S., Nicklas, D.: Towards Quality Aware Sensor Data Stream Processing in a Smart City Living Lab pp. 36–41 (2017). http://ceur-ws.org/Vol-1858/paper8.pdf

Alavi, A.H., Jiao, P., Buttlar, W.G., Lajnef, N.: Internet of Things-enabled smart cities: State-of-the-art and future trends. Measurement **129**, 589–606 (2018) https://doi.org/10.1016/j.measurement.2018.07.067. http://www.sciencedirect.com/science/article/pii/S0263224118306912

Alpaydin, E.: Introduction to Machine Learning. MIT Press (2020). Google-Books-ID: tZnSDwAAQBAJ

Appelrath, H.J., Geesen, D., Grawunder, M., Michelsen, T., Nicklas, D.: Odysseus: a highly customizable framework for creating efficient event stream management systems. In: Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems - DEBS '12, pp. 367–368. ACM Press, Berlin, Germany (2012). https://doi.org/10.1145/2335484.2335525. http://dl.acm.org/citation.cfm?doid=2335484.2335525

Atzori, L., Iera, A., Morabito, G.: The Internet of Things: A survey. Computer Networks **54**(15), 2787–2805 (2010)https://doi.org/10.1016/j.comnet.2010.05.010. https://linkinghub.elsevier.com/retrieve/pii/S1389128610001568

Batini, C., Scannapieco, M.: Data Quality: Concepts, Methodologies and Techniques. Data-Centric Systems and Applications. Springer-Verlag, Berlin Heidelberg (2006). https://doi.org/10.1007/3-540-33173-5. https://www.springer.com/de/book/9783540331728

Benabbas, A., Geißelbrecht, M., Nikol, G.M., Mahr, L., Nähr, D., Steuer, S., Wiesemann, G., Müller, T., Nicklas, D., Wieland, T.: Measure particulate matter by yourself: data-quality monitoring in a citizen science project. Journal of Sensors and Sensor Systems **8**(2), 317–328 (2019) https://doi.org/10.5194/jsss-8-317-2019. https://jsss.copernicus.org/articles/8/317/2019/

Benabbas, A., Hornig, H., Nicklas, D.: Semi-Automatic Ontology Population for Online Quality Assessment of Particulate Matter Sensors. In: I. Chatzigiannakis, Y. Tobe, P. Novais, O. Amft (eds.) Intelligent Environments 2018—Workshop Proceedings of the 14th International Conference on Intelligent Environments, Rome, Italy, 25-28 June 2018, *Ambient Intelligence and Smart Environments*, vol. 23, pp. 119–128. IOS Press (2018). https://doi.org/10.3233/978-1-61499-874-7-119

Benabbas, M.A., Steuer, M.S., Nicklas, D.: Towards Adaptive Sensor Data Quality Improvement based on Context Models. Workshop on Context and Activity Modeling and Recognition (CoMoReA) p. 6 (2020)

Buchholz, T., Schiffers, M.: Quality of Context: What It Is And Why We Need It. In: In Proceedings of the 10th Workshop of the OpenView University Association: OVUA'03 (2003)

Cao, J., Carminati, B., Ferrari, E., Tan, K.L.: CASTLE: Continuously anonymizing data streams. Dependable Secure Comput. IEEE Trans. **8**, 337–352 (2011). https://doi.org/10.1109/TDSC.2009.47

Chandra, A., Heintz, B., Sitaraman, R.: Optimizing Geo-Distributed Streaming Analytics. In: Sakr, S., Zomaya, A. (eds.) Encyclopedia of Big Data Technologies, pp. 1–5. Springer International Publishing, Cham (2018)

Chin, J., Callaghan, V., Lam, I.: Understanding and personalising smart city services using machine learning, The Internet-of-Things and Big Data. In: 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE), pp. 2050–2055 (2017). https://doi.org/10.1109/ISIE.2017.8001570. ISSN: 2163-5145

Compton, M., Barnaghi, P., Bermudez, L., García-Castro, R., Corcho, O., Cox, S., Graybeal, J., Hauswirth, M., Henson, C., Herzog, A., Huang, V., Janowicz, K., Kelsey, W.D., Le Phuoc, D., Lefort, L., Leggieri, M., Neuhaus, H., Nikolov, A., Page, K., Passant, A., Sheth, A., Taylor, K.: The SSN ontology of the W3C semantic sensor network incubator group. Journal of Web Semantics **17**, 25–32 (2012) https://doi.org/10.1016/j.websem.2012.05.003. http://www.sciencedirect.com/science/article/pii/S1570826812000571

