Comparing Apples and Oranges in IoT context: A deep dive into methods for comparing IoT platforms

Adriana Mijuskovic, Ikram Ullah, Rob Bemthuis, Nirvana Meratnia, and Paul Havinga

***Abstract*—Many researchers try to make a comparison between various IoT platforms based on specific requirements. However, none of the reviewed studies proposed a thorough analysis of the variety of comparative methods. Since there is a lack of comparison frameworks for IoT platforms, individuals or companies have difficulties when selecting a suitable IoT platform matching their associated business requirements. In order to support this selection process, a set of functional and non-functional requirements is identified. A framework containing methods in selecting an IoT platform is presented. The methodology is based on statistical and visualization techniques to recommend a suitable IoT platform. Five IoT platforms: Azure, AWS, SaS, ThingWorx, and Kaa IoT are studied to evaluate the performance of the framework. Different comparison methods are proposed, and multi-criteria decision analysis method was applied by using an Analytical Hierarchical Process (AHP). One of the methods clusters the functional requirements and compares the IoT platforms based on their ability in supporting a specific requirement or not. K-means clustering was applied to determine the clusters of functional requirements. The comparison was made based on hierarchical level of requirements per main requirement. The other methods use the following statistical tests: Error Bar test, One-way Anova Test, and Tukey's Honest Significant Difference Test. Based on the selected requirements, an approach is suggested which IoT platform can be used.**

***Index Terms*—IoT platforms, statistical comparative techniques, functional requirements, visualization.**

I. Introduction

INTERNET OF THINGS (IoT) is the interconnection of things with Internet as a fundamental communication system in order to provide smart connectivity between the users and the environment [1]. In an IoT ecosystem, IoT refers to uniquely identifiable objects and their virtual representations in an Internet-like structure. The term Internet of Things was coined in 1999 in the domain of supply chain management by Kevin Ashton [1]. IoT has the potential to provide enormous applications in many fields of life. Over time, IoT has been embedded in the everyday life in order to improve daily lifestyle. Similarly, IoT can substantially provide new business models and generate new opportunities to enhance our way of living. It has been predicted that number of devices connected to the Internet will reach 50 billion by 2020 [2]. This enormous growth in number of connected devices generates huge amounts of data that needs to be stored, processed, visualized, and used for decision making, which is commonly managed by IoT platforms.

Such a platform consists of hardware and software solutions, used as a basis to develop applications or services. A typical IoT platform, as shown in Figure 1, is a multi-layer technology that enables straightforward provisioning, management, and automation of connected devices within the IoT universe. It basically connects hardware, although diverse, to the cloud by using flexible connectivity options, enterprise-grade security mechanisms, and broad data processing powers [3]. The advantages of using IoT platforms are numerous. As mentioned in [1], IoT platforms accelerate user time to market and increase the potential to deliver distinguished products. The distinguishing characteristics of IoT platforms relate to their ease of use, simplicity, intelligence, and scalability [1]. The emerging IoT platforms in the market support the requirements of different application domains like smart home, utilities, manufacturing, health-care, transport, government, agriculture, and logistics [4].

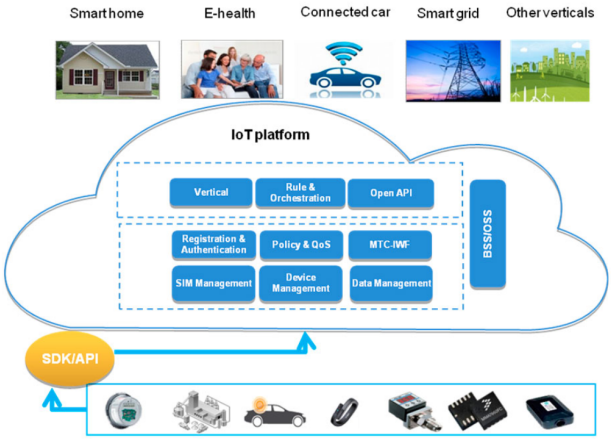


Fig. 1: IoT platform taken from [5]

There is a huge variety of different IoT platforms. They are quite extensive and present in different emerging fields. Therefore, there are many arising challenges when comparing them. Additional obstacle is to access, collect, and interpret the necessary information. It is also likely that the actual implementation of the tailored IoT solutions is different from the one presented in theory. One reason for this is that different IoT solutions are used only for specific industries. Due to

Manuscript submitted for review January 31, 2020.

Adriana Mijuskovic is with the EEMCS faculty, PS group, University of Twente, 7522 NH, The Netherlands, e-mail:a.mijushkovikj@utwente.nl (see https://people.utwente.nl/a.mijushkovikj).

The other authors are also with the EEMCS faculty, PS Group, University of Twente, 7522 NH, The Netherlands (e-mail: i.ullah@utwente.nl, r.h.bemthuis@utwente.nl,n.meratnia@tue.nl, p.j.m.havinga@utwente.nl)

the existing challenges when comparing the IoT platforms, at many points it seems like we are comparing apples and oranges.

In the literature a few studies investigate different comparison techniques that can be used by companies or individuals to choose the appropriate IoT platform. Additionally, none of the studies provides a clear and detailed taxonomy of functional and non-functional requirements which can be used as a basis for comparing the existing IoT platforms. In this paper, the current existing IoT platforms are explored by developing a new in-depth taxonomy for functional requirements and by providing a toolbox of techniques for comparative analysis. The developed taxonomy and comparative techniques will be used as a guideline for scholars and practitioners to select a suitable IoT platform.

This paper follows the guidelines of the Design Science Research Methodology (DSRM) as discussed in [6]. This methodology has been used to guide and structure the study, as reflected in the following sections. The problem statement and research objective are explained above. Section II describes the motivation for doing this research. IoT functional requirements are explained in detail in Section III, which contains the background material as part of the main design artifact. Section IV represents the taxonomy of IoT non-functional requirements, which can be used as a method for comparing IoT platforms. Section V introduces related work on IoT platform comparisons, which was the inspiration to conduct this research survey. Section VI presents the selected methods used to compare IoT platforms and provides a detailed description of the compared platforms, which is the main design artifact. The evaluation results of the proposed methodology are demonstrated in Section VII. Additionally, this section also provides an extensive discussion regarding the results. Section VIII suggests ideas for future work and discussion regarding the constraints of the conducted research. Section IX concludes this paper. More details about the application of DSRM can be found in the designated sections below.

II. Motivation

The technical advancements and reduced price of programmable hardware platforms, such as Arduino, Beagleboard, Intel Edison, Intel Galileo, Libelium, and Raspberry Pi2, renders IoT into a successful reality in every domain of life. Furthermore, the support for IoT from major IT companies like IBM, AWS, HP, Intel, Apple, and Google, shows the real potential of IoT. With the success of IoT, IoT platforms are becoming a mainstream technology and their potential capabilities are increasing in many application domains. Business cases can have diverse functional requirements depending on the services provided. Different IoT platforms support distinctive functional requirements at various levels of scale and cost. Thus, it is relevant to select an IoT platform according to individual's needs and requirements.

The aim of this work is to provide novel comparison techniques to overcome the existing cumbersome procedures of selecting the appropriate IoT platforms, since certain platforms are strong in specific requirements but weak in others. Hence, this article presents (1) a detailed taxonomy of functional requirements, (2) a detailed taxonomy of non-functional requirements, (3) a set of statistical and visual techniques to perform comparative analysis on functional requirements, and (4) a comparison of IoT platforms, covering a wide spectrum of functionalities and available platforms. As further discussed in the related work section (Section V), the main design artifact comprises a structured approach to recommend an appropriate IoT platform. The major focus of this research is initially on selection of set of statistical and visual techniques to perform comparative analysis, and conducting the comparison of IoT platforms.

III. IoT architecture functional requirements

The definition of requirements is a careful assessment of the needs that a particular system needs to fulfill [7]. It must refer to the reason why the system is needed based on the current or foreseen conditions, which can be internal operations. It should address what features the system will serve and satisfy. Functional requirements describe what the platform should do [8]. They are mainly statements of the specific services that the system should provide, how the system reacts to inputs, and how the system should behave. As opposed to functional requirements, non-functional requirements describe the constraints of the services of a software system [8]. These are not directly describing the specific services, but how well the system is performing. However, this study focuses on a selection of functional requirements as a basis for development of IoT platform architectures. A detailed taxonomy of functional requirements is developed, which can be used as a reference to select a suitable IoT platform and to compare various IoT platforms. A total of sixteen main functional requirements are defined. After a review of other related research studies and journals [4], [9], [10], a more in-depth taxonomy is created, which resulted in formation of 67 sub-classes of functional requirements. Adhering to the DSRM [6] as methodology followed throughout this paper, the classification of functionalities mainly based on literature reviews is determined. Further motivation is given in the designated sections to motivate why certain (sub-)classifications are made.

The resulting taxonomy of functional requirements is presented in Appendix A. The main functional requirements are organized into three main perspectives and are discussed below. The functional requirements in this section are discussing the authors' perspectives in terms of how these requirements can be classified. The authors decided to distinguish three perspectives.

A. Management Perspective

The management perspective perspective includes device management, platform management, network management, system management, deployment management, and resource management requirements.

1) Device management: The device management is required to remotely monitor and manage IoT devices. There is, for example, a standard developed by Open

Mobile Alliance (see [11]), which provides interfaces between Machine-to-Machine (M2M) devices and servers [9]. NETCONF Light protocol [12] can be used for device management and supply methods to install, manipulate, and delete configurations of network devices [9]. Furthermore, OMA device management working group [11] has developed specifications of protocols and methods for management of devices and services in resource constrained environments. The device management requirements are divided into several sub-classes including resource discovery, interoperability, portability and energy awareness.

2) Platform management: The platform management requirement is essential, because it is responsible for processing of the information exported in decision-making processes. Processing techniques can be used to extract and use the information, thereby converting the data into meaningful knowledge, which then can be used by users accessing the platform. Two important procedures are part of the platform management. The first one serves the consumer of the middleware component and the second one behaves as a producer. In the first decision making stage, the data processing techniques are implemented while in the second one, they are being completed. Various algorithms can be applied for the intelligent data processing, events, and decisions. The platform management requirement can be further divided into several sub-classes of requirements: interoperability, energy-awareness, fault-tolerance, and disaster recovery.

3) Network management: Network management is important to manage and administer networks for fault analysis and performance management [13]. The network management is divided into two sub-classes: type of supported communication protocols and network scalability. Platform users may have different devices with different capabilities and data communication requirements. Some of the well-known data communication protocols are: MQTT, COAP, AMQP, and Websocket. Scalability should also be addressed appropriately, as the number of interconnected devices increases substantially [13]. IoT platforms that do not support scalability do not accommodate the expansion of IoT network and such platforms would be unsuitable for many businesses.

4) System management: System management identifies how distributed systems are managed [14]. In this analysis, this requirement refers to whether the platform is centralized or decentralized.

5) Deployment management: The deployment management refers to the realization of the IoT platform. For this analysis, the deployment management is classified into operating system support, ease of deployment, and extensibility [15]. In an IoT ecosystem, devices are diversely heterogeneous in many aspects as well as supported in operating systems. Therefore, IoT platforms that support diverse number of operating systems are more suitable for heterogeneous environments than the platforms that only support one operating system. The platform deployment should be user-friendly, which means that it can be deployed and managed easily [15].

6) Resource management: The resource management describes how the platform monitors and manages the resources in an IoT platform [16]. It is further classified into the following requirements: resource discovery, maintaining resources, removal of resources, and resource management tool. Managing resources is important because it can lower energy consumption and maintenance costs [16].

B. Security and privacy perspective

The second perspective comprises security and privacy management requirements.

1) Security management: IoT networks are vulnerable to a large number of cyber attacks [17], [18], [19]. Some of the well-known attacks are: Denial of service, Distributed denial of service, Eavesdropping, Sybil, Black hole, Greyhole, Jamming, SYN flooding, Vampire, Wormhole, Hello flood, Node capture, and Camouflage. Trust is required to enhance the security in IoT. Some of the trust related attacks that can disrupt the trust in IoT are: Bad-mouthing, Self promotion, Ballot stuffing, Opportunistic service, and On-off attacks [20]. Furthermore, some of the IoT platforms are not secured by design. The security and privacy-related requirements mentioned below, should be present in IoT platforms and the requirements should be implemented according to the best practices, where applicable. The following best practices, refer to a set of standard effective policies, techniques, and procedures that strengthens the security in many different ways:

- Security auditability: Security auditability is a series of manual and systematic tests done to verify that security controls meet all the expectations and requirements. Security audit guarantees the renewal and correctness of the information security infrastructure.
- Accessibility: Accessibility refers to authentication, which is the process of verification that an individual, entity, or website is who it claims to be [21]. Secure authentication mechanisms are required to allow access only to authorized users. The authentication mechanisms should also follow OWASP authentication general guidelines [21]. As the usage of IoT devices expand in many application domains, scalable and interoperable accessibility mechanisms are required.
- Enhanced identification and authentication: Enhanced identification and authentication comprehend mechanisms using multiple factors to authenticate users. Unlike single factor authentication mechanisms, multiple factor authentication mechanisms are more secure because passwords are vulnerable to attacks such as guessing weak passwords, or phishing, while in multiple factor authentication mechanisms along with passwords other security factors are added for additional security. Additional factors could be: token, fingerprint, facial, or behavior. For enhanced security, robust multi-factor authentication schemes [22] are required.

