Service discovery and selection in IoT: A survey and a taxonomy

Meriem Achir, Abdelkrim Abdelli, Lynda Mokdad, Jalel Ben-Othman

Version of Record: https://www.sciencedirect.com/science/article/pii/S1084804521003167
Manuscript_eb6e2aa8b443245f04dc5c9032f20c4c

Service discovery and selection in IoT: a survey and a taxonomy

Meriem ACHIR[1], Abdelkrim ABDELLI[2], Lynda MOKDAD[3], and Jalel Benothman[4]

[1] *LSI laboratory USTHB University* , Algiers, Algeria , machir@usthb.dz
[2] *LSI laboratory USTHB University* , Algiers, Algeria , Abdelli@lsi-usthb.dz
[3] *Univ Paris Est Creteil, LACL* F-94010 Creteil , Paris, France , lynda.mokdad@u-pec.fr
[4] *Univ Paris-Saclay, CNRS, CentraleSupélec Lab. L2S,* , Univ Sorbonne Paris Nord , jalel.benothman@l2s.centralesupelec.fr

Abstract

Recently, Internet has evolved into a new generation, called Internet of Things, thus enabling the connection between the physical and the digital worlds by creating an ubiquitous and self-organizing network. A huge number of smart objects are becoming now identifiable and addressable while being able to communicate with each other. Moreover, the integration of cloud infrastructures in the design of IoT, has moved this new trademark technology into a new dimension, enabling virtualisation and service provisioning. Billions of cloud services with different performance levels, requirements and functionalities are thus being offered in IoT, raising however the issues of their management, discovery and selection. In the literature, a considerable effort has been invested to address service discovery and selection in the context of IoT, despite the lack of standardization that meets the IoT requirements. In this paper, we propose an exhaustive taxonomy to classify service discovery approaches in the context of IoT, that we subsequently evaluate according to different aspects and criteria. Then, we discuss the gaps and advantages of each class of our taxonomy and locate the context and the requirements under which each can operate. Finally, we identify the challenges and future research directions in this domain.

Keywords: Taxonomy, service discovery, Service selection, IoT, QoS, QoE, Classification, Architecture, Object discovery.

1 Introduction

Nowadays, a growing number of people are connected to the Internet, 24 hours a day, relying on one or more smart devices including: smartphones, exercise and health monitors, e-readers, tablets, etc. In 2020, we count more connected smart devices than people around the world. Since the emergence of Internet, a significant number of technological innovations has been introduced in our daily lives. The Internet of Things is the new trademark technology that has revolutionized the cyberspace by impacting the present human modern life and shaping its future. Indeed, IoT is pushing our daily lives into a new technology era by investing a variety of applications that cover several areas, including: smart homes, smart healthcare, smart cars, smart buildings, smart transports, etc. IoT is providing the physical and software frameworks to enable the interconnection of billions of virtualized physical entities (Things), through user's smartphones, or by using ad hoc communication protocols.

Due to its fast development, IoT is now demanding more scalable and flexible resources (hardware and software) to handle the increasing number of services provided by these connected objects. More particularly, a high processing speed and a large bandwidth are required to process and analyze the real time interactions, as well as the huge amount of generated data, that are loudly soliciting the actual Internet infrastructures capacities. Therefore, to improve the resource management and provisioning in IoT systems, the Cloud Computing is more and more considered when designing the global IoT infrastructures. The CC appears to be the miracle cure to actual IoT problems that hindered so far its large deployment. Indeed, the CC is offering a large amount of resources, both in terms of data storage and the ability to bring flexible and scalable processing resources to the analysis of data. However, the traditional CC is essentially based on a centralized architecture, so a single point of failure, fault-tolerant and exposed to security risks [1]. Besides, the increased physical distance between data centers and users, escalates the transmission latency and hence the response time [2]. For this purpose, a new paradigm, called Fog/Edge Computing [2][3], has emerged recently to extend the CC to new concepts that fully support the IoT vision. Unlike CC, Fog/Edge is a distributed computing environment composed of several fog nodes which are placed at one-hop distance from the user. Fog Computing aims at pushing the intelligence, processing and storage of data closer to the edge of the network to provide computer-related services faster and near to the interconnected smart things that form a part of the IoT [4]. Such an architecture reduces the service latency, increases the context awareness, conserves the network bandwidth while improving the quality of provision and enhancing the operational efficiency. It also provides a better user experience and speeds up the real-time processing as well as storing sensitive data closer to where it is generated, while offering a better support for mobile communications.