Cui, L., Yang, S., Chen, F., Ming, Z., Lu, N., Qin, J.: A survey on application of machine learning for Internet of Things. Int. J. Mach. Learn. Cybern. **9**(8), 1399–1417 (2018). https://doi.org/10.1007/s13042-018-0834-5

Daum, M., Lauterwald, F., Baumgärtel, P., Pollner, N., Meyer-Wegener, K.: Efficient and cost-aware operator placement in heterogeneous stream-processing environments. In: Proceedings of the 5th ACM international conference on Distributed event-based system—DEBS '11, p. 393. ACM Press, New York, New York, USA (2011). https://doi.org/10.1145/2002259.2002327. http://portal.acm.org/citation.cfm?doid=2002259.2002327

Elmamooz, G., Finzel, B., Nicklas, D.: Towards Understanding Mobility in Museums. In: B. Mitschang, N. Ritter, H. Schwarz, M. Klettke, A. Thor, O. Kopp, M. Wieland (eds.) Datenbanksysteme für Business, Technologie und Web (BTW 2017), 17. Fachtagung des GI-Fachbereichs „Datenbanken und Informationssysteme" (DBIS), 6.-10. März 2017, Stuttgart, Germany, Workshopband, *LNI*, vol. P-266, pp. 127–134. GI (2017). https://dl.gi.de/20.500.12116/904

Elsaleh, T., Enshaeifar, S., Rezvani, R., Acton, S.T., Janeiko, V., Bermudez-Edo, M.: IoT-Stream: A Lightweight Ontology for Internet of Things Data Streams and Its Use with Data Analytics and Event Detection Services. Sensors **20**(4), 953 (2020) https://doi.org/10.3390/s20040953. https://www.mdpi.com/1424-8220/20/4/953. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute

Fan, Q., Bai, J., Zhang, H., Yi, Y., Liu, L.: Delay-aware Resource Allocation in Fog-assisted IoT Networks Through Reinforcement Learning. arXiv:2005.04097 [cs, eess] (2020)

Fung, B.C.M., Wang, K., Chen, R., Yu, P.S.: Privacy-preserving data publishing: A survey of recent developments. ACM Comput. Surv. **42**(4), 14:1–14:53 (2010)

Geisler, S., Quix, C., Weber, S., Jarke, M.: Ontology-Based Data Quality Management for Data Streams. J. Data Inf. Qual. **7**(4), 18:1–18:34 (2016). https://doi.org/10.1145/2968332

Golab, L., Özsu, M.T.: Issues in data stream management. ACM SIGMOD Record **32**(2), 5–14 (2003). https://doi.org/10.1145/776985.776986

Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of Things (IoT): A vision, architectural elements, and future directions. Fut. Gen. Comput. Syst. **29**(7), 1645–1660 (2013)https://doi.org/10.1016/j.future.2013.01.010. http://www.sciencedirect.com/science/article/pii/S0167739X13000241

Guth, J., Breitenbücher, U., Falkenthal, M., Leymann, F., Reinfurt, L.: Comparison of IoT platform architectures: A field study based on a reference architecture. In: 2016 Cloudification of the Internet of Things (CIoT), pp. 1–6 (2016). https://doi.org/10.1109/CIOT.2016.7872918

Heintz, B., Chandra, A., Sitaraman, R.K.: Optimizing Grouped Aggregation in Geo-Distributed Streaming Analytics. In: Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing - HPDC '15, pp. 133–144. ACM Press, Portland, Oregon, USA (2015). https://doi.org/10.1145/2749246.2749276. http://dl.acm.org/citation.cfm?doid=2749246.2749276

Hern, A.: Fitness tracking app strava gives away location of secret us army bases. https://www.theguardian.com/world/2018/jan/28/fitness-tracking-app-gives-away-location-of-secret-us-army-bases . Accessed: 2020-10-11

Hernandez, A., Xiao, B., Tudor, V.: ERAIA - Enabling Intelligence Data Pipelines for IoT-based Application Systems. In: 2020 IEEE International Conference on Pervasive Computing and Communications (PerCom), pp. 107–116. Austin, Texas, USA (2020)