- Data-at-rest encryption: Data-at-rest encryption ensures the confidentiality and integrity of the data. Encrypting data at rest is vital. Secure cryptographic algorithms need to be supported in order to protect the data at rest. Data at rest can be local data storage or cloud data storage. Best practices related to data encryption are provided by National Institute of Standards and Technology [23].
- Real-time detection and alerting: Real-time detection and alerting refers to traffic analysis used to inspect network traffic and to identify anomalous behaviors that can pose threat to the network. In case any anomalous behavior is identified, real time alert mechanisms are necessary to prevent potential threats. Therefore, real-time analysis of the network traffic and alarm generation for suspicious events and behaviors are required [24].
- Security risk management: Security risk management considers security risks that need to be identified. This impact needs to be assessed and plans to mitigate, remediate, and transfer the risks need to be made. Frameworks from NIST [25] may be helpful in addressing security risk management.
- Liability: Liability is about the responsibility of cyber attacks, which happen regularly. No matter how strong the security controls are, many enterprises become victims of cyber attacks. The impact of these attacks can range from data breaches to privacy loss [26]. Clear policies are required on whom to hold responsible in case different security breaches occur. In these policies, it should be clearly mentioned under which conditions the user or the IoT platform provider has the data ownership.
- Managed security services: Managed security services is about outsourcing the security in order to reduce the work burden and cost of in-house security teams and resources. At the same time, it is also used to utilize the expertise and to update tools and technologies of security service providers. For example, the U.S department of Defense has provided in [27] guidelines regarding outsourcing managed security services.
- Cloud resource segregation: Cloud resource segregation is acting upon the many security concerns that can arise when multi-tenants share the same memory and disk space. IoT platforms should support customer level cloud segregation, which means a separate memory and disk for each customer [28].

2) Privacy management: The platform should ensure the privacy of the users by adopting privacy preservation standards and frameworks. Some of the standards that can be implemented are: General Data Protection Regulations (GDPR) [29], Payment Card Infrastructure Data Security Standard (PCI-DSS) [30], Health Insurance Portability and Accountability (HIPPA) [31], and ISO 27018 [32].

C. Data management and analytics perspective

This third perspective considers data management and analytics, which consists of data management, services, analytics, IoT platform type, cloud architecture, and event management requirements.

1) Data management: Data management is the implementation of policies and procedures that put organizations in control of their business data regardless of where the data resides [33]. For this analysis, data management is classified into the following classes: data centers, presence in countries, backup solutions, and database management. Data centers describe how many data storage locations the platform has and its presence in different countries. This is important because some users might prefer their data not be stored in a foreign country. IT solutions are vulnerable to technical faults and security attacks, therefore it is crucial that the platform provides backup solutions. Through database management the platform should ensure that the data are organized and accessible.

2) Database management: IoT platforms should provide support for different types of databases in order to meet individual user needs:

- Relational Database: among the relational databases, SQL, MySQL PostgreSQL, SAP Hana, H2, MariaDB, and Oracle DB are commonly used databases in IoT platforms.
- Non-relational: key-value, document-based, column-based, in-memory database, and graph-based are within the group of non-relational databases.

3) Services: Services requirement illustrates the type of services: different types of solution packages, interfaces, firmwares, and visualizations provided by IoT platforms. With these requirements the IoT platforms can be differentiated in terms of usage convenience [34].

4) Analytics: Data analytics plays a significant role in the growth and success of IoT platforms. Analytics tools enable businesses to provide effective use of their data sets [35]. The selected sub-requirements of analytics are: advanced analytics and processing, algorithm support, prescriptive analytics, predictive maintenance, web services/web API, and mobile analytics solution.

- Advanced analytics: Advanced analytics implies that the IoT platform should include machine learning support, predictive analytics, decision management, and business intelligence [36].
- Algorithm support: Algorithm support refers to how many and which algorithms are supported by the selected platforms [37].
- Prescriptive analytics: Prescriptive analytics can provide recommendations to decision makers and help them achieve business goals by solving optimization problems [38]. It is dedicated to find the best course of an action for a specific situation.
- Predictive maintenance: Predictive maintenance techniques are developed to determine the condition of an in-service equipment in order to predict when a particular maintenance should be performed [39]. This approach would enable costs savings over time-based preventive maintenance, since activities and tasks are completed when warranted.

- Web services/Web APIs: Web services/Web APIs contain services that may receive data from gateways or directly from devices. Different web services may provide different type of functionalities such as data storage, data processing, data management, and data querying [40].
- Mobile data analytics solution: Mobile data analytics solution involves data analysis processes in mobile platforms, mobile sensors, and delivery to mobile customers. Mobile data can be used to extract certain patterns and context out of the information. Mobile data solutions are mainly focused on the use of mobile devices (smart phones, wearables, and vehicles) in performing data analysis [41].

5) IoT platform type: IoT platform type describes the different type of IoT platforms such as: Platform-as-a-Service (PaaS), Software-as-a-Service (SaaS), Infrastructure-as-a-Service (IaaS), and Machine-to-Machine (M2M) [42].

6) Cloud architecture: There are different types of cloud architectures: public, private, and hybrid [43]. It is important for the IoT platforms to support different types of cloud architectures. Users can select an architecture which is suitable for their needs or concerns like security and privacy.

7) Event management: Event management specifies how the IoT platform analyzes and visualizes the IoT devices data and generates events out of them. The applications of event management can be related to event stream processing and event notification [44]. Platforms that support event management are more suitable for real-time applications [44].

IV. IoT architecture Non-Functional Requirements

Similar to functional requirements, the non-functional requirements also play an important role in delivering the required capabilities in the IoT context. The non-functional requirements include: wireless communication protocol support, support, freedom of use, performance, costs, and development support. Each of these requirements is briefly described below. The statistical tests applied to the functional requirements can be similarly applied to the non-functional requirements. In this study, a taxonomy of non-functional requirements is provided, which can be used as a method for further comparison of IoT platforms.

A. Wireless Communication Protocol Support

IoT devices need to communicate with each other in order to collect and exchange data. IoT network uses both traditional communication protocols as well as protocols designed specifically for IoT. There are certain differences between these technologies like communication range, data rate, energy consumption, topology, network type, and operating systems compatibility. Some of the IoT technologies are briefly discussed below.

- Near Field Communication (NFC): NFC is a short range and low power consumption communication protocol. NFC allows a secure two way communication between devices by touching or bringing these devices close to each other. NFC makes transactions, exchanges data and connects devices in an easier manner. NFC operates in three main modes: card emulation mode (passive mode), reader/write mode (active mode), and peer-to-peer mode [45].
- Radio Frequency Identification System (RFID): RFID tags are attached to objects to identify them. RFID reader sends a signal to the tag, then the tag sends a response signal. The RFID uses the received signal to identify the objects. RFID tags can be passive, active, or semi passive active. The passive tags are not powered by a battery, whereas the active ones are. If required, semi-passive-active tags use power of the board [9].
- Bluetooth: Bluetooth uses short-wavelength radio to exchange data between the devices. The communication range is short and the power consumed is also low.
- Zigbee Zigbee is a mesh local area network protocol. It is suitable for applications that require low data rate, longer battery life and secure network devices. Compared to Bluetooth, Zigbee uses little power and has low data transfer rate [46].
- Z-wave: Z-Wave is a low-power communication technology, intended for home automation. Like Zigbee, Z-wave is also a mesh protocol in which all the Z-wave devices are connected together in order to form mesh network. It uses radio waves to communicate.
- WiFi: Wifi uses radio waves, allowing the devices to communicate and exchange data. Wifi is more suitable where large volumes of data transfer are required and high power consumption is not an issue.

B. Support

For the smooth usage and convenience, an IoT platform should provide its users with different types of support like documentation, forums, and knowledge base where the users and providers can share knowledge related to the platform. Also, the users should be able to directly interact with the technical support team via a call service.

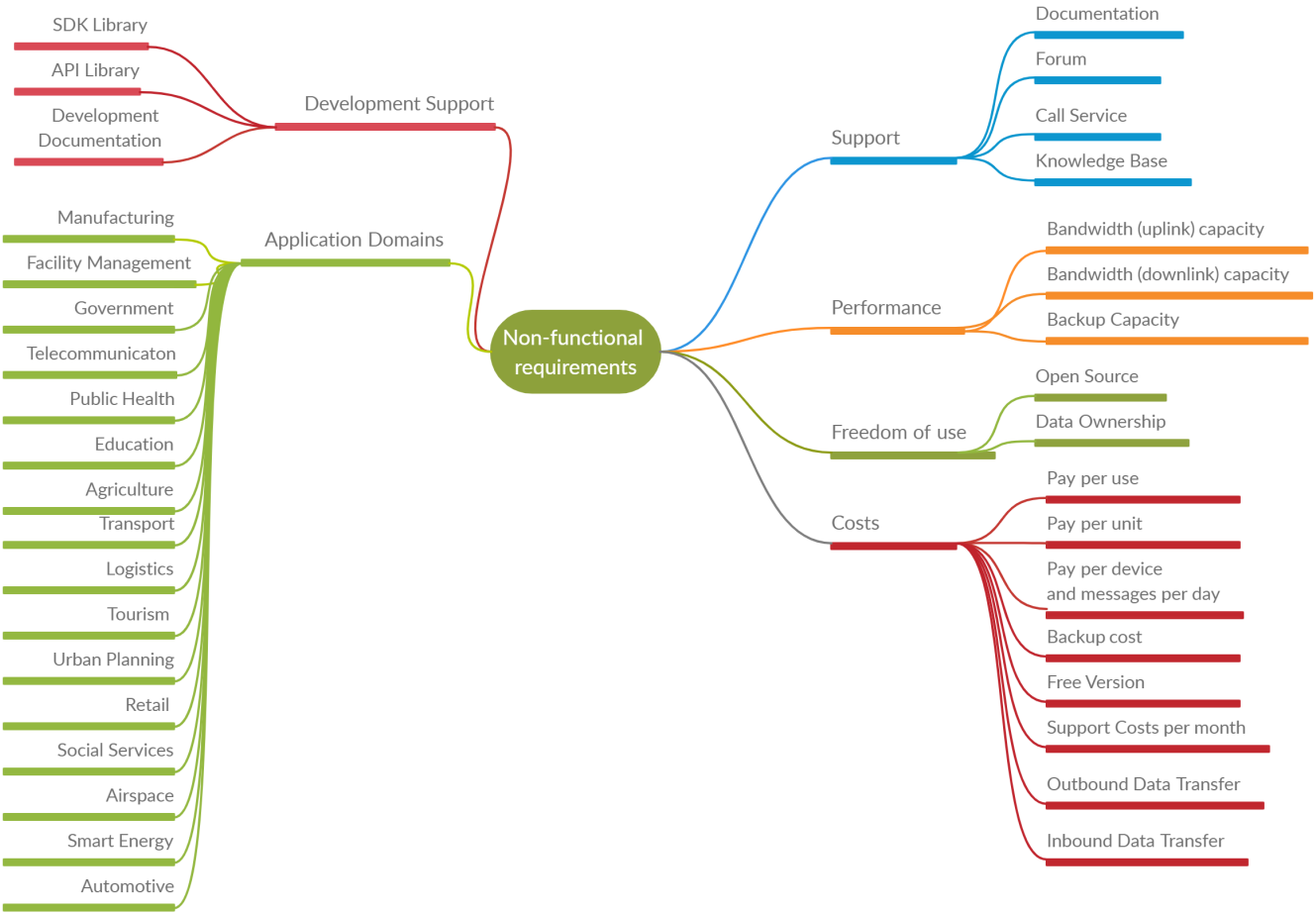
C. Freedom of use

The freedom of use describes the platform as a big open source or proprietary and ownership of the data. Freedom of use should be clearly mentioned in the policies, stating the owner of the user data, the user or the platform provider or any other third parties who have access rights.

D. Performance

For fast application performance, the maximum uplink and downlink bandwidth capacity is crucial for real-time applications and applications that have a large number of users. Also, the platform should support different types of backup solutions and maximum backup storage capacity. Therefore, IoT platforms having maximum uplink, downlink, and backup storage are favourable and suitable for many different users' applications.

Fig. 2: Non-functional requirements



E. Costs

Different platforms have different cost structures such as free, on demand and pay-as-you-go. Charging users for the services they really need and how much they need is one of the best cost strategies. As different platforms provide different services, solutions, and have different cost models, it is challenging to compare them.

