IoT

MDPI

Article

Process Automation in an IoT–Fog–Cloud Ecosystem: A Survey and Taxonomy

Hossein Chegini [1], Ranesh Kumar Naha [2,*], Aniket Mahanti [1,3] and Parimala Thulasiraman [4]

[1] School of Computer Science, University of Auckland, Auckland 1010, New Zealand; h.chegini@auckland.ac.nz (H.C.); a.mahanti@auckland.ac.nz (A.M.)
[2] School of Information and Communication Technology, University of Tasmania, Hobart, TAS 7005, Australia
[3] Department of Computer Science, University of New Brunswick, Saint John, NB E2L 4L5, Canada
[4] Department of Computer Science, University of Manitoba, Winnipeg, MB R3T 2N2, Canada; thulasir@cs.umanitoba.ca
* Correspondence: raneshkumar.naha@utas.edu.au

Abstract: The number of IoT sensors and physical objects accommodated on the Internet is increasing day by day, and traditional Cloud Computing would not be able to host IoT data because of its high latency. Being challenged of processing all IoT big data on Cloud facilities, there is not enough study on automating components to deal with the big data and real-time tasks in the IoT–Fog–Cloud ecosystem. For instance, designing automatic data transfer from the fog layer to cloud layer, which contains enormous distributed devices is challenging. Considering fog as the supporting processing layer, dealing with decentralized devices in the IoT and fog layer leads us to think of other automatic mechanisms to manage the existing heterogeneity. The big data and heterogeneity challenges also motivated us to design other automatic components for Fog resiliency, which we address as the third challenge in the ecosystem. Fog resiliency makes the processing of IoT tasks independent to the Cloud layer. This survey aims to review, study, and analyze the automatic functions as a taxonomy to help researchers, who are implementing methods and algorithms for different IoT applications. We demonstrated the automatic functions through our research in accordance to each challenge. The study also discusses and suggests automating the tasks, methods, and processes of the ecosystem that still process the data manually.

Keywords: internet of things (IoT); fog computing; automation; cloud computing

Citation: Chegini, H.; Naha, R.K.; Mahanti, A.; Thulasiraman, P. Process Automation in an IoT–Fog–Cloud Ecosystem: A Survey and Taxonomy. *IoT* **2021**, *2*, 92–118. https://doi.org/10.3390/iot2010006

Academic Editors: Jinan Fiaidhi, Sabah Mohammed, Simon Fong, Naseer Al-Jawad and Dalin Zhang
Received: 12 January 2021
Accepted: 2 February 2021
Published: 7 February 2021

1. Introduction

Connecting various objects to the Internet has created a significant shift in the information-based economy. The flow of information will play a crucial role in future Information Technology (IT) ecosystems [1]. The connectivity of different types of objects, sensors, and end devices to the Internet is known as the Internet of Things (IoT). The expected large number of sensors connecting to the Internet will cause severe issues for IoT task management, data process, and decision-making systems. Because providing the same services to the users will face latency in data management and data processing of IoT sensors. Therefore, ensuring a good Quality-of-Service (QoS) with a low-latency resolution for IoT applications, IT services, user tasks, and client events in the ecosystem is crucial to support latency-aware real-time applications.

Corotinschi and Găitan [2] show that the evolution of industrial production has occurred in four stages culminating in the IoT: (1) mechanization and the invention of the steam engine and water power, (2) mass production and assembly lines, (3) the computer and automation processing, (4) the Internet of Things (IoT), (5) the collaboration between man and machine in industry and robotic tasks conversion. These stages are referred to as Industry 1.0, Industry 2.0, Industry 3.0, Industry 4.0, and Industry 5.0.

Geissbauer [3] describes that the advantage of the industrial Internet is to advance the management of value chains. Digitalization and interconnection are essential benefits provided by Industry 4.0. In addition to management, advanced computation, and intelligent decision making, IoT will provide us with control over the entirety of any process, such as the life cycle of products. Frankó et al. [4] depicts an example of industry 4.0 in supply chain management and asset monitoring. Massaro and Galiano [5] demonstrates another example of industry 4.0 in the optimization of pasta line production. The IoT sensors monitor and track the workflow tasks, and industry 5.0 is also added for double-checking of the production with image processing.

Cloud computing is a global infrastructure environment that provides sophisticated resources for tasks and IT services. Google Drive, Dropbox, and Microsoft Azure are examples of Cloud computing. Weerasiri et al. [6] demonstrates a comprehensive framework of Cloud Computing according to its tools, models, structure, and methods. Maiti et al. [7] states in 2018 that 50 billion IoT devices will be connected to the Internet by 2020. According totheir statement and now that we are in 2021, we should have faced more than 50 billion IoT devices, which continuously generate data. High latency is the reason that Cloud Computing cannot be employed for processing the generated data. Therefore, Cloud Computing is not an efficient solution for hosting IoT applications and data processing.

Groover [8] defines automation and automatic processes in industry. By the definition, automation is any task, method, technique, algorithm, process, and control, which can minimize human intervention and assistance. The significant advantage of automatic processes is to minimize latency in any points, where human decision-making is required and tasks are interrupted when human actions are still not made.

When more devices and IoT sensors are joining the ecosystem, defining and designing automatic components for data process and task management of such massive IoT data is essential and pertinent. Since Cloud Computing suffers from high latency, and many IoT applications need on-demand processing and real-time decision making, relying on the Cloud is not a suitable solution. Additionally, the real-time and time-sensitive characteristics of Industry 4.0 IoT applications make the Cloud an unsuitable option for their hosting. Hence, data processing in the Fog environment is a preferable solution.

Fog computing, a term first used by Cisco, evolves the Cloud to an extent close to end-devices and users to host the time-sensitive applications [9]. Fog computing uses distributed hardware platforms resided in IoT and Fog layers for data processing.

Figure 1 is a schematic diagram of the IoT–Fog–Cloud ecosystem. The ecosystem contains three layers: IoT, Fog, and Cloud, which collaborate for data processing. In the first layer (IoT), different types of objects are connected to the Internet, such as vehicles, humans, houses, and electronic devices. The objects and sensors are connected to the Fog layer, where different types of Fog nodes are available for processing data. The Fog layer is also known as the intelligent layer. Raspberry Pis, Gateways, routers, and smart hubs for homes are examples of Fog nodes. The devices need to connect and communicate for any decision making. Each layer has its characteristics. The first layer (IoT) has cheap resources with local information obtained from IoT sensors with a decentralized architecture. The second layer (the Fog) contains Fog nodes and is responsible for intelligent and real-time actions. The third layer (the Cloud) has global information with sophisticated resources and centralized architecture.

Fog computing is a reliable and resilient hosting environment that emerged with the motivation of resolving the problems of Cloud Computing [10]. Puliafito et al. [11] depicts the fog capabilities in hosting over increasing IoT data and its features in Virtual Machines (VM) and container migration, making it a flexible and dynamic hosting environment. Deshmukh and More [9] also noted the preferences of fog layer and fog computing to Cloud Computing such as low latency, distributing computation, wireless capability, and more security, which makes it an intelligent layer in the ecosystem. As an intelligent layer, the Fog decides when to host a task and transfer it to the Cloud. Therefore, the Fog layer

should be resilient and flexible to host various real-time IoT-generated events and tasks. Therefore, Fog resiliency is a challenge in the ecosystem.

As the number of IoT end devices are increasing, a large number of objects will connect to the Internet and will generate a enormous data stream, called big data. Processing such huge amounts of data requires automatic operating tasks. Therefore, handling and managing big data is another essential challenge among the current challenges.

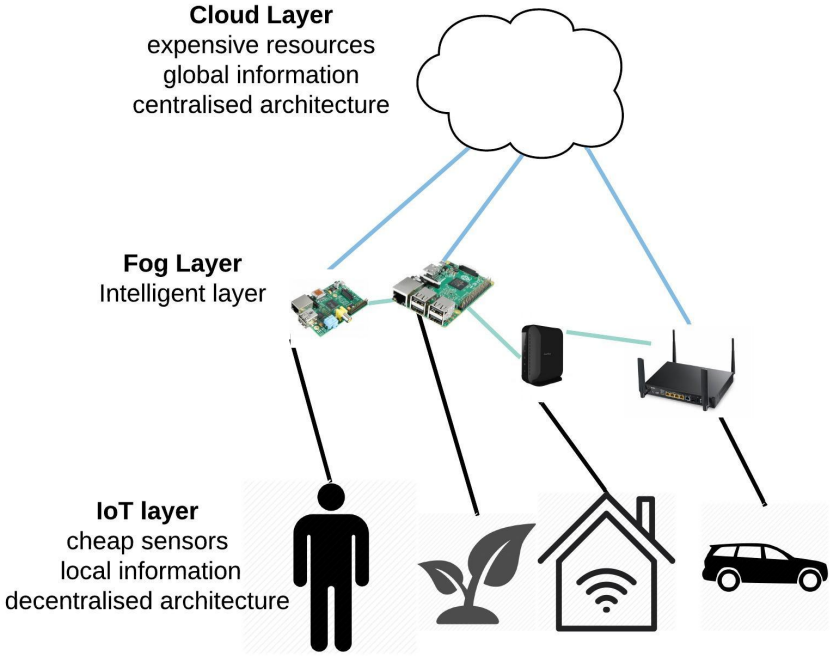


Figure 1. Schematic diagram of the interaction and interplay between the IoT, Fog, and Cloud layer.

On the other hand, there are various types of IoT components and applications in terms of data, software, hardware, and actors. Different types of information are introduced to the IoT application: local information (close to the IoT layer) and global information (close to the Cloud layer).

In terms of hardware, various hardware platforms collaborate for data processing in the ecosystem, such as computational resources, storing resources, disk resources, and network resources with multiple attributes. As for software, different programming codes exist. Communicating agents, small code scripts, data types, modular codes, compilers, Application Programming Interface (API), Operating Systems (OS), applications, and Machine Learning (ML) applications are all examples of various types of software codes.

To summarize, the three main challenges in any IoT–Fog–Cloud are: (1) increasing Fog resiliency, (2) real-time processing of Big data and (3) managing the heterogeneity. Tasks, algorithms, and functions need to be defined in a way to tackle these challenges.

We also analyzed the number of publications in different digital libraries such as "Web of Science" and "Scopus" between years 2010 and 2021 with chosen terms "context-aware systems", "automation", and "decision-making".

Figures 2 and 3 show the number of publications of IoT in combination with "context-aware systems", "automation" and "decision-making". These figures show that the noted topics are respected and considered for research. The number of publications in the respected area is increased year by year. Automation in IoT is increasingly attracting the research community's interest with a 43% growth in publications on automation in 2018 compared with 2017, and a 60% growth in 2017 compared with 2016. This shows that the topic of automation in the ecosystem is becoming very important and will attract more

interests from the research community in the future compared to other research topics such as context-aware systems and decision-making.