Hirzel, M., Soulé, R., Gedik, B., Schneider, S.: Stream Query Optimization. In: Sakr, S., Zomaya, A. (eds.) Encyclopedia of Big Data Technologies, pp. 1–9. Springer International Publishing, Cham (2018)

Hirzel, M., Soulé, R., Schneider, S., Gedik, B., Grimm, R.: A catalog of stream processing optimizations. ACM Computing Surveys **46**(4), 1–34 (2014). https://doi.org/10.1145/2528412

Kamilaris, A., Gao, F., Prenafeta-Boldu, F.X., Ali, M.I.: Agri-IoT: A semantic framework for Internet of Things-enabled smart farming applications. In: 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT), pp. 442–447 (2016). https://doi.org/10.1109/WF-IoT.2016.7845467

Kaur, N., Sood, S.K., Verma, P.: Cloud resource management using 3Vs of Internet of Big data streams. Computing **102**(6), 1463–1485 (2020) https://doi.org/10.1007/s00607-019-00732-5. http://link.springer.com/10.1007/s00607-019-00732-5

Kavakiotis, I., Tsave, O., Salifoglou, A., Maglaveras, N., Vlahavas, I., Chouvarda, I.: Machine Learning and Data Mining Methods in Diabetes Research. Computational and Structural Biotechnology Journal **15**, 104–116 (2017) https://doi.org/10.1016/j.csbj.2016.12.005. http://www.sciencedirect.com/science/article/pii/S2001037016300733

Klein, A., Lehner, W.: Representing Data Quality in Sensor Data Streaming Environments. J. Data Inf. Qual. **1**(2), 10:1–10:28 (2009). https://doi.org/10.1145/1577840.1577845

Kuka, C., Nicklas, D.: Supporting quality-aware pervasive applications by probabilistic data stream management. In: Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems, DEBS '14, pp. 330–333. Association for Computing Machinery, Mumbai, India (2014). https://doi.org/10.1145/2611286.2611319

Lauterwald, F., Pollner, N., Daum, M., Meyer-Wegener, K.: Data Stream Application Manager (DSAM). In: Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems - DEBS '12, pp. 381–382. ACM Press, Berlin, Germany (2012). https://doi.org/10.1145/2335484.2335532. http://dl.acm.org/citation.cfm?doid=2335484.2335532

Mahdavinejad, M.S., Rezvan, M., Barekatain, M., Adibi, P., Barnaghi, P., Sheth, A.P.: Machine learning for internet of things data analysis: a survey. Digital Communications and Networks **4**(3), 161–175 (2018) https://doi.org/10.1016/j.dcan.2017.10.002. http://www.sciencedirect.com/science/article/pii/S235285481730247X

Mineraud, J., Mazhelis, O., Su, X., Tarkoma, S.: A gap analysis of Internet-of-Things platforms. Comput. Commun. **89–90**, 5–16 (2016) https://doi.org/10.1016/j.comcom.2016.03.015. http://www.sciencedirect.com/science/article/pii/S0140366416300731

Mormul, M., Stach, C.: A Context Model for Holistic Monitoring and Management of Complex IT Environments. In: Proceedings of the 2020 IEEE International Conference on Pervasive Computing and Communications Workshops (CoMoRea), pp. 1-1. IEEE Computer Society (2020). Backup Publisher: Universität Stuttgart, Fakultät Informatik, Elektrotechnik und Informationstechnik, Germany Type: Workshop-Beitrag

Patil, S.S., Thorat, S.A.: Early detection of grapes diseases using machine learning and IoT. In: 2016 Second International Conference on Cognitive Computing and Information Processing (CCIP), pp. 1–5 (2016). https://doi.org/10.1109/CCIP.2016.7802887

Pelekis, N., Theodoridis, Y.: Privacy-Aware Mobility Data Exploration. In: Pelekis, N., Theodoridis, Y. (eds.) Mobility Data Management and Exploration, pp. 169–185. Springer, New York (2014)

Pietzuch, P., Ledlie, J., Shneidman, J., Roussopoulos, M., Welsh, M., Seltzer, M.: Network-Aware Operator Placement for Stream-Processing Systems. In: 22nd International Conference on Data Engineering (ICDE'06), pp. 49-49. IEEE, Atlanta, GA, USA (2006). https://doi.org/10.1109/ICDE.2006.105. http://ieeexplore.ieee.org/document/1617417/