F. Development support

Commonly, users would like to develop their own customized applications on top of the existing IoT platform application or integrate the platform with external applications. Therefore, it can be desirable that the IoT platforms deliver a development support. The sub-requirements in the specified development support are: SDK library, API library, and development document.

G. Application domains

The application domains of IoT are huge e.g.: public health, education, logistics, agriculture, transport, smart energy, consumer and retail industry, facility management, government, telecommunication, social services, urban planning, manufacturing, aerospace, defense, and automotive. A comparative analysis of IoT platforms on the basis of application domains can be beneficial, especially for users or companies to select a platform that supports their application domain.

V. Related Work on IoT Platform Comparison

Comparing IoT platforms is like comparing apples with oranges, because the IoT platforms can significantly differ from each other. Some cover broad application domains, while others are targeting specific (niche) applications. There are many challenges in performing this type of comparison. One of them is the lack of publicly available documentations related to the IoT platforms. Another challenge is the need to identify concrete requirements to be able to precisely perform the comparison of IoT platforms. The related work is limited. Most of the articles define a new taxonomy of the functional requirements or compare specific IoT platforms based on several features. No articles where IoT platforms were compared by using multiple different (statistical) methods were found. Nevertheless, in the literature, there are many researchers who made attempts to perform this comparison. For example, Solapure and Kenchannavar [47] emphasized the importance of IoT platforms and discussed the gap issues of IoT architectures and platforms to help different users to select a platform according

to their needs. IoT requirements and different IoT architectures were also presented. One of the limitations of their work is that the classification of the platforms is done on the basis of a few and general characteristics such as open source, cloud based, centralized, and type of tools used.

Mahalank et al. [48] performed analysis of only non-functional requirements in the design of an IoT based smart traffic management system. The non-functional requirements considered were cost, sensitivity, design complexity, storage capacity, development process, response criteria, and environmental impact. Danylenko and Lowe [49] proposed a recommendation system based on context-aware composition in order to allow system designers to execute automated decisions related to non-functional requirements during system design. The proposed recommendation system is specifically focused on non-functional requirements such as performance, dynamic memory consumption, and memory footprint of the code.

In [28], security, flexibility, and data intelligence are considered as important requirements in IoT platforms. A concrete description is given on how to achieve these requirements, but beyond them, there are many other requirements that are also critical and should have been considered. Oliveira et al. [50] compared IoT platforms on the basis of middleware requirements. Functional and non-functional requirements of middleware considered in this work are: resource discovery, event management, resource management, code management, data management, scalability, timeliness, reliability, popularity, availability, security and privacy, and easy-of-deployment.

Tayeb et al. [17] reviewed high-level conceptual layered architectures for IoT from a computational perspective. Fog computation was included in the architecture to address issues associated with cloud computing. IoT requirements mentioned include: connectivity, low power, cost-effective microprocessor, limited storage, security, reliability, and sensing. No comparison is done in this research study. To solve the heterogeneity-driven problems, Kim et al. [51] proposed a security generic service interface based on common characteristics of IoT platform. The platforms mentioned in their study are: Axeda, Everything, ThingWorx, and Eclipse M2M. This work is about how to remove the heterogeneity barrier in IoT environments by proposing an interface to automatically perform service configuration, service discovery, and service re-usability.

Patti et al. [52] proposed requirements that need to be considered in order to develop an IoT platform for Smart City. The considered requirements include: interoperability, real-time data collection, real-time data transmission, micro-service approach, expose Web services and API, and awareness to end-users. Two IoT platforms for energy management in Smart city were introduced and platforms were compared based on these requirements.

Mineraud et al. [47] compared 39 proprietary and open-source middleware solutions based on the characteristics considered by authors as fundamental to meet the expectations of both application developers and users. These characteristics include: support of heterogeneous devices, type of platform, architecture, open source, support REST, data access control, and service discovery. A gap analysis was presented, which was identified in the functionality of these middleware solutions. The aim of the gap analysis serves to assess the maturity of the current solutions by studying their short comings along several dimensions. The dimensions included in this analysis were: support of heterogeneous devices, data ownership, data processing and sharing, developer support, ecosystem formation, and IoT marketplace. Recommendations for the development of IoT middleware were provided, in order to address the gaps identified in the existing middleware solutions. The evaluation is focused on middleware solutions, while our research is focused on (more) generic IoT platforms.

Kondratenko et al. [53], used the existing methods of multi-criteria decision making for selecting a rational IoT platform. This work presents the application linear convolution and multiplicative convolution methods using two different approaches to forming weight coefficients for criteria: simple ranking and proportional. The authors provided a specific criteria and alternatives for selection of a suitable IoT platform. Based on the knowledge obtained from other surveys, the authors selected the following criteria as being relevant when choosing an IoT platform: device management, integration level, level of safety and reliability, protocols for data collection, variety of data analytics, usefulness of visualization, and database functionality. The platforms that have been considered in their comparison are: AWS IoT platform, Kaa IoT, IBM Watson IoT, Microsoft Azure IoT, and Bosch IoT Suite. The results of applying methods of multi-criteria decision making to select a rational IoT platform show that Microsoft Azure IoT platform is the most rational from the point of view of DM.

Liviu Dimitru et al. [54] conducted an online survey of open source IoT platforms which are used by developers for deployment of IoT projects. Open-source platforms were used for reasons such as cost, scalability, and control. In this study, five criteria approach was applied when using the open-source platforms such as: technology ownership, costs, interoperability, and integration with third party solutions. The online survey included five main functionalities such as device management, integration, security, protocols for data collection, and analytics types. The analyzed IoT platforms were: Kaa IoT, Particle cloud for Raspberry Pi, CARRIOTS IoT platform, Everything IoT platform, and TEMBOO IoT platform. Based on Arduino and Raspberry Pi development platforms users, the research was using online questionnaires that were sent to a total of 30 users asking which of the nominated platforms were used in their projects and a total of 18 users responded to those questionnaires. The paper discusses that for the user it is important that the interface is friendly and accessible regardless of the device on which it is deployed and regardless of the operating system used by the device.

VI. Methods to compare IoT platforms based on functional requirements

IoT applications are evolving rapidly and IoT platforms have also gained much attention recently. As the number of available IoT platforms is increasing, selecting the right IoT platform to fulfill specific requirements can become

challenging. Appropriately, techniques to identify the most effective and/or efficient platform are essential, especially for fulfilling the business and user requirements. In this section, a comparison procedure and the proposed methods will be described.

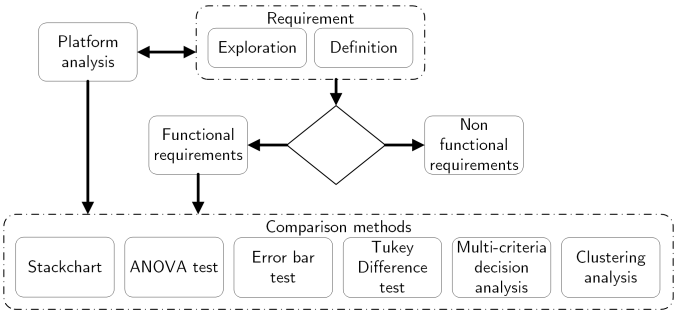


Fig. 3: Overview of provided toolbox

To compare IoT platforms based on the functional requirements as discussed in Section III, a set of functional requirements that can be used to compare the platforms is identified. Requirements studied in this work broadly cover elements that are of importance for IoT platforms. Five different IoT platforms are selected to implement the proposed comparison methods. There are multiple methods that can be applied when comparing the selected IoT platforms (known as "apples and oranges"). The following methods from Figure 3 are used:

- Comparison of IoT platforms represented via a stackchart;
- ANOVA test;
- Error bar test;
- Tukey Difference test;
- Multi-criteria decision making (in this case Analytical Hierarchical Process, AHP);
- Clustering analysis.

A. Data acquisition and description

The corresponding data regarding these platforms are collected as follows. Data are collected from the official IoT platforms' websites and online documentation. The available documentation is studied and the authors assessed whether the platform supports a specific requirement. In some situations, direct contacts with the platform providers is made to acquire extra (non publicly available) information.

Data are acquired in order to create a taxonomy for functional and non-functional requirements. The developed taxonomy for functional requirements is used as a base framework for multiple methods to compare IoT platforms. The complete dataset, given in Table VI, from Appendix A, is consisted of 16 main functional requirements and 67 sub-requirements, and 5 features (i.e., columns) of different IoT platforms (AWS IoT, Azure IoT, SAS IoT, Thingworx IoT, and Kaa IoT). The requirements support refers to the labels that contain values, where a value of 1 indicates that the platform supports a requirement and NA means that either the platform does not support the requirement or the data are not publicly available by the platform provider.

B. Data analysis methods

The analysis of the data has been done by using RStudio [55] and Excel. Firstly, the platforms based on the prediction were scored. The initial result is a taxonomy of requirements represented in Table VI from the Appendix A. The following techniques are used:

- The data from Table VI were used to create the **stackchart**.
- **Anova test** to compare the variance between the different groups together with the variability within each of the groups. An Anova test indicates whether the platforms are different from each other based on the total number of supported requirements. In this comparative analysis, the strength of each platform is measured based on the total number of requirements supported out of the total sixteen main functional requirements. The output of one-way Anova test can be correlated with the Table VII shown in the Appendix B. Table VII demonstrates the means values which are calculated for each main class based on the number of sub-classes supported. A platform having the mean value equal to 1 is considered strong, if the mean value is between 0 and 0.6 it is considered weak, while a platform having mean values between 0.7 and 0.9 is considered as average.
- **Error bar test** was performed by using data from Table VII of Appendix B, to understand the variability in each requirement class among the compared platforms. By variability is referred to the strength's fluctuation of the requirement support.
- **Tukey's honest significant difference test** to pairwise compare platforms by examining the difference in terms of total number of supported requirements.
- **Multi-criteria decision-making** (MCDM) to assess several alternative options based on the presence of many decision criteria [56]. MCDM involves a general class of operations research models (e.g., formal approaches) which attempt to take explicit account of multiple often conflicting criteria in making an appropriate decision. Some advantages of using these techniques are that they provide structure, give focus and language for discussion, help to learn about the problem, values, and constraints, and assist in justification and communication [57]. A wide range of MCDM methods exists. In this study, the Analytical Hierarchy Process (AHP) method will be demonstrated. This method is a widely used MCDM and can be carried out based on managerial judgement. AHP decomposes the decision problem into subproblems and analyzes the subproblems individually, by means of pairwise comparison of each criterion [58]. The final ranking is based on the global weights.
- **Cluster analysis** was done by using a K-means clustering algorithm, which was performed on the existing data from Table I, which emerged from Table VI shown in Appendix A. Table I shows the initial assumptive classification, which refers to the number of sub-requirements supported by each IoT platform per requirement.
  In Table I, the numbers 1-16 represent the numbering

TABLE I: Predictive classification of requirements in three main perspectives

Functional requirements	Perspectives	AWS	Azure IoT	SaS	Thing-Worx	Kaa
1. Device management	1	4	4	0	1	2
2. Platform management	1	4	3	3	1	3
3. Network management	1	2	2	2	2	2
4. Data management	3	4	3	3	4	1
5. Relational database	3	5	4	5	6	3
6. Non-relational database	3	4	3	1	2	3
7. Deployment management	1	2	3	2	3	1
8. Services	3	5	5	4	4	4
9. Analytics	3	6	6	5	5	2
10. System management	1	1	1	1	1	1
11. Event management	3	2	2	2	2	2
12. Security management	2	10	11	8	9	3
13. Privacy management	2	1	1	1	1	1
14. Resource management	1	4	4	4	2	1
15. IoT platform type	3	3	4	3	3	3
16. Cloud architecture	3	3	3	3	2	0

of the functional requirements, the feature Perspectives refers to the group(class) in which the requirements can be classified, while the values (0-11) shown in all five features (AWS, Azure, SAS, Thingworx, and Kaa IoT) represent the total number of supported sub-requirements per main requirement. Then, K-means clustering was performed to validate the initial assumption regarding the selected number of requirement perspectives.

Cluster analysis has been performed in order to initially cluster the requirements and then to identify which IoT platform performs the best in a specific cluster of requirements. This will help the users and service providers to identify in which set of requirements a specific IoT platform performs the best and for what type of IoT architecture the platform is suitable for. The use of the clustering analysis is more to help us in performing the classification of the requirements and then the user can be inspired to perform an additional comparison of the IoT platforms per cluster (specific group of requirements).