Therefore, the advent of the IoT and its underlying technologies has led to the proliferation of connected objects enabling a wide range of ubiquitous services with different performances and functionalities. A service in the IoT environment can be any software or hardware entity that can be invoked by a user to perform specific tasks, as for instance, any data reading, sensing or measurement, etc. Moreover, services can be composed together to provide complex functionalities and improve the Quality of Service performances to meet user requirements [5][6].

IoT service discovery and selection is the process of exploring the services provided by the IoT objects to promote the optimal ones, while dealing with IoT constraints, as well as user requirements so that to improve the QoS and the QoE [7]. However, the lack of standardization in terms of architectures and communication technologies that meet completely IoT requirements, has raised the urgent need to design holistic approaches for service discovery and selection. So far, different concepts, architectures and paradigms have been considered in the design of service selection and discovery approaches already defined in the literature. Therefore, there is a real need to conduct a complete survey with a contextualization and a classification of these different solutions, in order to identify the gaps and the common features between their conceptual and implementation designs. In the literature, most of the existing surveys that address this topic are considering only limited aspects such as: context, data, social relationships, composition etc [8][9][10][11][12][13] [14][15]. To overcome this lack of studies within this topic, we put forward in this paper a much exhaustive and elaborated taxonomy that includes several aspects to identify and classify IoT service-object discovery and selection approaches. Furthermore, we discuss and compare the characteristics and the functioning of the most recent techniques defined in the literature. We respond to some research questions and highlight guidelines to overcome the challenges of IoT service discovery and selection.

The remainder of this paper is organized as follows. Relevant survey articles are presented and compared in Section II. Section III recalls some concepts and backgrounds. Section IV is devoted to present our taxonomy scheme. Sections V, VI, VII detail the different classes of our taxonomy. Section VIII discusses some research questions and compares the surveyed solutions according to different performances criteria. The last section concludes this paper and highlights the future research directions and challenges.

2 Related works

In this section, we review the most recent important studies in the literature that survey service discovery and selection approaches in the IoT context. We compare these works according to their contributions, and discuss whether they provide a general classification and a comprehensive overview of the existing techniques in terms of conceptual and implementation designs.

One of the first surveys in our study has been provided by Abdellatif *et al.* in [8]. The authors introduced a new classification that takes into account several aspects: architecture type (centralized, distributed or hierarchical), the operating mode (push/pull), the discovery interface (Search engine, Recommendation systems, Global directory), the discovery scope (remote and/or local) and the dependency to Internet protocols. However, some other important IoT aspects such as service description, discovery and selection mechanisms have not been included and discussed in this work.

In [16], Shinde and Olesen reviewed a few number of service description and discovery approaches and compared them according to four parameters: the discovery technique used, user preferences, context-awareness, and lightweightness. However, this study does not consider other aspects of service discovery and selection.

In 2018, Pattar *et al.* proposed a new classification of service discovery approaches into two main categories: data-oriented and object-oriented [9]. Data oriented solutions are then distinguished into: *Content-based*, which focuses on observational and measurement data read by sensors; and *context-based*, which refers to the data describing the context and the state of IoT objects. On the other hand, object-oriented solutions are classified into three categories: (i) location-based (geographic or relative to the IoT objects in the network); (ii) user-oriented, that takes into account the user's preferences and social relationships with objects; and finally (iii) heterogeneous, that uses several types of data representation, status description and other attributes of IoT objects.

More recently, a quite similar taxonomy to [9] is provided by khalil *et al.* in [10]. They proposed further to classify the protocols (CoAP, MQTT and UPnP) of resource discovery in IoT environment into four main classes: scope; methodology; architectures; and approaches. First, the *scope class* is divided into: local (UPnP) and remote (CoAP, MQTT). The *methodology class* is either a *publish/subscribe (MQTT)* or a *request/response* (CoAP, UPnP). Third, the *architecture class* is either a *client/server* (MQTT) or *Peer-to-peer* (CoAP) and finally the *approaches class* is divided into *centralized* (CoAP, MQTT), *distributed* (UPnP) and *semantic-based discover* (CoAP, MQTT). However, this new classification doesn't review all the protocols used in the IoT SD. Moreover, some important aspects have not been discussed, such as: Description methods, Bio-inspired paradigm, QoS, etc.