Pike, M., Mustafa, N.M., Towey, D., Brusic, V.: Sensor Networks and Data Management in Healthcare: Emerging Technologies and New Challenges. In: 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), vol. 1, pp. 834–839 (2019). https://doi.org/10.1109/COMPSAC.2019.00123. ISSN: 0730-3157

Rath, D.K., Kumar, A.: A Primer on Internet of Things Ecosystem and 5G Networks. In: 2018 International Conference on Information Technology (ICIT), pp. 233–238. IEEE, Bhubaneswar, India (2018). https://doi.org/10.1109/ICIT.2018.00055. https://ieeexplore.ieee.org/document/8724146/

Samie, F., Bauer, L., Henkel, J.: From Cloud Down to Things: An Overview of Machine Learning in Internet of Things. IEEE Internet Things J. **6**(3), 4921–4934 (2019). https://doi.org/10.1109/JIOT.2019.2893866. Conference Name: IEEE Internet of Things Journal

Sasidharan, S., Somov, A., Biswas, A.R., Giaffreda, R.: Cognitive management framework for Internet of Things: — A prototype implementation. In: 2014 IEEE World Forum on Internet of Things (WF-IoT), pp. 538–543 (2014). https://doi.org/10.1109/WF-IoT.2014.6803225

Schelter, S., Bießmann, F., Januschowski, T., Salinas, D., Seufert, S., Szarvas, G.: On Challenges in Machine Learning Model Management. IEEE Data Eng, Bull (2018)

Schmidt, S., Berthold, H., Lehner, W.: QStream: Deterministic Querying of Data Streams (2004). https://doi.org/10.1016/B978-012088469-8/50148-0

Schneider, S., Hirzel, M., Gedik, B.: Tutorial: stream processing optimizations. In: Proceedings of the 7th ACM international conference on Distributed event-based systems - DEBS '13, p. 249. ACM Press, Arlington, Texas, USA (2013). 10.1145/2488222.2488268. http://dl.acm.org/citation.cfm?doid=2488222.2488268

Sharma, M., Singh, G., Singh, R.: A review of different cost-based distributed query optimizers. Prog. Artif. Intell. **8**(1), 45–62 (2019). https://doi.org/10.1007/s13748-018-0154-8

Steuer, S., Benabbas, A., Kasrin, N., Nicklas, D.: Challenges and Design Goals for an Architecture of a Privacy-preserving Smart City Lab. Datenbank-Spektrum **16**(2), 147–156 (2016)

Vlacheas, P., Giaffreda, R., Stavroulaki, V., Kelaidonis, D., Foteinos, V., Poulios, G., Demestichas, P., Somov, A., Biswas, A.R., Moessner, K.: Enabling smart cities through a cognitive management framework for the internet of things. IEEE Commun. Mag. **51**(6), 102–111 (2013). https://doi.org/10.1109/MCOM.2013.6525602. Communications Magazine Conference Name: IEEE

Wang, S., Tan, Z., Gao, X.: Query Optimization over Distributed Data Stream. In: 2009 Ninth International Conference on Hybrid Intelligent Systems, pp. 415–418. IEEE, Shenyang, China (2009). https://doi.org/10.1109/HIS.2009.198. http://ieeexplore.ieee.org/document/5254496/

Weak passwords banned in california from 2020. https://www.bbc.com/news/technology-45757528. Accessed: 2020-10-11

Wu, W., Cheng, X., Ding, M., Xing, K., Liu, F., Deng, P.: Localized Outlying and Boundary Data Detection in Sensor Networks. IEEE Trans. Knowl. Data Eng.**19**(8), 1145–1157 (2007) https://doi.org/10.1109/TKDE.2007.1067. http://ieeexplore.ieee.org/document/4262542/

Xu, R., Palanisamy, B., Joshi, J.: QueryGuard: Privacy-Preserving Latency-Aware Query Optimization for Edge Computing. In: 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), pp. 1097–1106. IEEE, New York, NY, USA (2018). https://doi.org/10.1109/TrustCom/BigDataSE.2018.00153. https://ieeexplore.ieee.org/document/8456022/