This method aims to identify the clusters of functional requirements and to assess whether and to what degree an IoT platform supports specific functional requirements. The procedure is as follows:

1) Count the sub-requirements supported by each platform per main requirement.
2) Conduct a K-means clustering method, to determine the number of clusters. To this end, the Elbow method is used. This method is based on assessing a plot of the within-groups sum of squares against the number of clusters [59]. Additionally, this method gives the user an idea on what a good number of clusters would be, based on sum of squared distance (SSE) between the data points and their assigned clusters' centroids. K is selected at the spot where SSE starts to flatten out and forms an "elbow". SSE value was calculated of the specified cluster [59]. The number of clusters is based on a visual inspection of the graph, by checking up to which number of clusters the difference of the sum of squares with regard to using one cluster less is not yielding any significant lower sum of group squares.
3) Afterwards, a cluster solution is applied by using K-Means Cluster Analysis and a new data frame is retrieved, which includes a new column fit-cluster. The new data frame after the performed clustering analysis, is shown in Table V, where the notations from 1 to 16 refer to the sequence of functional requirements. The five IoT platform features contain values of the total sub-requirement support per main requirement and the fit-cluster (values 1-3) refers to the class where the actual main requirement belongs to.

Clustering is considered as an unsupervised learning method, since there is no ground truth to compare the output of the clustering algorithm to the true labels that are used to evaluate its performance. It is required to investigate the structure of the data by grouping the data points into specific subgroups. K-means is a fast and a simple clustering method with a smaller number of iterations [59]. The algorithm divides the data into k section. The distance between each object and the center of each cluster is calculated and resulted in an optimal cluster solution. Objects within a particular cluster are adjacent to each other.

C. Data visualization

Several data visualization methods and tools were used to visualize the retrieved results. Google Charts, D3.js, and WebStorm were used for creating interactive visualizations. Sankey diagrams, offered by Google Charts, were used as a solution to visualize the direct relations between the two entities: requirements and IoT platforms. Additionally, non-interactive visualizations (e.g., stack charts and 3D plots) were generated by using RStudio libraries [55]. The stack chart shows the comparison between the IoT platforms based on sub-requirement support per functional requirement. Grouped horizontal bar chart and Sankey diagrams were displayed to visualize the comparison between the different platforms and determine which IoT platform performs the best in a specific cluster of requirements. Then, the user or service provider can decide which IoT platform is suitable for a particular architecture (management, security, and/or resource management). Grouped horizontal bar chart shows the side by side comparison of each and every requirement in all compared platforms.

D. Compared IoT platforms

The selected platforms for comparison are: AWS, Azure, SaS, Kaa, and ThingWorx. These platforms were selected based on their popularity and wide use.

1) AWS IoT: AWS IoT [60] provides IoT services such as AWS GreenGrass, AWS IoT Core, AWS IoT device management, AWS IoT Analytics, Amazon FreeRTOS, AWS IoT 1-Click, AWS IoT Button, and AWS IoT Device Defender. These IoT services help in data collection and, consequently, in transferring the data to the cloud. It provides services that include data loading with ease-of-use, data analysis, and device management [60].

2) Azure IoT: Azure IoT Suite [61] is an IoT solution in the cloud. This IoT solution architecture contains three main components: device connectivity, data processing and analytics, and presentation [61]. The devices are dedicated to sending telemetry to the cloud for data storage and processing. The devices can also receive and respond to messages arriving from the cloud end-points. Security of the connected devices is one of the major challenges for IoT solutions. Data processing can either occur in the cloud or on the device itself (also known as edge computing). The third component refers to presentation and business connectivity. This layer provides the users with the ability to interact with the IoT solution and the devices. The presentation component is a view of the analyzed data, which is collected from devices. The views are actually a form of a dashboard or BI reports. Azure IoT provides a number of Azure services and solutions such as Azure Cosmos DB, IoT Hub, Machine Learning Studio, Stream Analytics, Event Hubs, Notification Hubs, Logic Apps, IoT Suite, Time Series Insights, Event Grid, IoT Edge, Azure Location Based Services, and Azure Sphere.

3) SAS IoT: SAS IoT Solutions [62] covers wide spectrum of analytics, from data discovery to deployment, by involving data visualization, machine learning and streaming analytics, which is in the data center or at the edge. SAS offers several IoT solutions: SAS Analytics for IoT, SAS Quality Analytics Suite, SAS Asset Performance Analytics, SAS Real-Time Decision Manager, SAS Customer Intelligence, SAS Visual Analytics, SAS Visual Statistics, SAS Event Stream Processing, and Data Management Software.

4) ThingWorx IoT: ThingWorx IoT [63] is an IoT platform that supports industrial enterprises in quick development, deployment, and extension of IoT apps and AR experiences. ThingWorx supports App. accelerators such as ThingWorx Navigate, ThingWorx Asset Advisor, and ThingWorx Manufacturing Apps.

5) Kaa IoT: Kaa IoT [64] is a multi-purpose middleware platform for the IoT, which allows building end-to-end IoT solutions, connected applications, and smart products. It is a platform that provides a toolkit for product development and it claims to reduce time, risks, and costs [64]. The key capabilities of Kaa IoT are: managing a large number of devices, cross-device interoperability, A/B service testing, perform remote-device provisioning, real-time device monitoring, cloud services for smart products, collecting and analyzing sensor data, analyzing user behavior, and distributing over-the-air firmware updates.

VII. RESULTS

This section discusses the results of a set of statistical and visual techniques, which was used to perform the comparative analysis. Each of these (statistical) methods is evaluated with the same number of data samples.

A. Results of Comparative Methods

1) Stackchart Results: The stack chart as shown in Figure 4, illustrates that AWS IoT and Azure IoT have the highest support for the functional requirements, while Kaa IoT has the lowest. The clustering of the functional requirements was not considered when comparing the IoT platforms.

2) One-way Anova test: Table II shows the one-way Anova test output. By analyzing results of one-way Anova, it is possible to conclude that we can reject the null hypothesis ($p<0.05$) and accept the alternative hypothesis. Null hypothesis is that all the platforms are the same. Rejecting null hypothesis means that not all platforms are scoring the same.

TABLE II: Analysis of Variance Table

	Df Sum	**Sq Mean**	**Sq F**	**value**	**Pr(>F)**
Platforms	4	1.4095	0.35238	5.3022	0.000815
Residuals	75	4.9844	0.06646		

3) Error bar test: Figure 5 shows the results of Error bar test. The aim of this error bar test is just to demonstrate how much each of these requirements differs among the platforms. Smaller error bar indicates the low spread of the data around the mean while larger error bar communicates the larger variability from the mean value. All platforms have the same value for the network management requirement, so the variability around the mean value is zero, which indicates all the platforms are equally supporting these requirements. Whereas, for security management, a larger error bar can be seen, which indicates that some platforms have high value for this requirement while others have low. By analyzing the dataset, Azure is strongest in security management by supporting all the security sub-requirements, while Kaa IoT is weakest in security by supporting just 3 sub-requirements. With this statistical analysis users can concentrate only on the requirements that are varying and select platform accordingly.

4) Tukey's Honest Significant Difference Test: One-way Anova test only illustrated that these platforms are not the same in the sense of supporting the requirements, but it does not tell how much each platform is different from the other ones. Therefore, Tukey's honest significant difference test as shown in Table III is performed to show the pairwise difference between the platforms. Additionally, one may observe other differences between the platforms.

The Tukey's difference, as presented in Table III and Figure 6, shows the pairwise comparison between the platforms. Each platform is compared with the other four platforms. Figure 7 shows mean values comparison based on data from Table VII. By analyzing Figure 6 and Figure 7 a comparatively significant difference between Kaa IoT and AWS IoT can be noticed and similarly there is a significant difference between Kaa IoT and Azure IoT. AWS IoT and Azure IoT are comparatively similar and SaS IoT and ThingWorx IoT are also almost similar. These results can be correlated with the mean values of the platforms shown in Table VII from Appendix B, whereas the mean values of AWS and Azure on

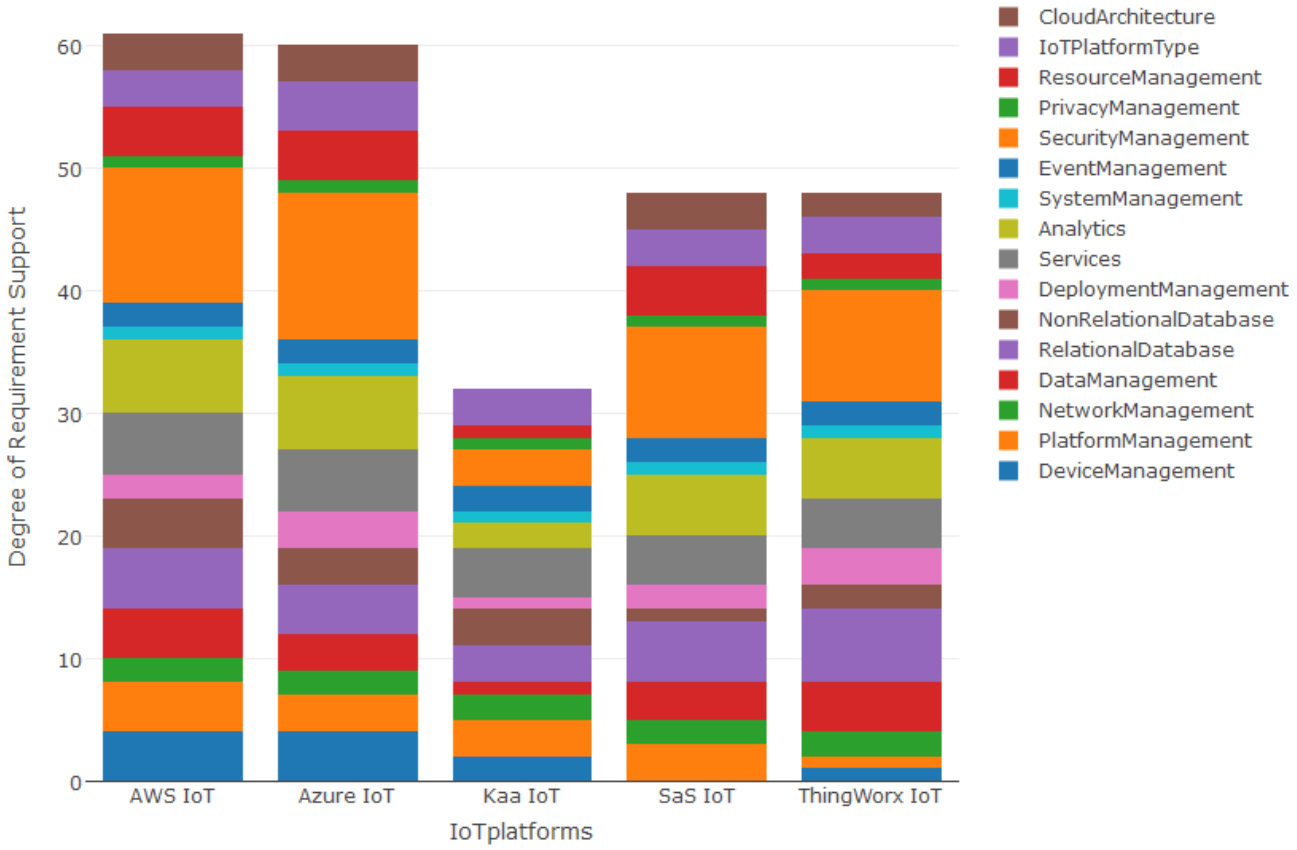


Fig. 4: Comparison of IoT platforms based on Degree of Requirement Support

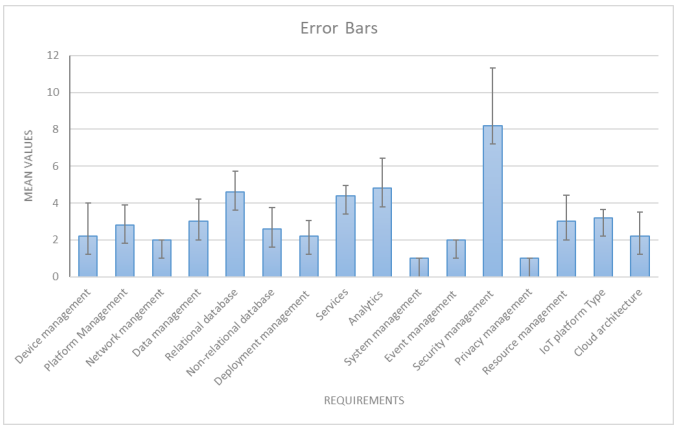


Fig. 5: Error Bar Test for IoT platform requirements

one hand, and those of SaS and ThingWorx on the other, are similar respectively. The mean value of Kaa IoT is the lowest.

By comparing the results of the statistical tests, it can be concluded that complementary results are retrieved: in overall, AWS and Azure are comparatively similar regarding requirement support, SaS and ThingWorx are also similar, while Kaa IoT is the weakest among all.