In another recent study [17], Zorgati *et al.* proposed to classify IoT SD algorithms into two main categories: those that are protocol-based (CoAP, MQTT and mDNS/DNS-SD) which provide syntactic resource discovery; and those that are semantic-based. Then, they consider six criteria to compare the different surveyed approaches, which are: description method (syntactic/semantic); architectural design (centralized and/or distributed); scope of discovery (local and/or remote); and four IoT requirements (interoperability, scalability, context-awareness and security). Although this study is more complete, it doesn't review all important and recent discovery protocols. Moreover, it lacks of details and discussion when presenting the different approaches.

Roopa *et al.* put forward a more exhaustive study in [12]. They reviewed the fundamental concepts of Social IoT paradigm, the prerequisites, the main challenges to overcome, as well as different use cases based on these concepts. Three classifications have been introduced for SIoT according to: SD mechanisms, network link selection, and trust composition. Discovery mechanisms are split into two classes: *extrinsic aspects* that includes four categories: location-based; time based; spatio-temporal based; and event-based. The *intrinsic aspects* class includes three categories: Encounter history based; mobility pattern based; and encounter history and mobility pattern based. Network link selection considers two classes: object friendship techniques and object similarity techniques. Trust composition is categorized by several reliability specification parameters. Although this survey is interesting as it considers new criteria in the proposed taxonomies, it focuses, however, mainly on social objects while neglecting other aspects.

Table 1: Related studies in the IoT SD and selection.

Survey	Year	Review type and main focus	Taxo	Description method aspect				Discovery and selection mechanisms					Architectural aspect		QoS metrics & IoT
				Syntactic-based	Semantic-based	Resource-based	Hybrid	Semantic-based	QoS-based	Protocol-based	Nature-inspired	Context-based	Architectural design	Implementation model	requirements used for evaluation
[8]	2018	IoT SD	Yes	Yes	Yes	–	–	–	–	–	–	–	Centralized, Distributed, Hierarchical	P2P overlay	–
[16]	2018	IoT SD	No	–	Yes	–	–	Yes	Yes	Yes (REST)	–	Yes	–	–	User preferences, context-awareness, lightweightness
[9]	2018	IoT Resources Discovery	Yes	–	–	–	–	Yes	Yes	–	Yes(Social)	Yes(location)	Centralized, Distributed	P2P overlay	–
[10]	2020	Resource discovery techniques in IoT	Yes	–	–	–	–	Yes	–	Yes (UPnP, MQTT, CoAP)	Yes(Social)	Yes (location)	Centralized, distributed	P2P overlay	Heterogeneity, latency, energy consumption, mobility
[17]	2019	IoT SD	Yes	Yes	Yes	–	–	Yes	–	Yes (DNS, MQTT, CoAP)	–	–	Centralized, distributed, hybrid	–	Interoperability, Scalability, Security and context-awareness
[12]	2019	Survey about Social-based approaches	Yes	–	–	–	–	–	–	–	Yes (Social)	Yes (Location, Event)	Centralized, Distributed	P2P overlay	Trust management, network navigability
[11]	2019	Multi-criteria classification of IoT SD	Yes	Yes (XML, Records)	Yes	Yes (REST, CoAP)	–	Yes	Yes (Security, Hybrid)	Yes (DNS, DPWS)	Yes (Bio)	Yes	Centralized, distributed	Layered, P2P overlay, Agents, Fog infrastructure	Context-awareness, the improved IoT requirements by each solution
[18]	2019	SLR SSA in IoT	Yes	–	–	–	–	–	–	–	–	–	Centralized, decentralized, hybrid	–	Accuracy, response time, execution time, scalability, reliability, processing time, execution cost, energy, latency, security and load balancing
[19]	2020	SLR of the IoT SD techniques	Yes	Yes	Yes	–	–	Yes	Yes (Energy, Hybrid)	–	–	Yes	Centralized, distributed, hierarchical	–	heterogeneity, fault-tolerance, energy consumption, delay, availability, security, mobility, overhead, scalability
[20]	2020	QoS-aware SSA for IoT	Yes	–	–	–	–	–	Yes	–	–	–	–	–	the improved IoT requirements by each solution
[13]	2020	SD in IoT	Yes	Yes	Yes	–	–	Yes	Yes (Security, Energy, Network, hybrid)	Yes (DPWS, DNS, MQTT, XMPP, REST, CoAP)	Yes (Bio, Social)	Yes (Location, Event, User requirements, User feedback)	Centralized, Distributed, Hybrid	–	the improved IoT requirements by each solution
Our work	–	Multi-criteria taxonomy of IoT SD and selection	Yes	Yes (XML, JSON, Records)	Yes (Ontology, RDF)	Yes (REST, CoAP)	Yes	Yes	Yes (Security, Energy, Network, Hybrid)	Yes (DPWS, DNS, MQTT, XMPP, REST, CoAP)	Yes (Bio, Social, physical)	Yes (Location, Event, User requirements, User feedback, Hybrid)	Centralized, Distributed, Hierarchical, Hybrid	Layered, P2P overlay, Agents, Broker, Federated, Cloud, Fog, SOA, Hybrid	interoperability, scalability, mobility, dynamicity, autonomy, fault-tolerance, availability, energy consumption, load balancing, reliability, accuracy, precision, context-awareness and user preferences, user feedback, trust, security, delay, overhead and network performance