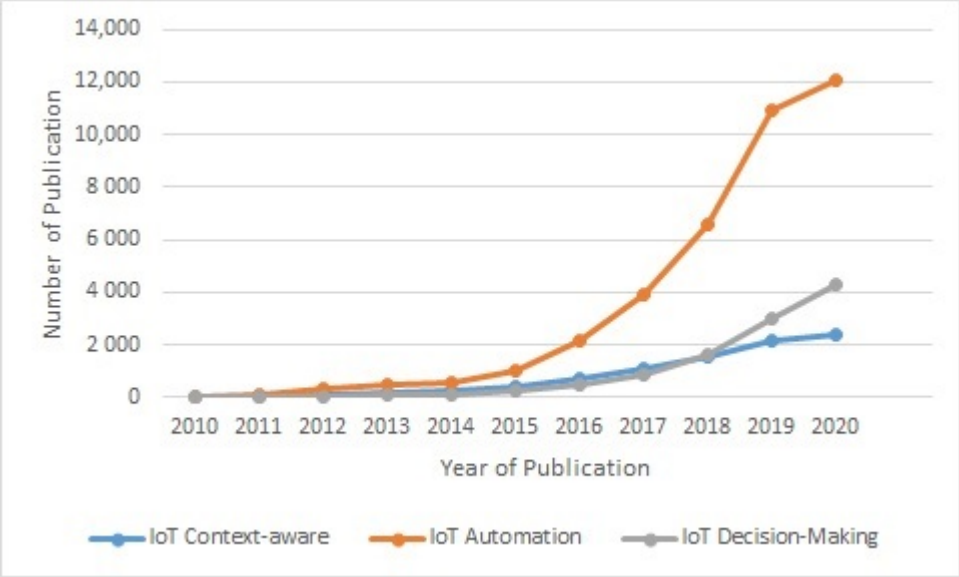


Figure 2. Number of publications of IoT in combination with "context-aware systems", "automation" and "decision-making" in the Scopus database.

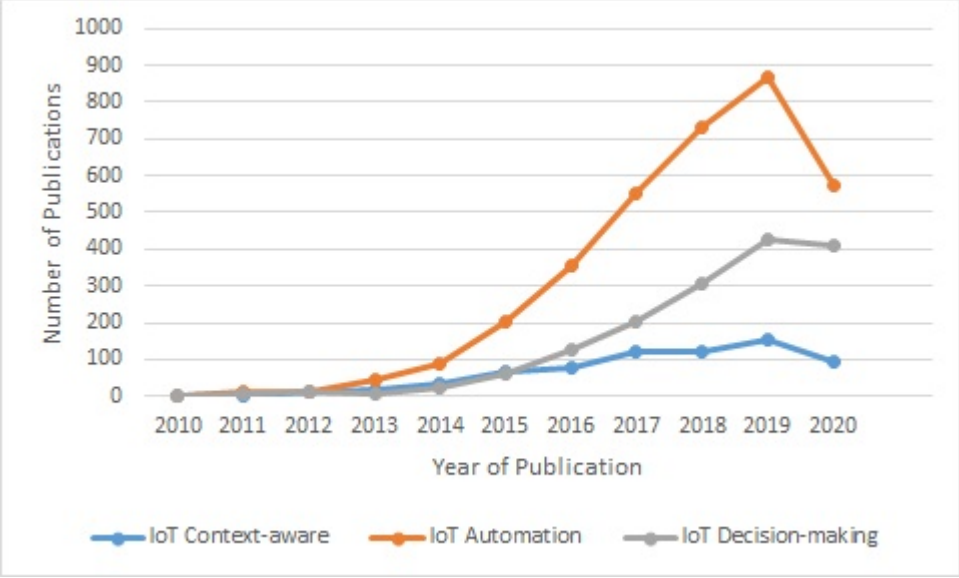


Figure 3. Number of publications of IoT in combination with "context-aware systems", "Automation" and "Decision-making" in Web of Science database.

As automation and IoT are the most important research topics, the "automation in IoT" and the algorithms, tools, technologies, and networking methods that can contribute to this topic require more attention. The main purpose of any IoT application is to help end-users in automating tasks, which are still manual. Once the IoT sensors are attached and the sensor data is introduced to the ecosystem, the data should be processed automatically to tackle the latency. Therefore, the main question of any IoT application is "How an IoT application should automate data processing in the ecosystem?". The objective of this survey is to represent a systematic discussion and analysis of different techniques, tasks, algorithms, and operations that contribute to a particular automation framework in the ecosystem. Each automatic function is designed and considered to resolve a particular

challenge. The survey discusses the purpose, tasks, operations, and techniques used in each automation as well. Discussing and analyzing the ecosystem with an abstract automation perspective has not been conducted yet. Having a systematic analysis of different high-level automatic aspects of the ecosystem can provide us with a clear road map in the design and implementation of diverse IoT systems.

The contribution of this survey paper is to design, frame, and suggest automatic components by studying and analyzing the existing literature and IoT application. Our study shows the main types of automation, which can underpin any IoT application. We have demonstrated the main types of automation, which can help researchers evaluate IoT applications to gain a better grasp of automatic data processing.

The rest of the paper is organized as follows. Section 2 discusses survey articles on IoT applications and architectures. Section 3 describes the major challenges in the design and implementation of IoT applications. Section 4 shows the six automatic functions used in the framework. Section 5 shows the motivation and the work of reviewed papers and their contribution to automation.

2. IoT and Fog Applications and Architectures

In this section, some important survey papers that have studied the IoT applications and Fog computing are reviewed by presenting the studies' achievements and limitations.

Gil et al. [12] classifies the IoT applications into industrial, health, and smart city. Farm management applications are an example from the industrial domain where sensors measuring different parameters of soil can increase the real-time farm-monitoring knowledge. This knowledge can then help the farmers to make informed decisions that can lead to an increase in the soil quality. Farmers using actuators will be able to take proper actions to meet soil threshold. Remote patient monitoring is another example from the health domain. The sensors measure the patients' life parameters. If a certain threshold is met, real-time preventive actions will be taken, or real-time reports will be sent to the medical staff. Traffic management is a practical application from the smart city domain. By equipping vehicles with Vehicle Speed Sensors (VSS), the generated data will be sent to the Fog layer. The decision-making actions will then manage the traffic lights to avoid congestion on the roads.

Perera et al. [13] studies several cases of smart city applications such as smart transportation, smart agriculture, smart health, smart waste management, and smart water management. Their survey depicts a comprehensive study of types of sensors, actuators, and Fog nodes, along with their responsibilities in providing a service in a smart city application. The authors illustrate the types of communications, protocols, and network existing in the smart city application. However, there is no discussion and insight into the process and task automation in the application.

Naha et al. [14] analyzes the definitions of Fog layer with a discussion of the taxonomy of Fog computation and architectures. They depict the types of software, platform, and architecture in the ecosystem. Additionally, they illustrate the attributes, key features, advantages, and weaknesses of each Fog-related architecture that serves IoT applications.

Maag et al. [15] presents a case study of air pollution in the IoT layer. It focuses on IoT sensor layer functionality for an air-pollution IoT application, showing how to deal with errors in the sensor layer in order to make the IoT application more responsive and accurate. Different attributes of sensors, such as errors, calibrations, and strengths, are discussed as well. The interconnection among sensors is elaborated but with no discussion on sensor collaboration and communication with other IoT layers such as the Fog. The sensor data and the calibration of the sensor layer to a network are studied. However, the article lacks a study on automating sensor selection and sensor data analysis.

Mukherjee et al. [16] studied structural differences between Fog and Cloud layers. They addressed some well-known IoT applications such as smart city, smart transportation, and healthcare and the role of Fog computing in managing the data transfer for these applications.

Al-Fuqaha et al. [17] discusses the operating systems(OS) of IoT services. The authors review communication techniques and protocols in the IoT layer. The network protocols , along with their packet/frame structure are explained and characterized in the ecosystem. They mainly focus on protocols, packets, and frequencies suitable for IoT applications.

Yassein et al. [18] reviewed the Message Queuing Telemetry Transport (MQTT) protocol in detail as it is the most important lightweight protocol for IoT. They addressed the non-reliable network, low-bandwidth management, and low-memory as factors, which can lead to high latency in real-time applications. The Machine-2-Machine (M2M) interaction and interoperability are carried out by the MQTT protocol. As the MQTT protocol is a well-known protocol in dealing with heterogeneity in devices and hardware running platforms, dealing with big data needs to be studied as well.

Maheswari et al. [19] discuss the security of the ecosystem in data integrity and verification when unwanted, suspicious tasks in the ecosystem happen. They study suspicious behavior such as data deletion, data alteration, and data duplication in the ecosystem. Several schemas are suggested for data security, which are signature-based schema, probability-based schema, tag-generation schema, and table-based schema.

Hathaliya and Tanwar [20] provides another insight into security issues in a healthcare domain. The research reviews and analyses various methods and algorithms for data encryption, and data classification in centralized and decentralized ecosystems.

Markus and Kertesz [21] depicts the software simulators of processes in the ecosystem. It counted the examples of software simulators with their advantages and shortages for the Cloud, Fog, and IoT researchers.

Yangui [22] surveys IoT services such as Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) to the users. They analyze several service solutions with different IoT applications and platforms as the criteria. Studying applications in heterogeneous environments, environments with IoT–Fog interaction, environments of IoT–Cloud interactions are among the criteria.

Varshney and Simmhan [23] surveys the scheduling in the ecosystem by providing a taxonomy on resource allocation. The study shows the taxonomy based on different metrics such as type of scheduling, techniques of scheduling, pricing, and Quality of Service (QoS)

All the reviewed papers have studied the characteristics of the communication, network protocols, and security with no concern on automation in the ecosystem. Table 1 shows the reviewed papers with their topic and key aspect. Some of the papers consider a particular IoT application and study the protocols/ecosystem architecture used in the application. For instance, Maag et al. [15] considers an air pollution application and focuses its study on the sensor layer, their calibration, and accuracy. Yassein et al. [18], as another example, studies the MQTT protocol in the ecosystem, which is one of the efficient and light-weight protocols in the ecosystem. Two articles illustrate the problem of security for data processing in the ecosystem.

Table 1. The concern and key aspect of the survey papers.

Paper	Topic	Key Aspect
Gil et al. [12]	IoT applications	smart city
Naha et al. [14]	IoT applications and architectures	Fog layer taxonomy
Mukherjee et al. [16]	IoT applications	ecosystem architecture
Perera et al. [13]	IoT applications	ecosystem protocols
Al-Fuqaha et al. [17]	IoT communication	network protocols
Yassein et al. [18]	IoT communication	MQTT protocol
Maag et al. [15]	air pollution application	IoT sensor layer
Maheswari et al. [19]	IoT data process	security
Hathaliya and Tanwar [20]	IoT healthcare application	security
Markus and Kertesz [21]	IoT simulation	software simulator
Yangui [22]	IoT application	IoT services
Varshney and Simmhan [23]	IoT resource allocation	scheduling

The simulation is an important aspect of illustrating task execution and latency because of the cost of execution of real IoT applications in the ecosystem. One paper examines different tools and simulation software programs in the ecosystem. Among the reviewed papers, the taxonomy of IoT applications, the taxonomy of network protocols and the taxonomy of ecosystem architecture, the taxonomy of simulators is covered. The automation of IoT applications and the types of functions and technologies leading to automation is not well covered and studied. Therefore, this survey paper has the motivation to base the study on automation and review the papers with an automation lens, which is one of the most important goals of any IoT application in the ecosystem.

Employing automatic mechanisms to resolve IoT challenges requires a survey study. The Fog layer should be intelligent and resilient to tackle two major challenges in the ecosystem: big data and heterogeneity

The aim of this survey paper is to study and discuss the IoT applications and their data processing mechanisms that form an essential point of view: automation. With this perspective, reviewed papers and IoT applications are discussed and analyzed along with their different aspects of task automation in the ecosystem.

3. Challenges in the IoT Layer

Many studies note the problems and challenges of data processing and task scheduling in the ecosystem. Maiti et al. [7] states the concept of fog nodes in processing light-weight data in the ecosystem and its deployments problems. The study notes the fog node deployment as an optimization problem in the ecosystem.

Reiss et al. [24] depicts the types of heterogeneity in the ecosystem, such as heterogeneity in resource type and duration and how the heterogeneity causes problems in high-performance computing and simultaneous processing. Singh and Chana [25] counts the resource uncertainty and resource heterogeneity in cloud computing. Tortonesi et al. [26] also described the problems of computation and network resources arising from heterogeneity.