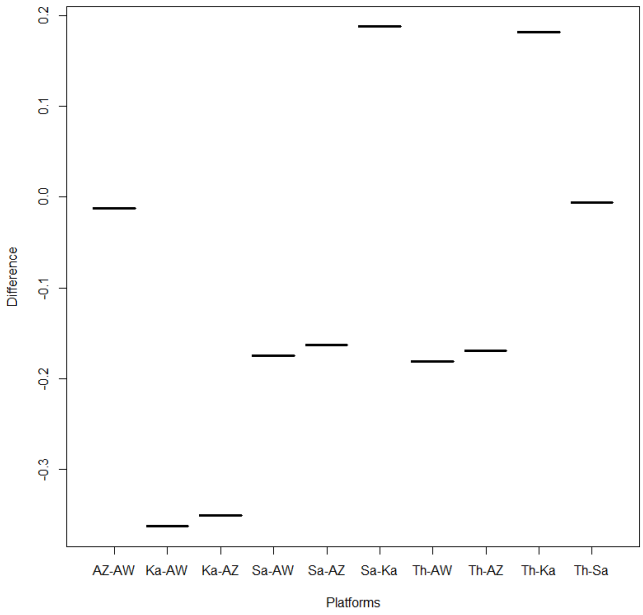


Fig. 6: Pairwise differences between the platform based on Tukey test

TABLE III: Tukey multiple comparisons of means

	diff	lwr	upr	p adj
Azure-AWS	-0.01250	-0.26727124	0.24227124	0.9999185
Kaa-AWS	-0.36250	-0.61727124	-010772876	0.0014602
SAS-AWS	-0.17500	-0.42977124	0.07977124	0.3158835
Thing-AWS	-0.18125	-0.43602124	0.07352124	0.2815629
Kaa-Azure	-0.35000	-0.60477124	-0.09522876	0.0023095
SAS-Azure	-0.16250	-0.41727124	0.09227124	0.3911047
Thing-Azure	-0.16875	-0.42352124	0.08602124	0.3524616
SAS-Kaa	0.18750	-0.06727124	0.44227124	0.2496299
Thing-Kaa	0.18125	-0.07352124	0.43602124	0.2815629
Thing-SAS	-0.00625	-0.26102124	0.24852124	0.9999949

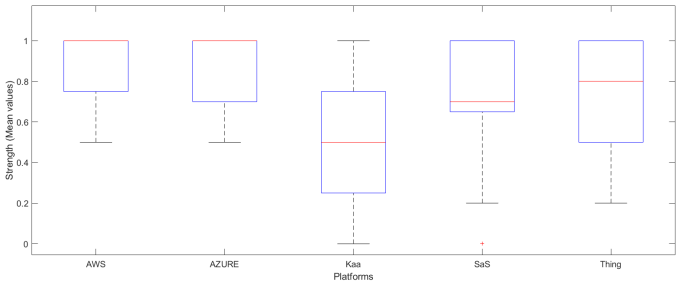


Fig. 7: Mean values comparison

5) Multi-criteria decision method AHP: The pairwise comparison was conducted first on each requirement and then on each subrequirement, following Saaty's relative importance scale [58]. First, a pairwise comparison was made between sub-requirements belonging to a particular requirement, after which a pairwise comparison between the requirements was conducted. Saaty's scale has been applied as follows: 1 = the two (sub-)requirements are equally important; 3 = experience and judgment slightly favor one (sub-)requirement over the other; 5 = experience and judgement strongly favor one (sub-)requirement over the other; 7 = a (sub-)requirement is favored very strongly over another, its dominance demonstrated in practice; 9 = the evidence favoring one (sub-)requirement over the other is of the highest possible order of affirmation. Further refinement (e.g., scoring a 2) was used if needed. For a description of implementing and using AHP, see [58].

The priorities indicating the relative preference of alternatives, relative to each requirement were derived. The judgements are further synthesized to ultimately provide a ranking of the alternatives based on our judgement scores. Figure 8 presents the weight assigned to each requirement and the corresponding score for each IoT platform (the weights on each sub-requirement is left out here, because the aim of this paper is not to extensively showcase the working of AHP). In the figure, the percentage depicted near the requirement name indicates the weight assigned to that requirement and the percentage scored on the (radius) axis indicates how well the IoT platform performs on that particular requirements (from 0% to 100%, where 100% means the relative best on that particular requirement). Using the AHP, relatively ranking of the requirements is as follows: Network management (8.0%), Device management (8.8%), Platform Management (8.8%), Data management (5.0%), Security management (13.7%), Privacy/Compliance management (13.7%), Relational Database (4.5%), Non-relational Database (4.5%) Analytics (6.7%), Event management (3.9%), System management (7.9%), Deployment management (6.1%), Services (3.1%), Resource management (3.3%), IoT platform Type (1.1%) and Cloud architecture (1.1%). It is interesting to note further that the Security Management, Privacy/Compliance Management, Device Management, Platform Management, and Network Management account for already 52.9% of the final weight.

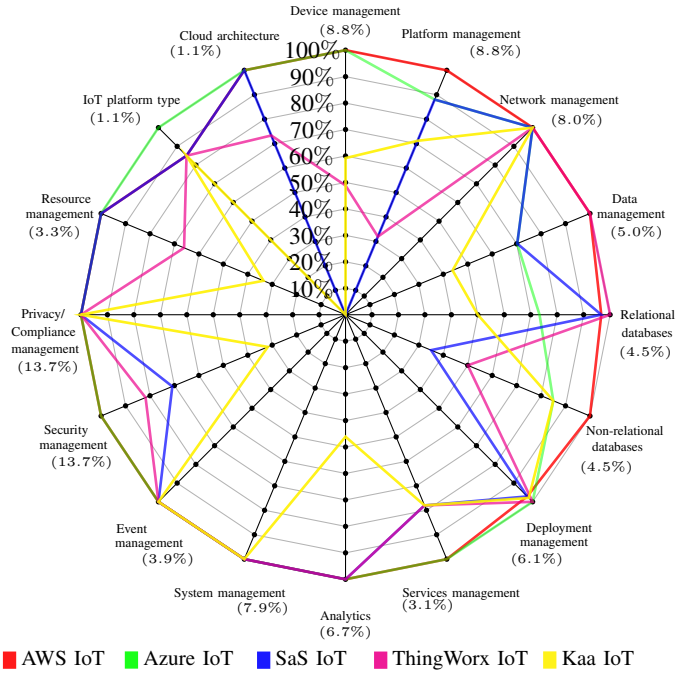


Fig. 8: Relative weight for each requirement in the AHP procedure and scoring of the IoT platforms

Table IV shows the global ranking for each platform. The final score is determined by the relative ranking using the AHP method. For each IoT platform, we determined the score per requirement and summed these scores. The final score is as follows: AWS IoT (91%), Azure IoT (88%), ThingWorx IoT (74%), SaS IoT (71%), Kaa IoT (60%). As our results are indicative, further (sensitivity) analysis needs to be done for a more thorough comparison.

TABLE IV: Final ranking of the AHP method

Platform	AWS IoT	Azure IoT	ThingWorx IoT	SaS IoT	Kaa IoT
Score	0.91	0.88	0.74	0.71	0.60

6) Interactive results based on K-Means Clustering: K-Means clustering was used to determine the clusters of requirements. The functional requirements were labeled in three different classes as discussed in Section VI. The first step was to specify the number of clusters that needed to be extracted (Figure. 9). The optimal number of clusters was determined by performing a plot of the within-groups sum of squares against the number of clusters. Figure 9, indicates the results of the within-groups sum of squares for 1 to 5 groups using the K-means algorithm. The solutions were plotted to see if there

TABLE V: New classification after K-Means applied

Funct. req. no.	AWS	Azure	SaS	ThingWorx	Kaa	Fit. cluster
1	4	4	0	1	2	1
2	4	3	3	1	3	1
3	2	2	2	2	2	1
4	4	3	3	4	1	3
5	5	4	5	6	3	3
6	4	3	1	2	3	1
7	2	3	2	3	1	1
8	5	5	4	4	4	3
9	6	6	5	5	2	3
10	1	1	1	1	1	1
11	2	2	2	2	2	1
12	10	11	8	9	3	2
13	1	1	1	1	1	1
14	4	4	4	2	1	3
15	3	4	3	3	3	3
16	3	3	3	2	0	1

will be any indication of the number of groups. It can be visualized that after three clusters, the observed difference in the within-cluster dissimilarity is not significant.

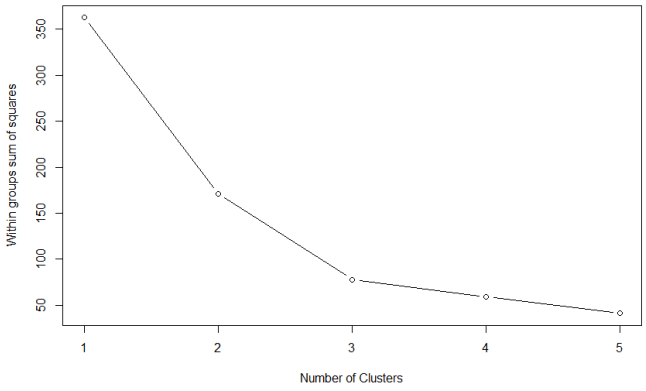


Fig. 9: Number of clusters

In the next step, a cluster solution is applied by using K-Means Cluster Analysis. A new data frame is retrieved, which includes a new column fit-cluster as shown in Table V. The new data classification is represented in Figure 10. The K-Means algorithm classified the requirements based on the sum of sub-requirement values per main requirement in 3 classes: Class 1, Class 3 and Class 2. The illustrated plot of Figure 10 shows the cluster solution. In this figure, the pluses in the red circle refer to Class 1, triangles in the blue circle refer to Class 3, and the pink circle refers to Class 2. However, one should interpret the results obtained via this cluster method with caution, because the results relate to the initial data points and also depend on the number of (sub-)requirements considered.

After completing the clustering of the requirements, a comparison of IoT platforms based on requirements support was conducted. The input data used for this type of comparison are represented in Table VI (Appendix A). Each of the selected IoT platforms supports certain number of sub-requirements

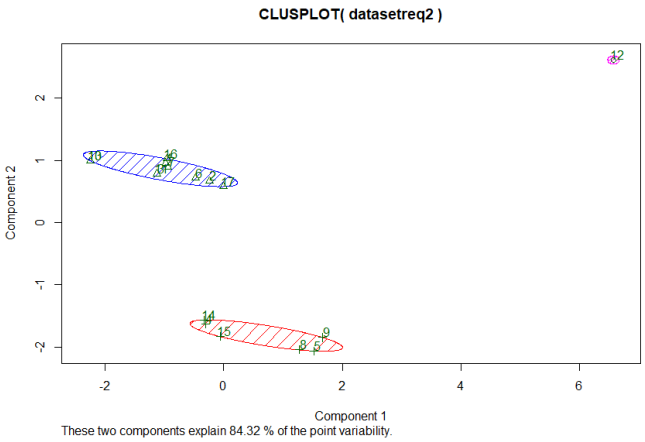


Fig. 10: Clustering plot

shown in Table VI (Appendix A). The number of these sub-requirements supported by each IoT platform, were counted for each main requirement. Based on this idea, a decision can be made about which platform will fulfil the requirements better. The three interactive Sankey diagrams developed for each main class of requirements represent the results of the requirements classification (clustering analysis) in Figure 11, Figure 12 and, Figure 13.

A Sankey diagram is a visualization that is used to depict a specific flow from one set of values to another. For example, in Figure 11 one set of values are the sub-requirements which emerge from the group of main functional requirements. These two nodes (main functional requirements and sub-requirements) are linked to a third node, an IoT platform (AWS IoT, Azure IoT, SAS IoT, ThingWorx, and Kaa IoT). The link between the main functional requirement and the sub-requirements represent that the sub-requirements are a sub-group of the main functional requirements. On the other hand, the link between the sub-requirements and the IoT platform represents whether the IoT platform supports the given sub-requirement or not. Sankey diagrams are the best to be used as it comes to show many-to-many mappings between two domains or multiple paths through stages. For instance, Google Analytics uses sankeys to show how traffic flows from pages to other pages on your web site [65].