Zanella, A., Bui, N., Castellani, A., Vangelista, L., Zorzi, M.: Internet of Things for Smart Cities. IEEE Internet Things J. **1**(1), 22–32 (2014). https://doi.org/10.1109/JIOT.2014.2306328. Conference Name: IEEE Internet of Things Journal

Zhu, J., Collette, M.: A dynamic discretization method for reliability inference in Dynamic Bayesian Networks. Reliab. Eng. Syst. Saf. **138**, 242–252 (2015) https://doi.org/10.1016/j.ress.2015.01.017. http://www.sciencedirect.com/science/article/pii/S0951832015000277

Zorbas, N., Zissis, D., Tserpes, K., Anagnostopoulos, D.: Predicting Object Trajectories from High-Speed Streaming Data. In: 2015 IEEE Trustcom/BigDataSE/ISPA, vol. 2, pp. 229–234 (2015). https://doi.org/10.1109/Trustcom.2015.588

Nasr Kasrin has had experience in both industry and academia. He completed an Engineering degree with a focus on Computer Science from the German University in Cairo (GUC), Egypt in 2008. He began his career working as a teaching assistant and researching in the cognitive sciences and artificial intelligence. After completing his masters project and publishing two papers from it, he moved on to work in a medium-sized organization that developed social media applications, where he worked on product design as well as software architecture. He later worked as a lecturer at The International University of Technology Twintech (IUTT), Sana'a, Yemen. Since 2015 he has been employed at the Chair of Mobile Software Systems / Mobility at the University of Bamberg, Germany, working in the area of data management. His current research interest is in designing distributed models for dataset sharing and exchange across organizations. He adopts an information-based approach where datasets are shared by sharing information about them.

Aboubakr Benabbas graduated from the TU Ilmenau in 2014 with a master's degree. Since 01.06.2014 he works as a research assistant at the Chair of Mobile Systems at the University of Bamberg. He is currently doing his doctorate at Uni-Bamberg with the research topic "Data quality in sensor-based applications" and is project leader of Living-Lab Bamberg, a research environment for sensor applications. Since 2015 he has also been a course advisor for the degree program "Master International Software Systems Science".

Golnaz Elmamooz Since June 2016, Golnaz Elmamooz has been a research assistant at the University of Bamberg at the Chair of Computer Science, in particular mobile systems. She received her Masters in Computer Software Engineering from Islamic Azad University, Najafabad, Iran in 2013. She worked on predicting type 2 diabetes using Bayesian classifiers. Her research interests are learning from sensor-based data and managing the Machine Learning models. It focuses on the continuous management of location-related data from sensors and other active data sources and their integration into so-called context-sensitive applications. She is currently working on data stream and ML model management technologies. These technologies can be applied to the areas of smart cities and smart farming.

Daniela Nicklas Since 2014, Daniela Nicklas is full professor at the University of Bamberg, Germany, and holds the Chair of Computer Science, in particular Mobile Software Systems / Mobility. Before that, she was a junior professor for database and internet technologies at the Universität Oldenburg and member of the Member of Executive Board in the Transportation division at the OFFIS institute for computer science. She came there from a PostDoc position at the Universität Stuttgart (2006-2008) where she also obtained her PhD in 2005, working on the integration of large-scale spatial context models for mobile applications. Her research interests are computer systems that bridge the gap between the physical world and the digital world. She focuses on the continuous management of data from sensors and other active data sources and their incorporation in so-called context-aware applications. Currently, she works on data stream management technologies. She applies these technologies to the domains of smart cities, smart factories, pervasive computing, intelligent transportation systems, and situational awareness in general. In 2009, she received the IBM Exploratory Stream Analytics Innovation "Award for Data Stream Technology for Future Energy Grid Control". She is a member of many programme committees and organizing committees of pervasive computing and database conferences and workshops (e.g., PerCom, MDM, BTW, ...), and of IEEE, ACM, and the German Gesellschaft für Informatik (GI).

Simon Steuer Since August 2016 he is working in the University Bamberg. Currently he is changing his reasearch directions from location privacy in data streams to hybrid location models in smart agriculture.

Michael Sünkel completed his master's degree in applied computer science at the University of Bamberg in 2017. Until 2018 he taught as an external lecturer for the chair for foundations of computer science. Since 12.03.2018 he has been working as a research assistant at the chair of mobile systems. There he is involved in the research network FutureIOT with the research focus on data management for sensor-based applications and distributed data stream management.