Greco et al. [27] discusses the problems of emerging sensor data on the Internet and the rising big data problems, which makes data analysis more challenging. Wang et al. [28] shows the inefficiency of cloud computing and cloud resources and storages in data computation and task scheduling.

Figure 4 shows the three main challenges of any IoT application:

- Data offloading;
- Heterogeneity;
- Big data.

Data offloading: As IoT applications are mostly real-time, latent data processing and management are not acceptable. One way of reducing the latency is to minimize the data transfer between the Cloud and IoT devices and facilitate computation in the Fog layer. Hosting IoT applications in the Fog layer while satisfying their QoS parameters without relying too much on Cloud resources is called Fog offloading. Providing good data offloading requires improved task hosting and task execution in the Fog layer with satisfying QoS timing parameters.

Heterogeneity: Distributed small devices with light-weight computation power should compete with Cloud resources in order to make the Fog layer a better hosting environment for IoT related sensitive tasks. Each of these devices and hardware components come to the Internet with different architecture, hardware power, and capability. Heterogeneity exists in data, processors, software, and communication types. Data heterogeneity exists in two types: local data accessible to IoT and Fog layer, and global data accessible to all layers. Heterogeneity in the ecosystem's components includes IoT light-weights processing devices, sensor generator devices, cellphones, and end devices. Hardware platforms are also heterogeneous in the ecosystem and vary from IoT sensors, Fog gateways and processing units to sophisticated, high computational resources in Cloud

layer(data centers and servers). Software heterogeneity exists in tools, scripts, executing batch files, and machine learning applications.

Big data: Enormous data stream resulted from connecting an ever-increasing number of sensors and objects to the Internet, causes hosting, scheduling, processing, and decision-making tasks hard-to-tackle.

The automatic functions are defined in this paper to resolve the main challenges in any real-time IoT application. The communication types in the ecosystem are three types: machine-to-machine, machine-to-human, and human-to-human.

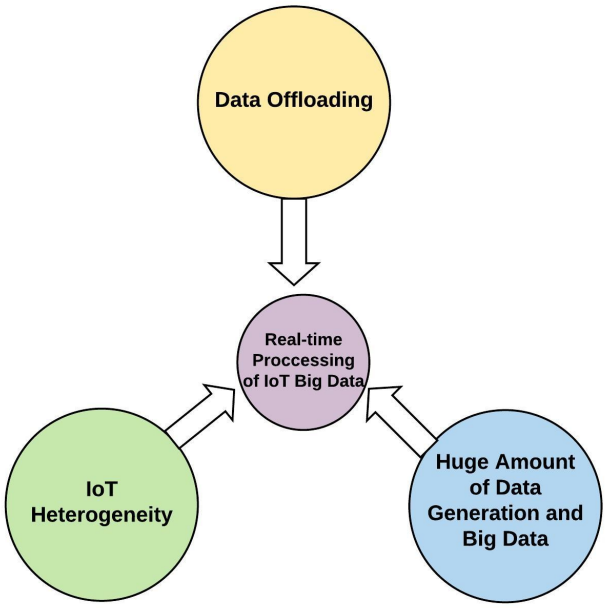


Figure 4. Three major challenges for any IoT real-time application.

4. Automation in the IoT–Fog–Cloud Ecosystem

As big data, ecosystem heterogeneity, and Fog offloading are the major challenges in any IoT application; automatic functions should be considered to deal well with the ecosystem challenges. This section discusses the concept of automation from a computer science point of view.

Uckelmann et al. [29] study the heterogeneous architecture of the IoT ecosystem. They show how the main function of any IoT application is to process and manage data in its life cycle, which starts with IoT sensors and terminates when delivered as a service to users. The data will be received to the ecosystem by IoT sensors as a data stream and will be converted to context afterward. The output context can be put to action or be converted to a proper service satisfying the desired user.

As an example, we consider an IoT logistics application, which should deliver items to the destination. Improving the transportation service, the responsible agents (services) should collaborate and communicate, such as delivering the item to the destination at the right time, maintaining the item's quality, providing some monitoring services to the customers, and doing all these with a minimum cost. Providing the services requires geographical data, quality data, and customer data at proper time-intervals. For any service, the data types integrate to produce the context and knowledge.

Uckelmann et al. [29] show the relationship between IoT, big data, and organizations. Here big data is collected by sensors, devices, humans, robots, environments, and AI applications. The authors state that automating operations in any IoT application is a major concern. The fundamental computer science questions of automation are: What can be automate and which part should be automated? Their work shows how a process can be automated efficiently and reliably. There is a shift from the paradigm of machine-centered

computing to human-centered computing. In machine-centered computing, there are three areas of study: theory, modeling, and design. In human-centered computing, a social study is also added. The authors also noted that automating things will lead to the creation of new technologies.

Denning and Staff [30] explain that search (finding a subset of participants in a decision-making process), deduction (making a decision based on knowledge and facts and existing rules), induction (creating the model of decision-making pattern for each event = acquiring a general rule), and collective intelligence (integrating event rules for a specific decision process) are the types of tasks that can be automated with computer science perspective. In any IoT application, data can be divided into tasks which can be further divided into sub-tasks [30]. Each sub-task can then be automated according to the automation defined for each type of task. Therefore, the first step of automation design is to classify IoT operations according to search, deduction, induction and computational intelligence [30].

Chegini and Mahanti [31] has proposed a framework for task automation in the ecosystem regarding the three challenges. They discuss the automation in different aspects such as learning the context, Machine-to-Machine (M2M) communication, and tasks' scheduling.

4.1. Automatic Context Management

The knowledge of any IoT application comes from the IoT sensors and end devices. Integrating different IoT datastreams produces the context of any IoT application as well. The context can then be processed to provide the services for IoT users. We call this process automatic context management. One responsibility of the automation is to prepare and pre-process data before delivering to other components. As the data comes in a stream from IoT sensors, getting the data from sensors and converting it into a file, table, or query is another responsibility, which is carried out mostly in the lower layers. This automation is also the main part of decision-making, rule firing, and service delivery. Once the users' requests come to the knowledge component, it is the rule-based component's responsibility for deciding which rules in terms of if-then propositions should be fired. The context component, rule-based component, and the machine learning algorithms (such as fuzzy rules) which execute the rules, can collaborate in this automation.

4.2. Automatic Task Workflow

Any IoT task can be divided into some sequential or parallel subtasks executed to satisfy a single user request. The responsibility of this automation is to recognise whether a task is sequential or parallel This automatic component is called an automatic task workflow.

4.3. Automatic Data Transfer

An important decision for IoT tasks is to choose the best IoT tasks host to consider their SLAs. In order to satisfy the SLAs, some tasks need to transfer to the Cloud, and some need to host at the Fog. The SLA parameters coming to this decision are QoS, energy efficiency, and local-awareness [32]. Here, the motivation is to host the IoT tasks on the Fog in order to increase the Fog offloading. We call this functionality the automatic data transfer.

4.4. Automatic Orchestration and Collaboration

As the IoT ecosystem contains diverse distributed components in three layers, managing and coordinating these parts for IoT tasks and producing the desired service requires an adaptive, flexible functionality. Orchestrating running components, resource execution, and machine collaborations are the role of the automatic function. We call this automation, automatic orchestration and collaboration. The automation assigns required processes to IoT task with SLAs based decision making. Another function of the automation is resources expanding and shrinking when a number of tasks are varied. This is called

resource elasticity. Assun et al. [33] discusses the resource elasticity according to different processing metrics.

4.5. Automatic Data Filtering

Data filtering, sampling frequency tuning, and trimming the extra information that is not required for the tasks, are among the solutions for big data processing. The functionality responsible for this task is called automatic data filtering. Trimming less important data and reducing the sampling rate will help with big data storage.

4.6. Automatic Task Scheduling

Any IoT task requires its context and processors. The context management, orchestration and collaboration automation are responsible for providing context and process requirements for any IoT task. Once the context and process are defined, they need to be scheduled tasks for execution which is a challenging. Many factors can determine the order of IoT tasks' execution, such as time, latency, cost, and energy defined in SLAs [34]. First Come First Serve (FCFS) is one scheduling strategy that executes the tasks according to their arrival time. In FCFS, the first IoT task coming to the ecosystem is executed first. Choosing adaptable scheduler algorithms for different IoT applications that schedule tasks according to different SLAs, is the responsibility of the automation.

Figure 5 demonstrates how these automatic components deal with the three challenges previously shown in Figure 4. The IoT devices and end-users are sending their sensor data to the data filtering component, which is the first automatic component for data processing in the ecosystem. The data transfer is also connected to the Cloud for when the data is required to be sent back to the Cloud for high computational processing. The context management is responsible for learning the context and producing the knowledge of the ecosystem. In addition to knowledge management, context management is responsible for any decision making in the context. The task scheduling component schedules, hosts, and executes the IoT tasks. The orchestration and collaboration automatic function communicate with other automatic components.

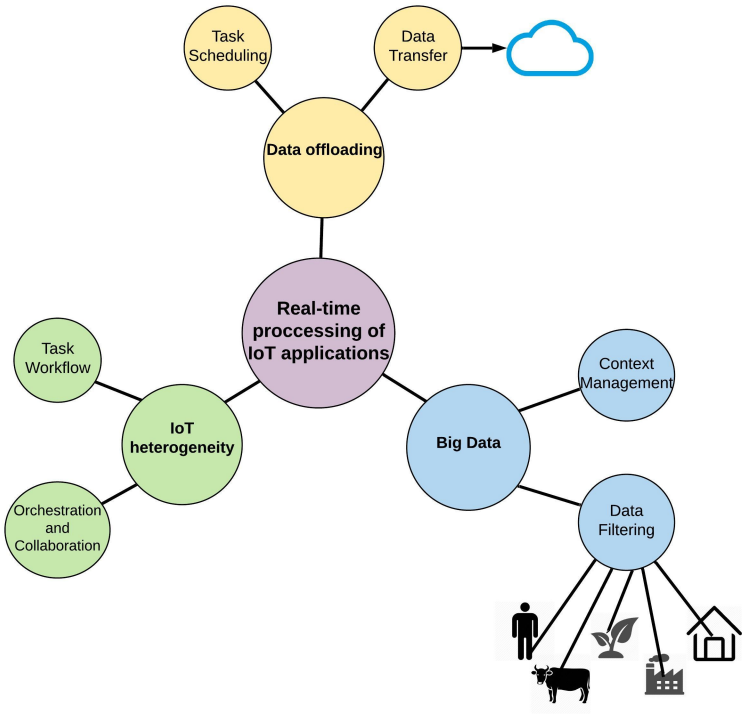


Figure 5. The main automatic components with their contribution to three major IoT challenges.

5. Fog Intelligence and Automation

This section reviews the automatic aspects of the IoT–Fog computing literature. Each automatic aspect is described while covered by the relevant papers. All the relevant papers are then discussed by their employed methods, algorithms, and studies around the automation.

5.1. Automatic Data Filtering

Kertiou et al. [35] discuss how the data from the IoT layer could be filtered by autonomous sensor selection for a specific user request. A technique called Skyline selects the most suitable sensors and filters the most important information and trims the less important information.

Perera et al. [13] note that for minimizing the energy of different components in the ecosystem, various factors should be considered, such as data sampling from IoT sensors, data conditional gathering, and communication frequency. Data science and feature engineering can support analyzing the data features and qualify any feature data that may increase communication costs and energy. After feature engineering and feature recognition, the framework software will be able to control and manage the energy based on data sampling.

Giaffreda et al. [36] studies a traveling IoT application. This application suggests and recommends accommodation, transport, and food for the travelers based on their location data in their 5G cellphones. The normal sampling rate of data can be defined as one data point every minute. The automatic data filtering can reduce the sampling rate to one in every 30 min when the tourists are slept and have no movement activity. According to the three mentioned studies, the challenge of automatic data filtering is to make data sampling flexible. Table 2 shows each paper's method, which is used to implement the automatic data filtering.