The results from the classification of the functional requirements based on K-Means Clustering were slightly different than the initial predictive classification shown in Table I. The functional requirements Database (Non-relational), Event Management, Cloud Architecture, and Privacy Management are categorized in Class 1 and Resource Management in Class 3. After the performed classification, the classes contain the following requirements:

- Class 1: Platform management, System management, Device management, Non-relational database, Network management, Event management, Privacy management, Deployment management, and Cloud architecture (Figure 11);
- Class 2: Security management (Figure 12);
- Class 3: Resource management, Analytics, Data man-

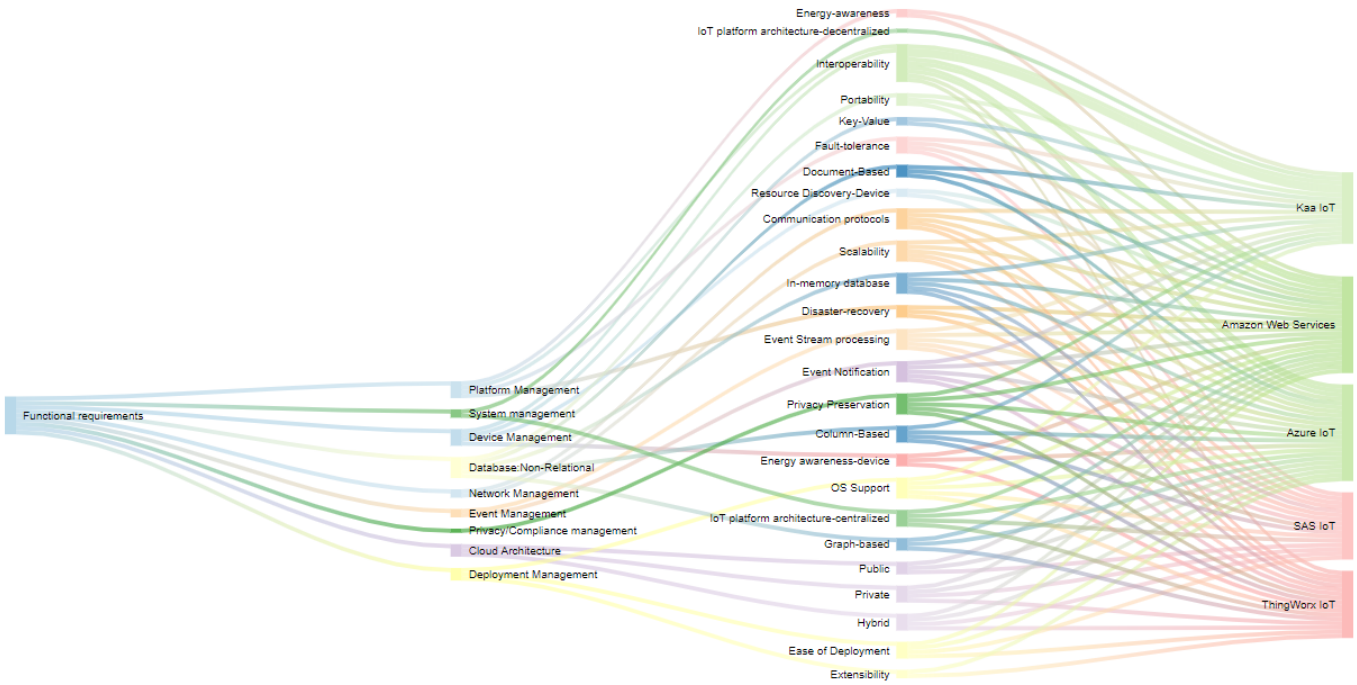


Fig. 11: Functional Requirements - Category 1 after K-Means applied

agement, Relational database, Services, and IoT platform type (Figure 13).

IoT platforms can be distinguished by analyzing the number of supported sub-requirements for each IoT platform. AWS IoT is the strongest platform in Class 1, in terms of satisfying higher number of requirements as shown in Figure 11. It can be observed that Azure IoT is the strongest platform in terms of supporting higher number of Security Management requirements that belong to Class 2 (Figure 12).

The platform which supports the lowest number of requirements in this class is Kaa IoT. Also, the Sankey diagrams indicate that AWS IoT is the strongest platform in terms of supporting higher number of requirements from Class 3 related to Analytics, Data Management, Resource Management, Services, and Database-Relational. Some of the most supported requirements that belong to the Analytics group are: Advanced Analytics and Processing, Mobile Analytics, Prescriptive Analytics, and Predictive Maintenance. Kaa IoT supports the lowest number of sub-requirements in this class.

VIII. Discussion and Future Work

Creating the future of IoT platforms and how they can be extended is not an easy task due to many existing challenges. The developed taxonomy of requirements and platform comparison approach in this research study can, nevertheless, serve as a motivational guide for developing new IoT platforms and for improving the existing platform services. That is why this work provides a good starting point for discussion and further research.

One source of weakness in this study, which could have affected the specification of requirements, is the unavailability of documentation and support for IoT platforms. Although the formulation of functional specifications was done thoroughly by having group discussions, by investigating online materials, and by contacting support communities, it is important to bear in mind the possible biases that could affect this study. For example, the online documentation may have positioned terminology or used certain technologies that are interpreted differently, resulting into either false-positives or true-negatives. For example, the scarcity of documentation tends to give a negative impact on the final scores. The findings should therefore, be doubtlessly more scrutinized. Especially with regard to the construction of more formal procedures for subtracting less-subjective information about the degree to which an IoT platform satisfies a requirement. For instance, a checklist with specific questions addressing the requirements up to the degree of actual implementation can be used.

Some of the issues emerging from the envisioned classification relate to the distinction between the functional requirements and to what level of detail the requirements are specified. The results need to be interpreted with caution because the degree to which a requirement is adhering to decoupling and cohesion currently plays a prominent role in the final outcome. Besides, following the previously defined definition of functional and non-functional requirements, users may perceive some functional requirements as being non-functional and vice versa. Furthermore, the assessment method mainly relies on binary conditions about whether or not a platform satisfies a requirement, but also it would be more interesting to examine how this requirement should be depicted in a bigger picture. For example, one IoT platform may have a unique feature which can be valuable for the user, that it should always outclass other platforms that do not have this particular feature. Moreover, one can also argue that the comparison approach is not completely unbiased, for example, it was relied

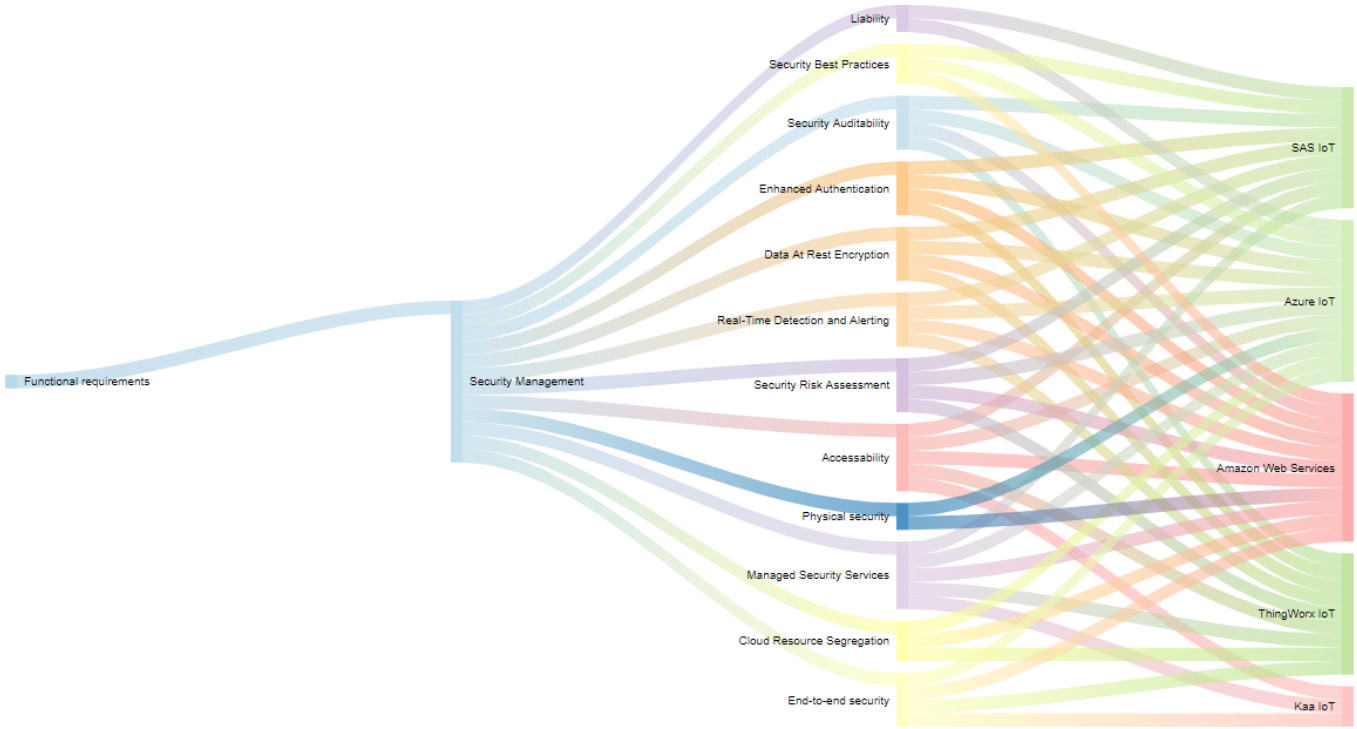


Fig. 12: Functional Requirements - Category 2 after K-Means applied

upon (expert) knowledge from assessors. Further studies could investigate the association between requirement cohesion and decoupling, by using weights instead of binary conditions. For instance, more sophisticated multi-criteria decision making methods or machine learning approaches can be used. Taken together, this discussion suggest that the defined approaches are far from complete and comprehensive, and further investigation and (objective) experimentation is recommended.

Another fruitful area for further work is about investigating a wider range of functional and non-functional requirements. For example, a comparison based on costs, platform usage, political, and demographic policies may be interesting to explore. A taxonomy of defined non-functional requirements was presented, but further work needs to be done to make a sound comparison in that regard. Additionally, researchers and practitioners can use the proposed methods to perform other type of comparative analysis.

For instance, the use of the clustering analysis can help to perform classification of the requirements. Interested readers can be inspired to perform a comparison of the IoT platforms per cluster (specific group of requirements). An example that was given (illustrated in figures 10-12), identifies which IoT platform performs best in Class 1 (Management architecture requirements), Class 2 (Security architecture requirements), or Class 3 (Resource and data management requirements).

In future, more advanced analytics involving a comparison of a larger number and a wider variety of IoT platforms can be implemented. An automated software tool can be used for the data collection from different data sources and a toolbox of statistical methods can be developed. Further work can also be done in developing a system where users can interactively compare platforms based on different features. For example, the user may select two or more platforms and several requirements, and then retrieve comparative results based on the performed selections.

More broadly, research is also needed to further generalize this work. The work on functional requirements can be linked to reference architecture models. One of the architectures may be the International Data Space (IDS) reference model [66]. IDS is about a virtual data space that uses standards and governance models, which are used to facilitate the secure exchange and easy linkage of data in business ecosystems [66]. It would be interesting to identify, analyze, and evaluate their defined requirements (of user companies) with regard to the taxonomy. Mapping the defined requirements to a generic architecture reference model is appealing. Inspired by this reference model, the requirements of each class could be grouped into a specific architecture. Complying with the discovered three classes of functional requirements, there can be three architectures:

- Management architecture, which corresponds to Class 1;
- Security architecture, complying with to Class 2;
- Resource and data management architecture, equivalent to Class 3.

The reference architecture model of IDS consists of four architectures: Business Architecture, Security Architecture, Data and Service Architecture, and Software Architecture [66]. From the sections above, it can be suggested that the management architecture and resource/data management architecture from the taxonomy may be beneficial to be added to the existing reference architecture model of IDS.

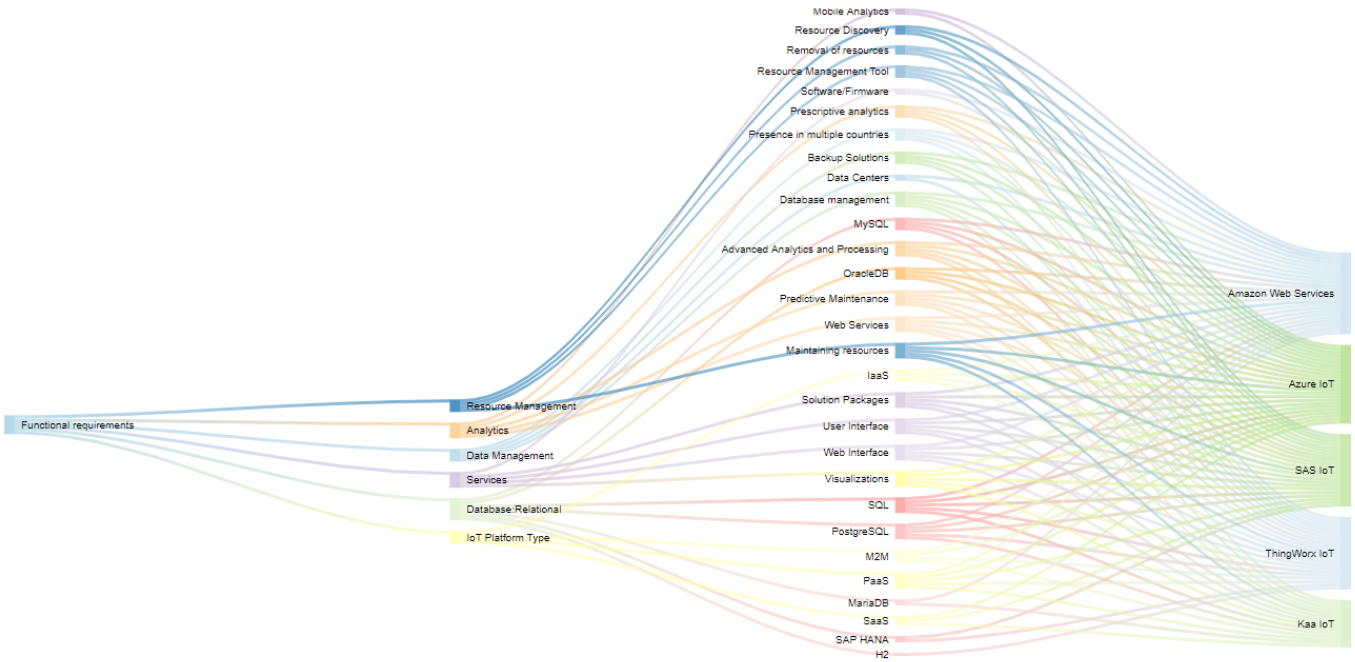


Fig. 13: Functional Requirements - Category 3 after K-means applied

Another direction for future research is that most of the compared IoT platforms are centralized-oriented, but most of their components can also be applied in distributed scenarios and applications. To this end, IDS can represent a distributed solution which can be used for addressing existing problems for IoT applications.

IX. Conclusion

IoT platforms complements the IoT in many different forms and manners. Therefore, with the success of IoT in many application domains, the importance of IoT platforms also soars proportionally. In general, IoT platforms are significantly different from each other in many aspects, for example, platform architecture, the type of services, the number of services, open source, proprietary, etc., and finding the appropriate platform adhering to the functional and non-functional requirements can be challenging. One reason for this is the fact that the platform development documents may not be publicly available or have limited information about how the features are precisely incorporated. Furthermore, interpreting the necessary information and conducting a thorough comparison of which IoT platform to select can also be a challenging and time-consuming procedure. A typical selection process depends on many conditions such as the application domain, user's expertise, and other managerial-based decisions. Consequently, the process of deciding upon which IoT platform to proceed with seems like comparing apples and oranges.

In this research, a considerable number of functional requirements for IoT platforms are described and studied, which are pivotal in many application domains. Platform users can utilize them as a source of reference. A series of methods and statistical tests are discussed for comparing the IoT platforms on the basis of how well they score on a requirement. These tests are extended with visualization methods such that users can also visually compare (on an individual requirement level or on a platform level).

According to the reviewed sources, this is the first attempt to extensively, statistically and visually, compare IoT platforms based on the functional requirements for IoT platforms. To summarize, using the proposed framework, it will be possible to identify to which requirements the IoT platforms adhere and conduct a qualitative-based assessment of IoT platforms. Despite this study's exploratory nature and limitations, it was attempted to provide some insight into comparing IoT platforms. Trying to highlight the similarities and differences between them, however, still remains ambiguous and challenging. The selection process about which platform to use is still difficult and resembles comparing apples and oranges.

APPENDIX A
REQUIREMENT SUPPORT FOR EACH IOT PLATFORM

TABLE VI: Requirement support for each IoT platform

Functional Requirements	Sub-requirements	AWS IoT	Azure IoT	SaS IoT	ThingWorx IoT	Kaa IoT
Device Management	Resource Discovery	1	1	NA	NA	NA
	Interoperability	1	1	NA	NA	1
	Portability	1	1	NA	NA	1
	Energy Awareness	1	1	NA	1	NA
Platform Management	Interoperability	1	1	1	1	1
	Energy Awareness	1	NA	NA	NA	1
	Fault-tolerance	1	1	1	NA	1
	Disaster Recovery	1	1	1	NA	NA
Network Management	Communication Protocols	1	1	1	1	1
	Scalability	1	1	1	1	1
Data Management	Data Centers	1	NA	NA	1	NA
	Presence in countries	1	1	1	1	NA
	Backup Solutions	1	1	1	1	NA
	Database Management	1	1	1	1	1
Relational Database	SQL	1	1	1	1	1
	MySQL	1	1	1	1	NA
	PostgreSQL	1	1	1	1	1
	SAP HANA	NA	NA	1	1	NA
	H2	NA	NA	NA	1	NA
	MariaDB	1	NA	NA	NA	1
	Oracle DB	1	1	1	1	NA
Non-relational Database	Key-Value	1	NA	NA	NA	1
	Document-based	1	1	NA	NA	1
	Column-based	NA	1	1	1	1
	In-memory	1	NA	NA	NA	NA
	Graph-based	1	1	NA	1	NA
Deployment Management	OS Support	1	1	1	1	1
	Ease of deployment	1	1	1	1	NA
	Extensibility	NA	1	NA	1	NA
Services	Solution Packages	1	1	1	1	1
	User Interface	1	1	1	1	1
	Web Interface	1	1	1	1	1
	Software/Firmware	1	1	NA	NA	NA
	Visualizations	1	1	1	1	1
Analytics	Advanced Analytics and Processing	1	1	1	1	NA
	Algorithms Support	1	1	1	1	NA
	Prescriptive Analytics	1	1	1	1	NA
	Predictive Maintenance	1	1	1	1	1
	Web Services/Web API(REST, SOAP, RPC)	1	1	1	1	1
	Mobile Analytics Solutions	1	1	NA	NA	NA
System Management	IoT platform architecture-centralized	1	1	1	1	NA
	IoT platform architecture-decentralized	NA	NA	NA	NA	1
Event Management	Event Stream Processing	1	1	1	1	1
	Event Notification	1	1	1	1	1
Security Management	Security Auditability	1	1	1	1	NA
	Accessability	1	1	1	1	1
	Enhanced Identification and Authentication	1	1	1	1	NA
	Data At Rest Encryption	1	1	1	1	NA
	Real-Time Detection and Alerting	1	1	1	1	NA
	Security Risk Assessment	1	1	1	1	NA
	Liability	NA	1	1	NA	NA
	Managed Security Services	1	1	1	1	1
	Cloud Resource Segregation	1	1	NA	1	NA
	End-to-End Security	1	1	NA	1	1
	Physical Security	1	1	NA	NA	NA
Privacy/Compliance Management	Privacy Preservation	1	1	1	1	1
Resource Management	Resource Discovery	1	1	1	NA	NA
	Maintaining Resources	1	1	1	1	1
	Removal of Resources	1	1	1	NA	NA
	Resource Management Tool	1	1	1	1	NA
IoT platform Type	PaaS	1	1	1	1	1
	SaaS	NA	1	1	NA	1
	IaaS	1	1	1	1	NA
	M2M	1	1	NA	1	1
Cloud Architecture	Public	1	1	1	NA	NA
	Private	1	1	1	1	NA
	Hybrid	1	1	1	1	NA

APPENDIX B
TOTAL NUMBER OF REQUIREMENT SUPPORTED BY EACH IOT PLATFORM

TABLE VII: Total number of requirement supported by each IoT platform

Functional Requirements	Sub-requirements	AWS IoT	Azure IoT	SaS IoT	ThingWorx IoT	Kaa IoT
Device Management	Resource Discovery	1	1	NA	NA	NA
	Interoperability	1	1	NA	NA	1
	Portability	1	1	NA	NA	1
	Energy Awareness	1	1	NA	1	NA
Mean value per main requirement		**1**	**1**	**0**	**0.2**	**0.5**
Platform Management	Interoperability	1	1	1	1	1
	Energy Awareness	1	NA	NA	NA	1
	Fault-tolerance	1	1	1	NA	1
	Disaster Recovery	1	1	1	NA	NA
Mean value per main requirement		**1**	**0.7**	**0.7**	**0.2**	**0.7**
Network Management	Communication Protocols	1	1	1	1	1
	Scalability	1	1	1	1	1
Mean value per main requirement		**1**	**1**	**1**	**1**	**1**
Data Management	Data Centers	1	NA	NA	1	NA
	Presence in countries	1	1	1	1	NA
	Backup Solutions	1	1	1	1	NA
	Database Management	1	1	1	1	1
Mean value per main requirement		**1**	**0.7**	**0.7**	**1**	**0.2**
Relational Database	SQL	1	1	1	1	1
	MySQL	1	1	1	1	NA
	PostgreSQL	1	1	1	1	1
	SAP HANA	NA	NA	1	1	NA
	H2	NA	NA	NA	1	NA
	MariaDB	1	NA	NA	NA	1
	Oracle DB	1	1	1	1	NA
Mean value per main requirement		**0.7**	**0.5**	**0.7**	**0.8**	**0.4**
Non-relational Database	Key-Value	1	NA	NA	NA	1
	Document-based	1	1	NA	NA	1
	Column-based	NA	1	1	1	1
	In-memory	1	NA	NA	NA	NA
	Graph-based	1	1	NA	1	NA
Mean value per main requirement		**0.8**	**0.6**	**0.2**	**0.4**	**0.6**
Deployment Management	OS Support	1	1	1	1	1
	Ease Of Deployment	1	1	1	1	NA
	Extensibility	NA	1	NA	1	NA
Mean value per main requirement		**0.6**	**1**	**0.6**	**1**	**0.3**
Services	Solution Packages	1	1	1	1	1
	User Interface	1	1	1	1	1
	Web Interface	1	1	1	1	1
	Software/Firmware	1	1	NA	NA	NA
	Visualizations	1	1	1	1	1
Mean value per main requirement		**1**	**1**	**0.8**	**0.8**	**0.8**
Analytics	Advanced Analytics and Processing	1	1	1	1	NA
	Algorithms Support	1	1	1	1	NA
	Prescriptive Analytics	1	1	1	1	NA
	Predictive Maintenance	1	1	1	1	1
	Web services/Web API(REST, SOAP, RPC)	1	1	1	1	1
	Mobile Analytics Solutions	1	1	NA	NA	NA
Mean value per main requirement		**1**	**1**	**0.8**	**0.8**	**0.3**
System Management	IoT platform architecture centralized	1	1	1	1	NA
	IoT platform architecture decentralized	NA	NA	NA	NA	1
Mean value per main requirement		**0.5**	**0.5**	**0.5**	**0.5**	**0.5**
Overall mean value for platform		**0.8**	**0.8**	**0.7**	**0.7**	**0.5**
Event Management	Event Stream Processing	1	1	1	1	1
	Event Notification	1	1	1	1	1
Mean value per main requirement		**1**	**1**	**1**	**1**	**1**

Continuation of Table IV

Functional Requirements	Sub-requirements	AWS IoT	Azure IoT	SaS IoT	ThingWorx IoT	Kaa IoT
Security Management	Security Auditability	1	1	1	1	NA
	Accessability	1	1	1	1	1
	Enhanced Identification and Authentication	1	1	1	1	NA
	Data At Rest Encryption	1	1	1	1	NA
	Real-Time Detection and Alerting	1	1	1	1	NA
	Security Risk Assessment	1	1	1	1	NA
	Liability	NA	1	1	NA	NA
	Managed Security Services	1	1	1	1	1
	Cloud Resource Segregation	1	1	NA	1	NA
	End-to-End Security	1	1	NA	1	1
	Physical Security	1	1	NA	NA	NA
Mean value per main requirement		**0.9**	**1**	**0.7**	**0.8**	**0.2**
Privacy/Compliance Management	Privacy Preservation	1	1	1	1	1
Mean value per main requirement		**1**	**1**	**1**	**1**	**1**
Resource Management	Resource Discovery	1	1	1	NA	NA
	Maintaining Resources	1	1	1	1	1
	Removal of Resources	1	1	1	NA	NA
	Resource Management Tool	1	1	1	1	NA
Mean value per main requirement		**1**	**1**	**1**	**0.5**	**0.2**
IoT platform Type	PaaS	1	1	1	1	1
	SaaS	NA	1	1	NA	1
	IaaS	1	1	1	1	NA
	M2M	1	1	NA	1	1
Mean value per main requirement		**0.7**	**1**	**0.7**	**0.7**	**0.7**
Cloud Architecture	Public	1	1	1	NA	NA
	Private	1	1	1	1	NA
	Hybrid	1	1	1	1	NA
Mean value per main requirement		**1**	**1**	**1**	**0.6**	**0**

ACKNOWLEDGMENT

This work is supported by OP Oost, project CountDown, and the Netherlands Organization for Scientific Research (NWO), project DataRel (grant 628.009.015). The authors would also like to thank all project partners.

REFERENCES

[1] M. Weyrich and C. Ebert, "Reference architectures for the internet of things," *IEEE Software*, vol. 33, no. 1, pp. 112–116, 2016.

[2] "Internet of things (IoT) solutions services," accessed: 12-Nov.-2019. [Online]. Available: https://www.cisco.com/c/en/us/solutions/internet-of-things/overview.html

[3] "What is the internet of things platform - All about IoT technology and applications," accessed: 12-Aug.-2020. [Online]. Available: https://www.kaaproject.org/

[4] P. P. Ray, "A survey of IoT cloud platforms," *Future Computing and Informatics Journal*, vol. 1, no. 1-2, pp. 35–46, 2016.

[5] "Open roads to a better connected world," accessed: 26-Jan-2019. [Online]. Available: https://www.huawei.com/minisite/mwc2015/en/articles/connections-and-data-management-in-the-iot-era.html

[6] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007.

[7] D. T. Ross and K. E. Schoman, "Structured analysis for requirements definition," *IEEE Transactions on Software Engineering*, no. 1, pp. 6–15, 1977.