Table 2. Automatic data filtering methods and techniques.

Paper	Method/Technique	Description
Kertiou et al. [35]	Skyline	sensor selection
Perera et al. [13]	Feature engineering	using data science techniques for highest feature selection
Giaffreda et al. [36]	Data sampling technique	data analysis and selection based on day time activity

One major issue not addressed among reviewed papers is how to adjust the data sampling in an IoT application accordingly. For any IoT application, there are time intervals when high-frequency data samples are required. There are also time intervals when low-frequency data samples suffice. Finding essential factors that determine the frequency of data sampling and density of data is the responsibility of this automation, which is not thoroughly examined by the reviewed studies. As the data sampling rate of IoT sensors can seriously affect the volume of data and complexity of IoT tasks and other automatic functions such as context management, orchestration, and task workflow, this automation should be reviewed and discussed in detail. Therefore, a study on different data sampling rates and their effects on IoT tasks is needed.

Table 3 shows the selected papers with their studied data filtering management. Many have no concern about the flexibility of sampling rate in their studies. Therefore, there is a lack of analysis among them. Bittencourt et al. [37], Kumar and Prakash [38], and Osman [39] analyze the number of users and modules but not the data volume and sampling rate. Forkan et al. [40], Giaffreda et al. [36], and Garcia-de Prado et al. [41] discuss the sampling rate but without any concern on automatic function.

Table 3. The review and analysis of papers of their concern on data filtering.

Paper	Application	Sampling Rate Concern or Limitation on Data Filtering
Bardram and Christensen [42]	health care monitoring	lack of sampling rate tuning in application
Bittencourt et al. [37]	gaming	Cloud module dynamicity defined based on number of players but not data sampling
Forkan et al. [40]	health care monitoring	a sampling rate based on user requests is analysed but not automated
Garcia-de Prado et al. [41]	healthcare monitoring	a sampling rate based on user requests is analysed but not automated
Esmaeilian et al. [43]	waste management application	lack of discussion on sampling rate
Kjeldskov et al. [44]	smart city: transportation	lack of discussion on sampling rate
Kumar and Prakash [38]	smart city	big data discussion such as data integration,shuffling,cleaning but not data filtering and sampling rate
Luan et al. [10]	smart city: shopping application	lack of discussion on sampling rate
Arbanowski et al. [45]	smart home	lack of discussion on sampling rate
Nandyala and Kim [46]	smart home	lack of discussion on sampling rate
Giaffreda et al. [36]	travel guide	two defined sampling rate on day time and night time
Newcomb et al. [47]	smart city: shopping application	lack of discussion on sampling rate
Osman [39]	smart city	an analysis on types of data such as batch,real-time but not sampling rate and its automation
Perera et al. [13]	smart city	lack of discussion on sampling rate

5.2. Automatic Data Transfer

Any IoT task or request needs to be hosted firstly. Each layer in the ecosystem has its capability in terms of hosting IoT tasks. The IoT layer with cheap and distributed resources is close to users. This layer, however, cannot handle complex, resource-demanding tasks. The Fog has more capabilities and efficiencies but still not suitable for high computation such as Machine learning (ML) applications. The Cloud is enriched with sophisticated resources and components such as servers and clusters but far away from users. Choosing the best hosting for any IoT task for satisfying their SLAs is the responsibility of the automatic data transfer.

Aazam et al. [48] suggest that every task hosts in the Fog layer by default, and if extra resources are needed, they will be transferred to the Cloud. The tasks are classified as critical(time-sensitive) and non-critical ones. The automation of data transfer is then carried out based on the classification. If a task is classified as a time-sensitive task and needs extra resources, resources should be provided by Fog nodes rather than Cloud hosts.

Similar to previous studies, Rescati et al. [49] show another aspect of data classification for events in a health monitoring application. The information is classified into event data, and raw data and event data is sent to the Cloud for further computation. This classification determines when tasks need to be transferred to the Cloud for storing or to the servers for computation.

Yi et al. [50] propose the Virtual Machine (VM) migration as a data transfer solution for failure handling and fault tolerance. The VMs need to be moved between different processing hosts to meet response time SLAs. Many times VM migration is to meet energy efficiency. Their study shows a classification algorithm on data types.

Mukherjee et al. [16] notes that computational time-sensitive tasks are better to be hosted in the Fog layer. By any QoS violation or resource overutilization, the tasks can be moved to the Cloud. By this method, any IoT task which requires a computational resource has the chance to be hosted in the Fog layer before migration to the Cloud.

Dastjerdi and Buyya [51] show the problem of data overflow by devices that are connected to IoT sensors. Data transmission latency between the IoT and the Cloud is addressed as a problem, which makes the applications' QoS unacceptable. They call Fog nodes private Clouds and depicts them as a potential solution for QoS response time improvement.

Atlam et al. [52] explain that IoT tasks can be classified into two categories: time-sensitive tasks and batch tasks. The time-sensitive tasks should be hosted in the Fog layer to satisfy QoS timing criteria, and batch tasks are better to transfer to the Cloud data centers. Nandyala and Kim [46] show which functions need to be transferred to the Fog, and which should be hosted on the Cloud layer. The classifier algorithm divides tasks into those that require fast decision-making and those that need latent big data analysis. They use an IoT game application as a use case to show how data transfer between hosts impacts network usage.

Mehdipour et al. [53] define a smart Fog-engine component, which is responsible for data analytics to support decision-making on an IoT application. The decision can automate the data transfer to proper hosts. The Fog layer is considered as the first hosting environment and the Cloud resource as the alternate one. The Fog engine can be a Fog node, smart device, or a network gateway in the Fog layer, which can analyze, process, and manage the hosting environment of a task.

Table 4 shows the papers with their techniques and methods to implement automatic data transfer. Decision making on data transfer are based on different classification algorithms. The latency and cost of running each algorithm on the ecosystem should be studied as well.

Table 4. Automatic data transfer methods and techniques.

Paper	Method/Technique	Description
Aazam et al. [48]	Task classification	classifying tasks into sensitive and non-sensitive for data transfer
Rescati et al. [49]	Task classification	classifying tasks into event data and raw data for data transfer
Yi et al. [50]	VM migration	fault tolerance management with VM migration
Mukherjee et al. [16]	Task classification	classifying tasks into computational and time sensitive for data transfer
Dastjerdi and Buyya [51]	Private Fog nodes	Making some Cloud nodes private as Fog latency improvement
Atlam et al. [52]	Task classification	classifying tasks into time-sensitive and batch tasks for data transfer
Mehdipour et al. [53]	Smart Fog engine	moving data analysis from Cloud to Fog

5.3. Automatic Orchestration and Collaboration

The automatic orchestration and collaboration is a high-level function to deal with heterogeneity in the ecosystem defined by this paper. Providing automation for orchestrating various and distributed devices to collaborate on IoT tasks is the defined role for the automation. This section discusses the orchestration and collaboration functionalities in network protocols, storage, software, and hardware platforms.

Virdis et al. [54] showed how to optimize messaging and communication between machines, devices, or sensors. There are cases in which the IoT devices need to send or receive data directly from or to the Cloud. Therefore, it is important to minimize messaging among the layers: IoT–Fog , Fog–Cloud, and IoT–Cloud.

Prabavathy et al. [55] remarks that in an IoT application, machines create social rings to collaborate and meet the requests coming from the users. The machines' social ring is a social network among machines similar to the one that exists among humans. The machines or software components collaborate in the ecosystem for decision making and executing IoT tasks.

Each program is an agent in the ecosystem responsible for providing a service to another agent. For any IoT request, the machines form a chain to process the data and deliver services to the users. The chain of all machines (or agents) in the software system is called Service Oriented Architecture (SOA) or Microservice software architecture.

Buzachis et al. [56] presents an experimental evaluation for response time measurement in Microservice software architecture while Microservice components communicate through different network protocols such as File Transfer Protocol (FTP) and Constrained

Application Protocol (CoAP) protocols. Automating plug and play IoT sensors to the Fog ecosystem is another concern of Microservice software. The role of Kubernetes technology in orchestration for load balancing has been described as well.

Truong et al. [57] shows that the problem of heterogeneity requires grouping of different network components in the ecosystem known as ensembling. For different requests and tasks, the ecosystem's components ensemble to provide services.

Interoperability is an essential attribute in the ecosystem where different agents with different structures and software platforms interact and exchange data. From this perspective, any service can be a low-level service or a high-level service. A low-level service is more defined on hardware platforms and low-level network protocols, while the high-level service is a software module. Therefore, ensembling services(low-level or high-level) in any layer(IoT, Fog, and Cloud) could be the aim of Microservice software system. Guerrero et al. [58] highlighted the ensembling as a challenge.

A ubiquitous light-weight network protocol suitable for machines' communication in the ecosystems is MQTT protocol. Yassein et al. [18] notes that MQTT is the primary communication protocol in an IoT application. This protocol works based on publish-subscribe messaging with a light-weight data transmission overhead, which is suitable in the Fog layer with limited resources. MQTT consists of different components, such as brokers, publishers, and subscribers. Brokers have an essential role in connecting agents in MQTT. The publishers send data on the network, and the subscribers request for particular data. The brokers deliver the published data to the subscribers. This way of communication is easy and straightforward between machines. Other attributes of MQTT are its flexibility in managing low-bandwidth, low memory, and non-reliable networks that suffer from high latency.

Luan et al. [10] presented the role of Fog computing as a reliable layer between the Cloud and the IoT for real-time applications. This reliability is in communication between the IoT and the Cloud with just a few hops through Bluetooth or WiFi connection. They show that reliability is to host data close to the computational machines.

Luan et al. [10] show another role of Fog nodes, which is in providing local information to users not feasible by Cloud nodes. When the local information is not accessible by a Fog node, its adjacent Fog nodes collaborate and provide the information. Different geo-distributed Fog nodes can perform this cooperation among Fog nodes. Fog servers in one Internet Network Provider (ISP) can also connect with low-cost connections. The challenge here is to find the best Fog server based on users' locations. Predicting the users' requests and providing location-context services is another role of Fog computing.

In terms of storage management among different machines, two technologies are highlighted by Luan et al. [10]: Content Delivery Network (CDN) and Information-Centric Network (ICN). The CDN clone and caches the context to several servers instead of one to reduce the download delay. Any user requesting for any context can download it automatically from the closest server. On the other hand, ICN provides content distribution to mobile users. In terms of communication protocols, Al-Fuqaha et al. [17] study the standard network protocols in the Fog layer. CoAP, MQTT, Extensible Messaging and Presence Protocol (XMPP), Advanced Message Queuing Protocol (AMQP), Data Distribution Service (DDS), and Hypertext Transfer Protocol (HTTP) are the reviewed protocols by Al-Fuqaha et al. [17].

MQTT is a very lightweight protocol and can be the first choice of a network protocol for an IoT application. This protocol works based on the publish-subscribe method. IoT sensors are publishers here, and apps will play the role of the subscriber. Once they acknowledge their connections to the brokers, the message can be transported between them. XMPP is an Extensible Markup Language (XML) based protocol with much larger overhead (compared to MQTT), which is suitable for IoT applications. In this protocol, the message format should be defined in the XML file, and it will cause the extra overhead.

AMQP is an Instant Messaging (IM) protocol suitable for IoT applications but not as lightweight as MQTT. AMQP is well-designed for communications between vendors that

have different business models. Because of reliability and flexible routing, this protocol can be the right candidate for communication between IoT objects and Cloud centers. DDS is another publish-subscribe protocol designed for M2M communications. This protocol is for data exchange between data readers and data writers. One advantage over MQTT is that in DDS more messages can be transported. While designing IoT architecture, suitable protocols should be designed based on their latency attributes.

While defining protocols among the ecosystem's components, many parameters should be considered, such as latency, overhead, types of communication, reliability, and security. The significant part of the automatic orchestration and collaboration is to define reliable network protocols for different components of the ecosystem.

Figure 6 shows a schematic view of the proposed protocols for communication between each component. Communication among components in the first and second layers is mostly MQTT. As the devices and objects in these layers have limited power and resource, this Protocol can be a suitable communication protocol. For components that have a more computational resource, XMPP can be the right candidate as well. Moving upward to the Cloud, AMQP, and DDS can be chosen because they are more flexible with supporting different types of messages and routing options. The non-constrained resources in the Fog layer and Cloud layer can process the overhead of heavyweight protocols such as AMQP and DDS.

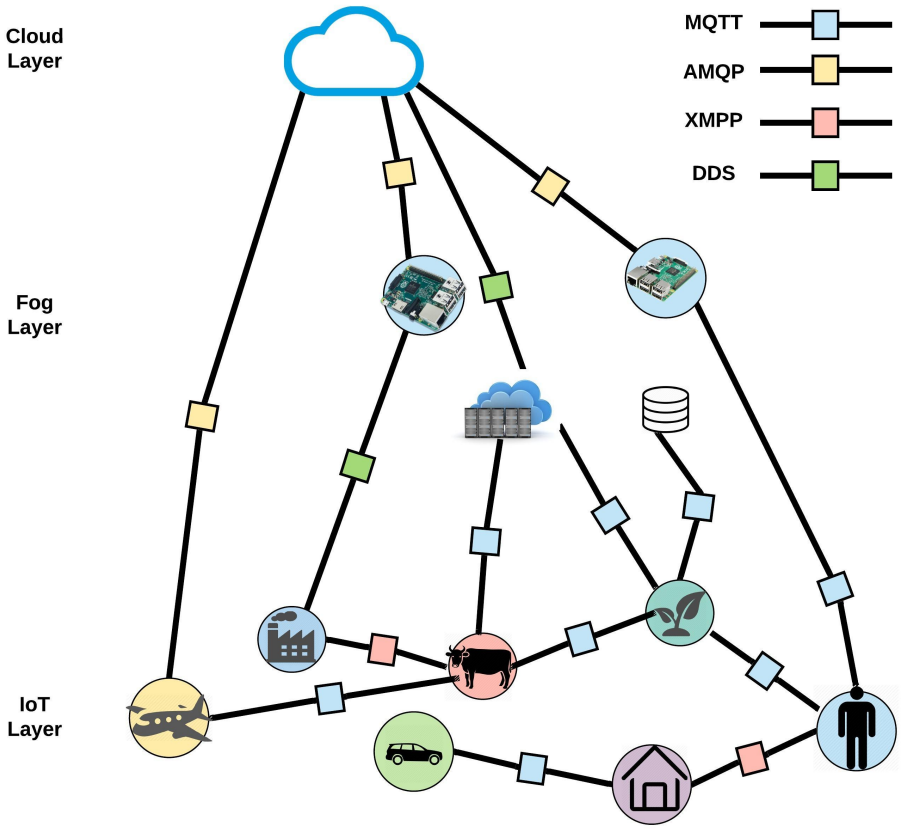


Figure 6. The IoT–Fog–Cloud ecosystem with the preferred protocols.

Kim et al. [59] discuss the Microservice software for automatic collaboration among various components. The Microservice is a distributed software structure contained different components in size and code types well suited for heterogeneous environments in the ecosystem.

Krivic et al. [60] explains the structure of Microservice software and its preferences over a monolithic one. The comparison between Microservice software and monolithic

software shows that the former has the best suitability in the ecosystem. One challenge here is the conversion of a monolithic software to Microservice. Scalability, development, changeability, integration, communication, maintenance, interoperability, interconnection, and upgrades are the positive features of Microservice software. The interoperability and interconnection of different IoT components are discussed by Osman [39]. The collaboration between IoT sensors, Fog nodes, and Cloud nodes is discussed in [41] in the healthcare domain, where providing the best timing services for patients is so vital. Portmann et al. [61] discusses the fuzzy rules and linguistic variables for defining communications. This paper shows that linguistic variables can interpret the Person-to-Person, Machine-to-Person, and M2M communications very well.

Mejtoft [62] discusses the topic of value co-creation in collaboration of things with studying the waste management application as a use case in the smart city domain. Mejtoft [62] demonstrates that in an IoT application, a higher number of connected things proportions to higher economic value. In other words, more connected users, objects, and machines, the Internet means more economic value. The resources can expand or shrink to satisfy a particular task. The resources will be selected and released when the number of collaborators is changed, as noted by Rahmani and Kanter [63]. Table 5 demonstrates the methods and techniques among the reviewed papers in this section, which are used for implementing the communication and collaboration in the ecosystem.

Table 5. Automatic orchestration and collaboration methods and techniques.

Paper	Method/Technique	Description
Virdis et al. [54]	Message minimization	minimizing messages among components for better communication
Prabavathy et al. [55]	Machine social ring	society of machines for better communication and automation
Buzachis et al. [56]	Kubernetes	as a technology for automating load balancing
Truong et al. [57]	Network ensembling	integrating network protocols for automating messaging
Yassein et al. [18]	MQTT	a light-weight messaging protocol for flexibility management and automation
Luan et al. [10]	Fog node	as a technology for resolving latency in communications
Luan et al. [10]	Fog node	as a technology in low connectivity and localization
Luan et al. [10]	CDN	a technology in automating context delivery
Luan et al. [10]	ICN	a technology to deliver context to mobile users
Al-Fuqaha et al. [17]	AMQP	a technology for IoT–Cloud communication
Al-Fuqaha et al. [17]	XMPP	a technology for IoT communication with computational overhead
Kim et al. [59]	Microservice	a software architecture for communication and automation
Krivic et al. [60]	Microservice	a software architecture for interconnection, communication, and flexibility
Portmann et al. [61]	Fuzzy sets and rules	a technology to manage ecosystem communication with humanistic variables

5.4. Automatic Task Scheduling

This component deals with assigning timing resources to IoT tasks for data processing and meeting QoS parameters such as response time. Yi et al. [50] suggest a monitoring system with two checkpoints data: the state of the task's execution and the task's location. The data is recorded and used as metrics for scheduling, replication, and resource assignment.

Mutlag et al. [64] study the challenges of task scheduling and the latencies dealing with Electrocardiogram (ECG) data in the healthcare domain. Fog nodes pre-process the tasks by applying networking functions for schedule.

Saqib and Hamid [65] introduce the FogR small computation units as other types of Fog nodes for intelligent decision-making. The FogRs are designed to engage the heterogeneous devices to computation in a smart traffic system. The devices such as traffic light, cameras, sensors, and pedestrian cellphones will use their small and light-weight

processing resources in computation. Therefore, the primary responsibility of FogR is to engage more IoT and Fog resources if Cloud computing is not accessible.

Bittencourt et al. [37] demonstrate the Cloudlets as another solution for increasing the efficiency of an IoT gaming application. As accessing Cloud resources is competitive with high latency, the Cloudlets are introduced to the ecosystem as other types of brokers. Increasing the number of Cloudlets can boost the efficiency of this solution.

The resource assignment is a vital task for any computational process in the ecosystem. The technologies such as Cloudlets and FogR are implemented for decreasing the task execution latency. In the ecosystem, the resources exist in different types and capabilities, and therefore an accurate resource monitoring algorithm is required for controlling any resource assignment. The cooperation of resource monitoring algorithms and resource assignment algorithms will complement the automatic task scheduling function. Table 6 shows the methods and techniques used for automating the task scheduling and management in the ecosystem.

Table 6. Automatic task scheduling methods and techniques.

Paper	Method/Technique	Description
Yi et al. [50]	Monitoring technique	a technique for task scheduling and resource management
Mutlag et al. [64]	Fog node	a technology for task scheduling
Saqib and Hamid [65]	FogR	a technology for intelligent decision making
Bittencourt et al. [37]	Cloudlet	a technology for IoT application efficiency

5.5. Automatic Task Workflow

It is evident that for any action, request, or change in the ecosystem, a series of tasks need to be undertaken. Computer network programmers, IT experts, and IoT skilled people should decide which tasks in the ecosystem should be automated. The correlated tasks and predictable tasks can be designed and executed by the task workflow automation. The task workflow function can be updated by new actions and tasks coming to the ecosystem [66].

Seiger et al. [67] demonstrate a workflow as a way of doing repetitive tasks autonomously. The morning process and healthcare emergencies are two IoT applications shown by proper workflow model design Seiger et al. [67]. The morning process is divided into sub-tasks such as initiating robot => delivering the task of paperboy => turning the light on => preparing the coffee. Some of these tasks are divided and converted to an object module with their execution time. For the emergency application, the tasks are considered as health status => call emergency service => medical personal arrives => unlocking door.

Hao et al. [68] describes the task scheduler as another type of automation in the ecosystem. Here, the Fog node is denoted as a device dedicated for scheduling workloads and applying policies. The workflow task automation can be tied to the context management automation as a note by Seiger et al. [67]. Furthermore, Mukherjee et al. [16] remarks the adding scalability attribute to workflow automation when new users are coming to the system or new IoT sensors are plugging in. The interconnection between workflow automation and context management need to be studied. The scalability of tasks and components is a significant challenge when an IoT application is growing in terms of its devices, software, data, and users. In such situations, the tasks and services need to be extended as well to satisfy SLAs.

Figure 7 shows a diagram of a primary task called workflow 1 divided into 6 sub-tasks. The left-hand side of the figure shows the IoT layer where data streams are coming to the workflow boxes for processing. Tasks 1, 2, and 3 can be executed in the same resource as they have a serial structure. Tasks 3, 5 and tasks 4, 6 are parallel tasks that require parallel resources to be performed as well. This example shows that it is essential to detect the resources needed for tasks before any action. The Fog can then prepare the resources and runs all the tasks at the right time. The yellow circles show the required manual point in

workflow 1 that should be applied after the execution of task 2, or 3. The grey circle is the endpoint of the main tasks. By increasing the users, requests, or resources, the tasks need to be extended as well, and this is the responsibility of the orchestration and collaboration component. Therefore, a collaboration between workflow automation and orchestration and collaboration automation is necessary. Table 7 shows the methods and technologies of two reviewed papers for conducting the task workflow in the ecosystem.

Table 7. Automatic task workflow methods and techniques.

Paper	Method/Technique	Description
Seiger et al. [67]	Task repetition	defining object module for each task in an IoT application
Hao et al. [68]	Fog node	a technology responsible for workload and policy scheduling

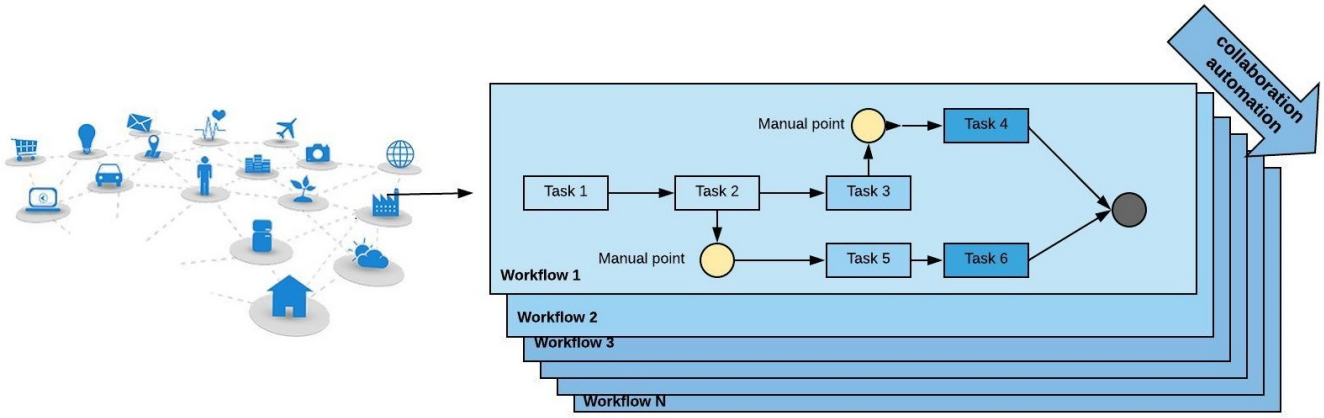


Figure 7. The Workflow diagram.