[8] W. Liming, N. Che Pa, R. Abdullah, W. N. Wanabrahman, and M. Tee, "Exploring functional and non-functional requirements of social media on knowledge sharing," *Journal of Theoretical and Applied Information Technology*, vol. 93, no. 2, pp. 595–605, 2016.

[9] A. I. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[10] P. Ray, "A survey on internet of things architectures," *Journal of King Saud University - Computer and Information Sciences*, vol. 30, no. 3, pp. 291 – 319, 2018.

[11] "Lightweight machine to machine technical specification," accessed: 12-Aug-2020. [Online]. Available: http://www.openmobilealliance.org/release/LightweightM2M/Lightweight_Machine_to_Machine-v1_1-OMASpecworks.pdf

[12] "Network configuration protocol light (NETCONF Light)," accessed: 12-Aug-2020. [Online]. Available: https://tools.ietf.org/html/draft-schoenw-netconf-light-01

[13] S. Sen and A. Balasubramanian, "A highly resilient and scalable broker architecture for IoT applications," in *2018 10th International Conference on Communication Systems & Networks (COMSNETS)*. IEEE, 2018, pp. 336–341.

[14] J. Mocnej, W. K. Seah, A. Pekar, and I. Zolotova, "Decentralised IoT architecture for efficient resources utilisation," *IFAC-PapersOnLine*, vol. 51, no. 6, pp. 168–173, 2018.

[15] M. Vögler, J. M. Schleicher, C. Inzinger, and S. Dustdar, "DIANE-dynamic IoT application deployment," in *2015 IEEE International Conference on Mobile Services*. IEEE, 2015, pp. 298–305.

[16] F. C. Delicato, P. F. Pires, and T. Batista, "The resource management challenge in IoT," in *Resource management for Internet of Things*. Springer, 2017, pp. 7–18.

[17] S. Tayeb, S. Latifi, and Y. Kim, "A survey on IoT communication and computation frameworks: An industrial perspective," in *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 2017, pp. 1–6.

[18] T. Azzabi, H. Farhat, and N. Sahli, "A survey on wireless sensor networks security issues and military specificities," in *2017 International Conference on Advanced Systems and Electric Technologies (IC_ASET)*. IEEE, 2017, pp. 66–72.

[19] R. Prabha, M. Krishnaveni, S. Manjula, K. Venugopal, and L. Patnaik, "QoS aware trust metric based framework for wireless sensor networks," *Procedia Computer Science*, vol. 48, pp. 373–380, 2015.

[20] J. Guo and R. Chen, "A classification of trust computation models for service-oriented internet of things systems," in *2015 IEEE International Conference on Services Computing*. IEEE, 2015, pp. 324–331.

[21] "Guide to cryptography-owasp," accessed: 12-Aug-2020. [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

[22] A. Ometov, S. Bezzateev, N. Mäkitalo, S. Andreev, T. Mikkonen, and Y. Koucheryavy, "Multi-factor authentication: A survey," *Cryptography*, vol. 2, no. 1, p. 1, 2018.

[23] K. Scarfone, M. Souppaya, M. Sexton *et al.*, "Guide to storage encryption technologies for end user devices," *NIST Special Publication*, vol. 800, p. 111, 2007.

[24] P. Paul, N. Dutta, B. A. Biswas, M. Das, S. Biswas, Z. Khalid, and H. N. Saha, "An internet of things (IoT) based system to analyze real-time collapsing probability of structures," in *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE, 2018, pp. 1070–1075.

[25] "Risk management framework for information systems and organizations: A system life cycle approach for security and privacy," accessed: 09-Dec-2019. [Online]. Available: https://csrc.nist.gov/CSRC/media/Publications/sp/800-37/rev-2/draft/documents/sp800-37r2-discussion-draft.pdf

[26] M. M. Losavio, K. P. Chow, A. Koltay, and J. James, "The internet of things and the smart city: Legal challenges with digital forensics, privacy, and security," *Security and Privacy*, vol. 1, no. 3, p. e23, 2018.

[27] J. Allen, D. Gabbard, C. May, E. Hayes, and C. Sledge, "Outsourcing managed security services," Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, Tech. Rep., 2003.

[28] Z. Ning, X. Kong, F. Xia, W. Hou, and X. Wang, "Green and sustainable cloud of things: Enabling collaborative edge computing," *IEEE Communications Magazine*.

[29] "The EU general data protection regulation (GDPR) is the most important change in data privacy regulation in 20 years," accessed: 12-Aug-2020. [Online]. Available: https://gdpr.eu/

[30] "Payment card infrastructure data security standard," accessed: 12-Aug-2020. [Online]. Available: https://www.pcisecuritystandards.org/pci_security/

[31] "Health insurance portability and accountability act," accessed: 09-Dec-2019. [Online]. Available: https://www.enisa.europa.eu/topics/threat-risk-management/risk-management/current-risk/laws-regulation/data-protection-privacy/health-insurance-portability-and-accountability-act

[32] "ISO/IEC 27018:2014," accessed: 15-Jan-2019. [Online]. Available: https://www.iso.org/standard/61498.html

[33] L. Fan, J. R. Gil-Garcia, D. Werthmuller, G. B. Burke, and X. Hong, "Investigating blockchain as a data management tool for iot devices in smart city initiatives," in *Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age*. ACM, 2018.

[34] B. Cheng, G. Solmaz, F. Cirillo, E. Kovacs, K. Terasawa, and A. Kitazawa, "Fogflow: Easy programming of iot services over cloud and edges for smart cities," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 696–707, 2018.

[35] M. Marjani, F. Nasaruddin, A. Gani, A. Karim, I. Hashem, A. Siddiqa, and I. Yaqoob, "Big IoT Data Analytics: Architecture, Opportunities, and Open Research Challenges," *IEEE Access*, vol. 5, pp. 5247–5261, 2017.

[36] S. Yang, "IoT Stream Processing and Analytics in the Fog," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 21–27, 2017.

[37] S. Earley, "Analytics, machine learning, and the internet of things," *IT Professional*, vol. 17, no. 1, pp. 10–13, 2015.

[38] J. Vater, L. Harscheidt, and A. Knoll, "Smart manufacturing with prescriptive analytics," 2019, pp. 224–228.

[39] M. G. Pecht and M. Kang, *Predictive Maintenance in the IoT Era*. IEEE, 2019, pp. 589–612.

[40] C.-Y. Huang and C.-H. Wu, "A web service protocol realizing interoperable internet of things tasking capability," *Sensors (Switzerland)*, vol. 16, no. 9, 2016.

[41] A. Zaslavsky and D. Georgakopoulos, "Internet of things: Challenges and state-of-the-art solutions in internet-scale sensor information management and mobile analytics," in *2015 16th IEEE International Conference on Mobile Data Management*, vol. 2. IEEE, 2015, pp. 3–6.

[42] P. Agarwal and M. Alam, "Investigating iot middleware platforms for smart application development," *arXiv preprint arXiv:1810.12292*, 2018.

[43] Z. Qureshi, N. Agrawal, and D. Chouhan, "Cloud based IOT: Architecture, Application, Challenges and Future," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol. 3, pp. 359–368, 2018.

[44] T. Sultana and K. A. Wahid, "IoT-Guard: Event-Driven Fog-Based Video Surveillance System for Real-Time Security Management," *IEEE Access*, vol. 7, pp. 134 881–134 894, 2019.
[45] S. Sarawi, M. Anbar, K. Alieyan, and M. Alzubaidi, "Internet of Things (IoT) Communication Protocols: Review," in *2017 8th International Conference on Information Technology (ICIT)*, 2017, pp. 685–690.
[46] Y. Idris and N. A. Muhammad, "A comparative study of wireless communication protocols: Zigbee vs bluetooth," 2016.
[47] J. Mineraud, O. Mazhelis, X. Su, and S. Tarkoma, "A gap analysis of internet-of-things platforms," *Computer Communications*, vol. 89, pp. 5–16, 2016.
[48] S. N. Mahalank, K. B. Malagund, and R. Banakar, "Non functional requirement analysis in IoT based smart traffic management system," in *2016 International Conference on Computing Communication Control and automation (ICCUBEA)*. IEEE, 2016, pp. 1–6.
[49] A. Danylenko and W. Löwe, "Context-aware recommender systems for non-functional requirements," in *2012 Third International Workshop on Recommendation Systems for Software Engineering (RSSE)*. IEEE, 2012, pp. 80–84.
[50] T. Gregorio, A. Oliveira, D. A. de Melo, and G. M. da Silva, "Comparing IoT Platforms under Middleware Requirements in an IoT Perspective," 2016.
[51] M. Kim, N. Lee, and J. Park, "A Security Generic Service Interface of Internet of Things (IoT) Platforms," *Symmetry*, vol. 9, no. 9, p. 171, 2017.
[52] E. Patti and A. Acquaviva, "IoT platform for Smart Cities: Requirements and implementation case studies," in *2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*. IEEE, 2016, pp. 1–6.
[53] Y. Kondratenko, G. Kondratenko, and I. Sidenko, "Multi-criteria decision making for selecting a rational iot platform," in *2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*.
[54] R. L. Dumitru, "Iot platforms: Analysis for building projects," *Informatica Economica*, vol. 21, 2018.
[55] "R packages," accessed: 26-Jan-2019]. [Online]. Available: https://rviews.rstudio.com/categories/r-packages/
[56] E. Triantaphyllou, "Multi-criteria decision making methods," in *Multi-criteria decision making methods: A comparative study*. Springer, 2000, pp. 5–21.
[57] V. Belton and T. Stewart, *Multiple criteria decision analysis: An integrated approach*. Springer Science & Business Media, 2002.
[58] T. L. Saaty, "Decision making with the analytic hierarchy process," *International journal of services sciences*, vol. 1, no. 1, pp. 83–98, 2008.
[59] D. Marutho, S. Hendra Handaka, E. Wijaya, and Muljono, "The determination of cluster number at k-mean using elbow method and purity evaluation on headline news," in *2018 International Seminar on Application for Technology of Information and Communication*. IEEE, 2018, pp. 533–538.
[60] "IoT Applications and Solutions - What is the Internet of Things (IoT)? - AWS," accessed: 26-Jan-2019]. [Online]. Available: https://aws.amazon.com/iot/
[61] "Introduction to the Azure Internet of Things (IoT)," accessed: 26-Jan-2019]. [Online]. Available: https://docs.microsoft.com/en-us/azure/iot-suite/iot-suite-what-is-azure-iot
[62] "IoT Analytics Solutions," accessed: 12-Aug-2020]. [Online]. Available: https://www.sas.com/nl_nl/solutions/iot.html
[63] "Accelerate Your Digital Transformation With the Industrial IoT," accessed: 12-Aug-2020]. [Online]. Available: https://www.ptc.com/en/products/iot
[64] "Kaa features overview: multi-purpose IoT technology," accessed: 12-Aug-2020]. [Online]. Available: https://www.kaaproject.org/overview/
[65] Google. Google Charts. [Online]. Available: https://developers.google.com/chart/interactive/docs/gallery/sankey
[66] B. Otto, S. Auer, J. Cirullies, J. Jürjens, N. Menz, J. Schon, and S. Wenzel, "Industrial data space: digital sovereignity over data," *Fraunhofer White Paper*, 2016.

Adriana Mijuskovic is a PhD candidate in the Pervasive Systems group at the University of Twente, The Netherlands. She received her MSc degree in Software Engineering and Telecommunication at South East European University, North Macedonia. Her research interests are machine learning, data analysis, data visualization, cloud computing, IoT, risk and safety management.

Ikram Ullah is a PhD candidate within Pervasive Systems group at the University of Twente, The Netherlands. He did his Bachelor in Electronics and Computer Engineering from Politecnico Di Torino in Italy. He has earned a dual Master degree in Cyber Security from the Technical University of Berlin and University of Twente. His research interests are in the areas of Blockchain technologies, IOTA, IoT security, and cloud security.

Rob Bemthuis is a PhD student within the Pervasive Systems group at the University of Twente, The Netherlands. He received a MSc degree in Industrial Engineering and Management in 2017, at the University of Twente. His current research interests include multi-agent systems, supply chain logistics, and Internet of Things. He was the receipt of the Best Doctoral Consortium Paper Award for the 23rd IEEE International EDOC 2019 Conference.

Nirvana Meratnia is associate professor in the Pervasive Systems Group at the University of Twente in the area of distributed sensor data analytics, machine learning, and reasoning in wireless sensor networks. Her research interests are in the area of smart sensor systems, Internet of Things (IoT), cyber physical systems, (underwater) wireless sensor networks, and wearable computing.

Paul Havinga received his Ph.D. in Mobile Multimedia Systems in 2000, and was awarded with the DOW Dissertation Energy Award for this work. Currently, he is professor of the Pervasive Systems research group at the University of Twente. His research interests are in the area of energy-efficient architectures and protocols, sensor networks, wireless communication, ubiquitous computing, personal communication systems, and (reconfigurable) hardware architectures.