5.6. Automatic Context Management

The primary task of IoT applications is to process raw data and provide the knowledge context of the ecosystem. The system conducting the process is known as a context-aware system. Context-aware systems come from a time when new information such as location, time, and identity of mobile owners introduced new context data for processing on the internet.

Bisgaard et al. [69] note that context is any information sensed by a system input and processed by system functions. The context can also affect the system's behaviour. Identity, time, and location are examples of contexts introduced by mobile phones to the internet. Kjeldskov et al. [44] noted that a routing context-aware system could be defined by processing location, time, and identity as input to calculate the best-optimized tour between users and their traveling destination. The system should keep its reliability when an interesting number of spots of users is increased. Processing the contexts and providing the services out of contexts are roles of context-aware systems. The context-aware system should react to system inputs automatically without relying too much on human intervention.

The context-aware software has four dimensions as explained by Bisgaard et al. [69]: manual information, automatic information, manual command, and automatic command. By any context modification or change, the system can react, whether automatically or manually.

The fundamental questions in any IoT applications from this survey paper's perspective are which commands need to be automated. Defining and designing automating functions is essential for any IoT application. This section shows some examples of different context-aware systems to illustrate how various services can be defined and delivered to users by processing context.

Fithian et al. [70] define a meeting scheduling system by getting the location and peoples' identity as the input. Selecting the most suitable time-slots for the people in a meeting is the service of this context-aware system. The system should work in complex situations such as when the number of people is increasing or the time slots are reducing.

Cheverst et al. [71] define a guiding system that directs and routes travelers to their interesting spots in a city. In the routing system, thetheirtorical and new information is controlled by three buttons: "Back", "hold", and "update". By clicking the back button, the information of the previous position will be shown on the screen. By clicking the hold button, the information of the current place will be kept on the screen, and by clicking the update button, the system shows the information of the current location. Cheverst et al. [71] describes another tour-guiding system, which uses the location input and directs the travelers to their desired spots.

The human I-centric communication is a context-aware system noted by Arbanowski et al. [45]. The system receives environmental information and estimates the user's next actions/functions. As an example, by getting the user's profile, the brightness and temperature of the bathroom can be set up before the user's entrance. The user profile is updated by a component called context handler. Each family member can have a loadable profile in the context handler. Here, the system should provide the desired service for each user profile. The system can be complicated when the number of profiles increases.

Retail and shopping centers can also use context-aware systems for purchase optimization and customers routing to their desired items. Newcomb et al. [47] showed that the relationship between a store and a shopper could be analyzed to provide two services: recommended items for customers and Customers' shopping patterns for retailers.

Bardram and Christensen [42] shows another example of medical context-aware systems, which assist nurses in visiting patients in their rooms at the right time. The context-aware system here complicates by increasing the number of patients and their medical services.

Table 8 shows the context, service, and context-aware system of the reviewed papers. The context of time is used in routing and meeting scheduling systems. The context of the user's identity is used for I-centric systems and healthcare applications. It is evident that the location is used for all the context-aware systems and is the most critical context. In any automating system, the location, time, and user's identity are significant contexts and should be considered in the design and implementation of a context-aware system.

Table 8. The context-aware systems and their used context and provided services.

Context	Service	System	Paper
Location	Direction	Tour-Guiding system	Cheverst et al. [71]
Location & Identity	Care	Healthcare system	Bardram and Christensen [42]
Location & Time	Route	Route Systems	Kjeldskov et al. [44]
Location & Identity & Time	Housing functions	I-Centric system	Arbanowski et al. [45]
Location	Directions	Guidance system	Cheverst et al. [71]
Location & Identity & Time	Meeting Management	Meetup System	Fithian et al. [70]
Location	Direction	Shopping Assistant	Newcomb et al. [47]

In the ecosystem, IoT sensors generate data continuously, and each piece of data can be considered as an event. While receiving more events, the processing will be complex and is called Complex Event Processing (CEP). The CEP is the main component of the context-aware system and is responsible for processing events, providing the proper relationship among them, firing rules, and delivering services. The rules, events, actions, and services defined for a particular IoT application are called knowledge. By receiving an event, the related if-then rule triggers the output service. Zhang and Sheng [72] described that providing such connectivity among events, rules, and services can be extremely challenging.

Bruns et al. [73] describes that CEP is the combination of several simple events that reflect the state of a particular environment. The paper denotes the responsibility of CEP to deal with two types of events: events happening inside the ecosystem and external events introduced by IoT sensors into the ecosystem. The role of CEP is to determine how to deal with these two types of events.Bruns et al. [73] demonstrates the CEP in its three components: event model, which records all events for a specific domain, set of rules implemented by an expert known as event rules, and CEP engine, which is responsible for processing events.

Papageorgiou et al. [74] define the knowledge of the system as the unity of the mentioned three components. The architecture of CEP components is Event-Driven Architecture (EDA). In this architecture, an event processing agent (EPA) is an individual CEP entity that has its rule engine and rule base. The network containing all the EPA is called an Event Processing Network (EPN).

Naha et al. [14] describes how CEP could process mobile and cell phones' data streams in a collaborative deployment of telecommunication and IT networking. Garcia-de Prado et al. [41] discusses the CEP concept in a healthcare domain, where patients having breathing problems have resided in two hospitals. Forkan et al. [40] has also used the CEP to provide real-time services for patients having heart attacks.

Figure 8 illustrates an example of a healthcare ecosystem. We assume two IoT requests with their SLAs coming to the ecosystem. One is highly time-sensitive, and another one should be responded with high accuracy. For the first one, the context management and CEP selects sensor data of local areas such as A1, A2, and A3. It ignores the global information of the Cloud and other Fog nodes because of their latency.

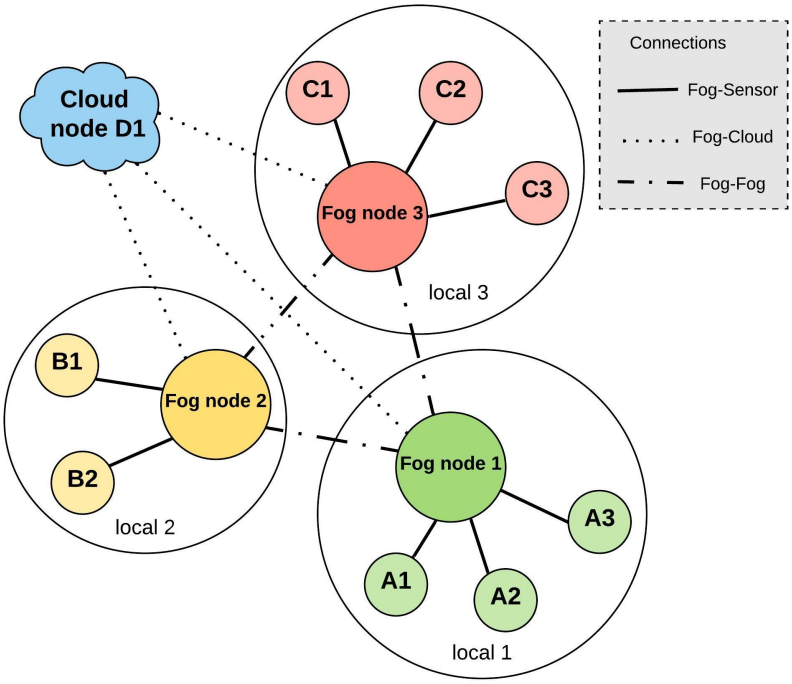
In the second request, the CEP selects A1, B2, C2, and D1 (all Fog nodes and global Cloud node) and integrate their data to produce a more accurate context. Here, global information such as air-pollution of the city can be used as extra information for event processing and decision making. The two scenarios show the role of a context-aware system in adaptive context selection and integration based on different SLAs.

Bruns et al. [73] shows that a combination of several components in CEP could lead to context management. The role of automatic context management here is to match between events and services. Papageorgiou et al. [74] has also applied the automatic context management to organize and order the knowledge of that particular IoT task. Kapitsaki et al. [75] shows how to gather context and how to design service from a specific context. Event Processing Language (EPL) is used for finding the relationship between causes and results by Kim et al. [59].

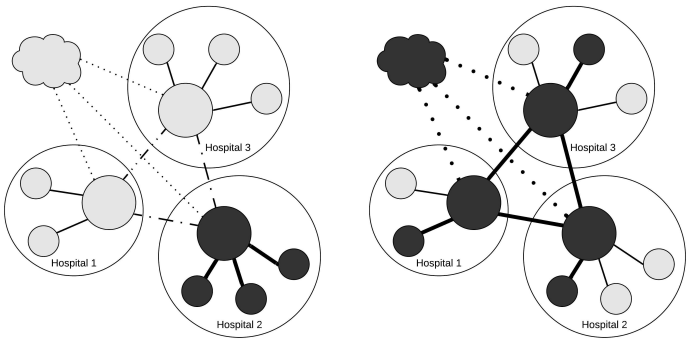
Garcia-de Prado et al. [41] shows a context-aware CEP model(COLLECT) in a SOA for handling big data in the healthcare domain. The context of local information in Fog nodes and global information in Cloud nodes is defined in COLLECT for achieving accurate prediction results in the healthcare domain. CoCoMAAL is a proposed context-aware system that processes and integrates different input such as person, place, and environment [40].

Kumar and Prakash [38] notes that an IoT smart city application is dependent on the country and geographical contexts. Therefore context management needs to process other related inputs such as country weather,theirtorical geo events, and the states of natural resources.

In the healthcare domain, some contexts such as ECG, Electroencephalogram (EEG), and Parkinson Speech data streams are analyzed and studied by Kraemer et al. [76]. The process and computation in a context-aware system with the aid of fuzzy linguistic variables, fuzzy rules, and fuzzy sets are considered by Portmann et al. [61].



(**a**) An example of components and their connections in the ecosystem



(**b**) M2M communication for SLA1 (**c**) M2M communication for SLA2

Figure 8. One example of ecosystem with two patterns of communication.

Hassani et al. [77] describes components of a context management system in a smart parking system domain. The components are Context Provides (CP), Context Consumer (CC), and the Context Query. One tool is noted here for processing the contexts: Context Description and Query Language (CDQL). The CDQL is responsible for processing the queries from CC and CP in both high level and low-level formats. The height of the car, fuel type, drivers' profile, preferred parking distance, and preferred cost is considered as the data constituting the context of a vehicle in the application.

Table 9 shows the components of a CEP system. Among the papers reviewed in this section, just one has proposed an architecture for event processing in a healthcare application [41]. As in any IoT application, on-line events come to the system continuously, the functions processing events and producing contexts should be flexible and automated.

Table 9. The description of CEP components in an event-driven IoT system reviewed and related to the section of context management.

No.	Name	Description
1	Context Event Processing(CEP)	The component in Fog responsible for dealing with streaming data and event
2	Event Driven Architecture (EDA)	The architecture of components in an event-driven IoT system
3	Event Processing Agent (EPA)	Any entity in a CEP system
4	Event Processing Network (EPN)	The network containing all CEP components and their inner and outer correlations
5	Context Provider (CP)	Any component responsible for generating and providing the context based on different events
6	Context Consumer (CC)	Any component which receives the context and consumes it
7	Context Description and Query Language (CDQL)	The data base and query language dealing with information as context

Figure 9 shows the abstract schematic view of automation concepts for an IoT real-time application. The aforementioned six automation components are illustrated with the discussed function, algorithms, or topics that can implement them. The VM migration technology and data classification algorithms can implement data transfer automation. For auto context management, EPA, CEP, and EPN are discussed topics and can complement this automation as well. Defining sequential tasks in the ecosystem can support workflow automation. Microservice is the proposed software architecture and MQTT, AMQP, and XMPP are examples of networking and communication protocols, which support orchestration and collaboration automation. Feature engineering is the supported knowledge for auto data filtering. FogR, Cloudlet, Droplet, and SmartFog Node are some examples of thought algorithms implementing task scheduling automation. Table 10 shows the types of automation noted by authors in their paper.

Figure 9. A high-level schematic view and a taxonomy of IoT automation components and their each sub-components.

Table 10. The evaluation of automation components among the reviewed papers.

	Automation Types					
papers	**Data Transfer**	**Task Scheduling**	**Data Filtering**	**Orchestration and Collaboration**	**Context Management**	**Task Workflow**
Aazam et al. [48]	✓					
Rescati et al. [49]	✓					
Mukherjee et al. [16]	✓					
Dastjerdi and Buyya [51]	✓					
Mutlag et al. [64]		✓				
Saqib and Hamid [65]		✓				
Bittencourt et al. [37]		✓				
Yi et al. [50]	✓	✓				
Atlam et al. [52]	✓					
Nandyala and Kim [46]	✓					
Mehdipour et al. [53]	✓					
Virdis et al. [54]			✓			
Kertiou et al. [35]			✓			
Prabavathy et al. [55]				✓		
Buzachis et al. [56]				✓		
Truong et al. [57]				✓		
Yassein et al. [18]				✓		
Al-Fuqaha et al. [17]				✓		
Luan et al. [10]				✓		
Krivic et al. [60]				✓		
Perera et al. [13]			✓			
Guerrero et al. [58]				✓	✓	
Kim et al. [59]				✓	✓	
Osman [39]				✓	✓	
Seiger et al. [67]						✓
Zhang and Sheng [72]					✓	
Bruns et al. [73]					✓	
Papageorgiou et al. [74]					✓	
Naha et al. [14]				✓	✓	
Garcia-de Prado et al. [41]				✓	✓	
Forkan et al. [40]					✓	
Kumar and Prakash [38]					✓	
Kraemer et al. [76]					✓	
Portmann et al. [61]				✓	✓	
Misra et al. [66]					✓	
Hassani et al. [77]					✓	
Kapitsaki et al. [75]					✓	
Mejtoft [62]				✓		
Esmaeilian et al. [43]						✓
Rahmani and Kanter [63]				✓	✓	

6. Conclusions

Although there are plenty of surveys on IoT tasks, IoT domains and case studies, the architecture, networks, and protocols, none of them has proposed a framework of automating different parts of the IoT ecosystem. This survey paper studies the tasks and operations of processing data in three different layers and provided a perspective of how automating functions are processing tasks in the ecosystem. The motivation of automation discussion is on three challenges in the ecosystem: big data, heterogeneity, and Fog layer resiliency. The automatic components are data filtering, data transfer, context management, orchestration and collaboration, task scheduling, and task workflow. The automatic components are also illustrated in sequential stages to discriminate their functionality, although there might be some common roles and tasks among them. Utilizing

the automatic components can over come the problems in Industry 4.0 and will lead to more collaborative robotic tasks in industry 5.0.

The future study of this research is to analyze the technologies doing context management and orchestration and collaboration automation. Conducting the required experiments to evaluate the techniques and their challenges when facing high heterogeneity and big data is essential and of interest in research as well.

Author Contributions: Conceptualization, H.C., R.K.N. and A.M.; Investigation, H.C. and A.M.; Methodology, H.C.; Supervision, A.M.; Writing—original draft, H.C.; Writing—review & editing, R.K.N., A.M. and P.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sniderman, B.; Mahto, M.; Cotteleer, M.J. *Industry 4.0 and Manufacturing Ecosystems*; Deloitte University Press: London, UK, 2016; pp. 1–23.
2. Corotinschi, G.; Găitan, V.G. Enabling IoT connectivity for Modbus networks by using IoT edge gateways. In Proceedings of the 2018 International Conference on Development and Application Systems (DAS), Suceava, Romania, 24–26 May 2018; pp. 175–179.
3. Geissbauer, R.; Schrauf, S.K.V. Industry 4.0-Opportunities and Challanges of the Industrial Internet. Available online: https://www.strategyand.pwc.com/gx/en/insights/2015/industrial-internet.html (accessed on 2 February 2021).
4. Frankó, A.; Vida, G.; Varga, P. Reliable Identification Schemes for Asset and Production Tracking in Industry 4.0. *Sensors* **2020**, *20*, 3709. [CrossRef]
5. Massaro, A.; Galiano, A. Re-engineering process in a food factory: An overview of technologies and approaches for the design of pasta production processes. *Prod. Manuf. Res.* **2020**, *8*, 80–100. [CrossRef]
6. Weerasiri, D.; Barukh, M.C.; Benatallah, B.; Sheng, Q.Z.; Ranjan, R. A Taxonomy and Survey of Cloud Resource Orchestration Techniques. *ACM Comput. Surv.* **2017**, *50*, 1–41. [CrossRef]
7. Maiti, P.; Shukla, J.; Sahoo, B.; Turuk, A.K. QoS-aware fog nodes placement. In Proceedings of the 2018 4th International Conference on Recent Advances in Information Technology (RAIT), Dhanbad, India, 15–17 March 2018; pp. 1–6. [CrossRef]
8. Groover, M. *Fundamentals of Modern Manufacturing: Materials, Processes, and Systems*; John Wiley & Sons, Inc: Hoboken, NJ, USA, 2020.
9. Deshmukh, U.; More, S.A. Fog Computing: New Approach in the World of Cloud Computing. *FInt. J. Innov. Res. Comput. Commun. Eng.* **2016**, *4*, 16310–16316. [CrossRef]
10. Luan, T.H.; Gao, L.; Li, Z.; Xiang, Y.; Wei, G.; Sun, L. Fog computing: Focusing on mobile users at the edge. *arXiv* **2015**, arXiv:1502.01815
11. Puliafito, C.; Vallati, C.; Mingozzi, E.; Merlino, G.; Longo, F.; Puliafito, A. Container Migration in the Fog: A Performance Evaluation. *Sensors* **2019**, *19*, 1488. [CrossRef]
12. Gil, D.; Ferrández, A.; Mora-Mora, H.; Peral, J. Internet of things: A review of surveys based on context aware intelligent services. *Sensors* **2016**, *16*, 1069. [CrossRef]
13. Perera, C.; Qin, Y.; Estrella, J.C.; Reiff-Marganiec, S.; Vasilakos, A.V. Fog Computing for Sustainable Smart Cities. *ACM Comput. Surv.* **2017**, *50*, 1–44. [CrossRef]
14. Naha, R.K.; Garg, S.; Georgakopoulos, D.; Jayaraman, P.P.; Gao, L.; Xiang, Y.; Ranjan, R. Fog Computing: Survey of Trends, Architectures, Requirements, and Research Directions. *IEEE Access* **2018**, *4*, 1–31. [CrossRef]
15. Maag, B.; Zhou, Z.; Thiele, L. A survey on sensor calibration in air pollution monitoring deployments. *IEEE Internet Things J.* **2018**, *5*, 1–15. [CrossRef]
16. Mukherjee, M.; Shu, L.; Wang, D. Survey of fog computing: Fundamental, network applications, and research challenges. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 1–30. [CrossRef]
17. Al-Fuqaha, A.; Guizani, M.; Mohammadi, M.; Aledhari, M.; Ayyash, M. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 2347–2376. [CrossRef]
18. Yassein, M.B.; Shatnawi, M.Q.; Aljwarneh, S.; Al-Hatmi, R. Internet of Things: Survey and open issues of MQTT protocol. In Proceedings of the 2017 International Conference on Engineering & MIS (ICEMIS), Monastir, Tunisia, 8–10 May 2017. [CrossRef]
19. Maheswari, K.; Bhanu, S.S.; Nickolas, S. A Survey on Data Integrity Checking and Enhancing Security for Cloud to Fog Computing. In Proceedings of the IEEE Xplore, Bangalore, India, 5–7 March 2020; pp. 121–127.
20. Hathaliya, J.J.; Tanwar, S. An exhaustive survey on security and privacy issues in Healthcare 4.0. *Comput. Commun.* **2020**, *153*, 311–335. [CrossRef]

21. Markus, A.; Kertesz, A. A survey and taxonomy of simulation environments modelling fog computing. *Simul. Model. Pract. Theory* **2020**, *101*, 102042. [CrossRef]
22. Yangui, S. A Panorama of Cloud Platforms for IoT Applications Across Industries. *Sensors* **2020**, *20*, 2701. [CrossRef]
23. Varshney, P.; Simmhan, Y. Characterizing application scheduling on edge, fog, and cloud computing resources. *Softw. Pract. Exp.* **2020**, *50*, 558–595. [CrossRef]
24. Reiss, C.; Tumanov, A.; Ganger, G.R.; Katz, R.H.; Kozuch, M.A. Heterogeneity and dynamicity of clouds at scale. In Proceedings of the Third ACM Symposium on Cloud Computing—SoCC'12, San Jose, CA, USA, 14 October 2012; ACM Press: New York, NY, USA, 14 October 2012. [CrossRef]
25. Singh, S.; Chana, I. QoS-Aware Autonomic Resource Management in Cloud Computing. *ACM Comput. Surv.* **2016**, *48*, 1–46. [CrossRef]
26. Tortonesi, M.; Govoni, M.; Morelli, A.; Riberto, G.; Stefanelli, C.; Suri, N. Taming the IoT data deluge: An innovative information-centric service model for fog computing applications. *Future Gener. Comput. Syst.* **2019**, *93*, 888–902. [CrossRef]
27. Greco, L.; Ritrovato, P.; Xhafa, F. An edge-stream computing infrastructure for real-time analysis of wearable sensors data. *Future Gener. Comput. Syst.* **2019**, *93*, 515–528. [CrossRef]
28. Wang, Y.; Yu, G.; Zhang, Y.; Han, Z.; Wang, G. (Eds.) *Big Data Computing and Communications*; Springer International Publishing: Shenyang, China, 2016. [CrossRef]
29. Uckelmann, D.; Harrison, M.; Michahelles, F. (Eds.) *Architecting the Internet of Things*; Springer: Berlin/Heidelberg, Germany, 2011. [CrossRef]
30. Denning, P.J.; Staff, U. *An Interview with Peter Denning on the Great Principles of Computing*; Ubiquity: San Jose, CA, USA, 2007; pp. 1–9.
31. Chegini, H.; Mahanti, A. A Framework of Automation on Context-Aware Internet of Things (IoT) Systems. In Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing —UCC'19, Auckland, New Zealand, 2-5 December 2019; ACM Press: New York, NY, USA, 2019. [CrossRef]
32. Sial, A.; Singh, A.; Mahanti, A. Detecting anomalous energy consumption using contextual analysis of smart meter data. *Wirel. Netw.* **2019**, 1–18. [CrossRef]
33. Assun, D.; da Silva Veith, A.; Buyya, R. Distributed data stream processing and edge computing: A survey on resource elasticity and future directions. *J. Netw. Comput. Appl.* **2018**, *103*, 1–17. [CrossRef]
34. Sial, A.; Singh, A.; Mahanti, A.; Gong, M. Heuristics-Based Detection of Abnormal Energy Consumption. Available online: https://link.springer.com/chapter/10.1007/978-3-319-94965-9_3 (accessed on 2 Feruary 2021).
35. Kertiou, I.; Benharzallah, S.; Kahloul, L.; Beggas, M.; Euler, R.; Laouid, A.; Bounceur, A. A dynamic skyline technique for a context-aware selection of the best sensors in an IoT architecture. *Ad Hoc Netw.* **2018**, *81*, 183–196. [CrossRef]
36. Giaffreda, R.; Cagáňová, D.; Li, Y.; Riggio, R.; Voisard, A. (Eds.) *Internet of Things IoT Infrastructures*; Springer International Publishing: Vicenza, Italy, 2015. [CrossRef]
37. Bittencourt, L.F.; Diaz-Montes, J.; Buyya, R.; Rana, O.F.; Parashar, M. Mobility-aware application scheduling in fog computing. *IEEE Cloud Comput.* **2017**, *4*, 26–35. [CrossRef]
38. Kumar, A.; Prakash, A. The role of big data and analytics in smart cities. *Int. J. Sci. Res.* **2014**, *14611*, 12–23.
39. Osman, A.M.S. A novel big data analytics framework for smart cities. *Future Gener. Comput. Syst.* **2019**, *91*, 620–633. [CrossRef]
40. Forkan, A.; Khalil, I.; Tari, Z. CoCaMAAL: A cloud-oriented context-aware middleware in ambient assisted living. *Future Gener. Comput. Syst.* **2014**, *35*, 114–127. [CrossRef]
41. Garcia-de Prado, A.; Ortiz, G.; Boubeta-Puig, J. COLLECT: COLLaborativE ConText-aware service oriented architecture for intelligent decision-making in the Internet of Things. *Expert Syst. Appl.* **2017**, *85*, 231–248. [CrossRef]
42. Bardram, J.E.; Christensen, H.B. *Supporting Pervasive Collaboration in Healthcare—An Activity-Driven Computing Infrastructure*; Technical Report; Centre for Pervasive Computing, Arhus CfPC: Aarhus, Denmark, 2004.
43. Esmaeilian, B.; Wang, B.; Lewis, K.; Duarte, F.; Ratti, C.; Behdad, S. The future of waste management in smart and sustainable cities: A review and concept paper. *Waste Manag.* **2018**, *81*, 177–195. [CrossRef]
44. Kjeldskov, J.; Howard, S.; Murphy, J.; Carroll, J.; Vetere, F.; Graham, C. Designing TramMateña context-aware mobile system supporting use of public transportation. In Proceedings of the 2003 Conference on Designing for User Experiences—DUX'03, San Francisco, CA, UAS, 6–7 June 2003; ACM Press: New York, NY, USA, 2003; pp. 1–4. [CrossRef]
45. Arbanowski, S.; Van Der Meer, S.; Steglich, S.; Popescu-Zeletin, R. The human communication space: Towards i-centric communications. *Pers. Ubiquitous Comput.* **2001**, *5*, 34–37. [CrossRef]
46. Nandyala, C.S.; Kim, H.K. From Cloud to Fog and IoT-Based Real-Time U-Healthcare Monitoring for Smart Homes and Hospitals. *Int. J. Smart Home* **2016**, *10*, 187–196. [CrossRef]
47. Newcomb, E.; Pashley, T.; Stasko, J. Mobile computing in the retail arena. In Proceedings of the Conference on Human Factors in Computing Systems—CHI'03, Lauderdale, FL, USA, 5–10 April 2003; ACM Press: New York, NY, USA, 2003; pp. 1–8. [CrossRef]
48. Aazam, M.; Zeadally, S.; Harras, K.A. Offloading in fog computing for IoT: Review, enabling technologies, and research opportunities. *Future Gener. Comput. Syst.* **2018**, *87*, 278–289. [CrossRef]
49. Rescati, M.; Matteis, M.D.; Paganoni, M.; Pau, D.; Schettini, R.; Baschirotto, A. Event-driven cooperative-based Internet-of-Things (IoT) system. In Proceedings of the 2018 International Conference on IC Design & Technology (ICICDT), Otranto, Italy, 4–6 June 2018; pp. 193–196. [CrossRef]

50. Yi, S.; Hao, Z.; Qin, Z.; Li, Q. Fog Computing: Platform and Applications. In Proceedings of the 2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb), Washington, DC, USA, 12–13 November 2015; pp. 73–78. [CrossRef]
51. Dastjerdi, A.V.; Buyya, R. Fog computing: Helping the Internet of Things realize its potential. *Computer* **2016**, *49*, 112–116. [CrossRef]
52. Atlam, H.; Walters, R.; Wills, G. Fog computing and the internet of things: A review. *Big Data Cogn. Comput.* **2018**, *2*, 10. [CrossRef]
53. Mehdipour, F.; Javadi, B.; Mahanti, A. FOG-Engine: Towards big data analytics in the fog. In Proceedings of the 2016 IEEE 14th International Conference on Dependable, Autonomic and Secure Computing, 14th International Conference on Pervasive Intelligence and Computing, 2nd Intl Conference on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), Auckland, New Zealand, 8–12 August 2016; pp. 640–646.
54. Virdis, A.; Vallati, C.; Nardini, G.; Tanganelli, G.; Stea, G.; Mingozzi, E. D2D Communications for Large-Scale Fog Platforms: Enabling Direct M2M Interactions. *IEEE Veh. Technol. Mag.* **2018**, *1*, 24–33. [CrossRef]
55. Prabavathy, S.; Sundarakantham, K.; Shalinie, S.M. Decentralized secure framework for social collaborative Internet of Things. In Proceedings of the 2017 Fourth International Conference on Signal Processing, Communication and Networking (ICSCN), Chennai, India, 16–18 March 2017; pp. 1–5. [CrossRef]
56. Buzachis, A.; Galletta, A.; Carnevale, L.; Celesti, A.; Fazio, M.; Villari, M. Towards osmotic computing: Analyzing overlay network solutions to optimize the deployment of container-based microservices in fog, edge and iot environments. In Proceedings of the 2018 IEEE 2nd International Conference on Fog and Edge Computing (ICFEC), Washington, DC, USA, 1–3 May 2018; pp. 1–10.
57. Truong, H.L.; Narendra, N.C.; Lin, K.J. Notes on ensembles of IoT, network functions and clouds for service-oriented computing and applications. *Serv. Oriented Comput. Appl.* **2018**, *12*, 1–10. [CrossRef]
58. Guerrero, C.; Lera, I.; Juiz, C. A lightweight decentralized service placement policy for performance optimization in fog computing. *J. Ambient Intell. Hum. Comput.* **2018**, *10*, 2435–2452. [CrossRef]
59. Kim, K.; Kim, H.; Kim, S.K.; Jung, J.Y. i-RM: An intelligent risk management framework for context-aware ubiquitous cold chain logistics. *Expert Syst. Appl.* **2016**, *46*, 463–473. [CrossRef]
60. Krivic, P.; Skocir, P.; Kusek, M.; Jezic, G. Microservices as Agents in Iot Systems. Available online: https://link.springer.com/chapter/10.1007/978-3-319-59394-4_3 (accessed on 2 Feruary 2021).
61. Portmann, E.; Tabacchi, M.E.; Seising, R.; Habenstein, A. (Eds.) Designing Cognitive Cities. Available online: https://www.springer.com/gp/book/9783030003166 (accessed on 2 Feruary 2021).
62. Mejtoft, T. Internet of Things and Co-creation of Value. In Proceedings of the 2011 International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing, Dalian, China, 19–22 October 2011; pp. 672–677.
63. Rahmani, R.; Kanter, T. Autonomous cooperative decision-making in massively distributed IoT via heterogenous networks. In Proceedings of the 1st International Conference on Internet of Things and Machine Learning—IML'17, Liverpool, UK, 17–18 October 2017; ACM Press: New York, NY, USA, 2017; pp. 1–5. [CrossRef]
64. Mutlag, A.A.; Ghani, M.K.A.; Arunkumar, N.; Mohammed, M.A.; Mohd, O. Enabling technologies for fog computing in healthcare IoT systems. *Future Gener. Comput. Syst.* **2019**, *90*, 62–78. [CrossRef]
65. Saqib, M.T.; Hamid, M.A. FogR: A highly reliable and intelligent computation offloading on the Internet of Things. In Proceedings of the 2016 IEEE Region 10 Conference (TENCON), Singapore, 22–25 November 2016; pp. 1039–1042. [CrossRef]
66. Misra, S.; Mukherjee, A.; Roy, A. Knowledge discovery for enabling smart Internet of Things: A survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2018**, *8*, 1–19. [CrossRef]
67. Seiger, R.; Assmann, U.; Huber, S. A Case Study for Workflow-Based Automation in the Internet of Things. In Proceedings of the 2018 IEEE International Conference on Software Architecture Companion (ICSA-C), Seattle, WA, USA, 30 April–4 May 2018; pp. 11–18. [CrossRef]
68. Hao, Z.; Novak, E.; Yi, S.; Li, Q. Challenges and software architecture for fog computing. *IEEE Internet Comput.* **2017**, *21*, 44–53. [CrossRef]
69. Bisgaard, J.J.; Heise, M.; Steffensen, C. *How Is Context and Context-Awareness Defined and Applied? A Survey of Context-Awareness*; Aalborg University: Aalborg, Denmark, 2004; pp. 31–40.
70. Fithian, R.; Iachello, G.; Moghazy, J.; Pousman, Z.; Stasko, J. The Design and Evaluation of a Mobile Location-Aware Handheld Event Planner. Available online: https://link.springer.com/chapter/10.1007/978-3-540-45233-1_12 (accessed on 2 Feruary 2021).
71. Cheverst, K.; Mitchell, K.; Davies, N. Exploring context-aware information push. *Pers. Ubiquitous Comput.* **2002**, *6*, 276–281. [CrossRef]
72. Zhang, Y.; Sheng, V.S. Fog-enabled Event Processing Based on IoT Resource Models. *IEEE Trans. Knowl. Data Eng.* **2018**, *31*, 1707–1721. [CrossRef]
73. Bruns, R.; Dunkel, J.; Masbruch, H.; Stipkovic, S. Intelligent M2M: Complex event processing for machine-to-machine communication. *Expert Syst. Appl.* **2015**, *42*, 1235–1246. [CrossRef]
74. Papageorgiou, N.; Verginadis, Y.; Apostolou, D.; Mentzas, G. Event-driven adaptive collaboration using semantically-enriched patterns. *Expert Syst. Appl.* **2011**, *38*, 15409–15424. [CrossRef]
75. Kapitsaki, G.M.; Prezerakos, G.N.; Tselikas, N.D.; Venieris, I.S. Context-aware service engineering: A survey. *J. Syst. Softw.* **2009**, *82*, 1285–1297. [CrossRef]

76. Kraemer, F.A.; Braten, A.E.; Tamkittikhun, N.; Palma, D. Fog Computing in Healthcare–A Review and Discussion. *IEEE Access* **2017**, *5*, 9206–9222. [CrossRef]
77. Hassani, A.; Medvedev, A.; Haghighi, P.D.; Ling, S.; Indrawan-Santiago, M.; Zaslavsky, A.; Jayaraman, P.P. Context-as-a-Service Platform: Exchange and Share Context in an IoT Ecosystem. In Proceedings of the 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), Athens, Greece, 19–23 March 2018; pp. 385–390.