In [11], Aziez *et al.* proposed a multi-criteria classification of IoT services by considering five aspects: level of complexity, service capability, physical location of discovering service, architectural aspect, and service representation. They provide a taxonomy to classify the surveyed approaches according to four perspectives: description method; discovery mechanism; architecture; and context-awareness (supported or non supported). Although, this classification is rich it remains that many used concepts and paradigms are confused which does not help to understand the general philosophy behind the proposed taxonomy.

Li *et al.* [18] have systematically reviewed the architectures used in IoT service selection algorithms and classified them into: centralized, decentralized, and hybrid. Besides, the authors provided an evaluation of the existing techniques using some QoS metrics, such as: accuracy, response time, reliability, execution time, scalability, reliability, processing time, execution cost, energy, latency, security and load balancing. The proposed classification focuses only on the architectural aspect of the solutions and doesn't address other features such as: service description, discovery mechanism, etc.

Pourghebleh *et al.* presented in [19] a systematic review of the existing IoT SD techniques which are classified into four categories: context-aware; energy-aware; QoS-aware; and semantic-aware. Then, they discussed and compared some selected approaches according to different metrics such as: heterogeneity, fault-tolerance, energy consumption, delay, availability, security, etc. and other criteria, like: the adopted architecture, the discovery method, the service description, the discovery scope, etc. However, this classification does not include all the aspects and many interesting approaches have not been included in this study.

In [20], Abosaif and Hamza classified some service selection algorithms proposed in the literature in three main levels: algorithm design; algorithm implementation; and algorithm performance evaluation. The algorithm design level includes three categories: SSA based on the process time phase (static/dynamic); SSA based on behavioral workflow management (orchestration/Choregraphy); and SSA based on optimization objective type (single/multi/many objectives). Moreover, the algorithm implementation level consists of two categories: QoS based on IoT architecture (Sensor/network/application layers); and the algorithm type (heuristic, meta/hyper/no heuristic). Finally, the performance evaluation level determines the most commonly used methods and datasets to assess the performances of the SSAs.

Finally, Achir *et al.* proposed in [13] to classify the SD and selection algorithms according to the main paradigm used in the discovery process: Semantic based, bio-inspired based, protocol based, context based and QoS based techniques. This taxonomy focuses mainly on SSA and does not propose a classification of other operations in the life cycle of IoT services, such as: description methods and architectural design types.

Table 1 summarizes the contributions of our current study while comparing those of the most important and recent surveys within the same topic.

3 Concepts and Background

This section is dedicated to recall some usual concepts and paradigms to ease the understanding of the topic and the proposed classification.

3.1 List of acronyms :

To improve the readability of the paper, we give in Table 2 the descriptions of all the abbreviations and acronyms used in this paper.

Table 2: List of acronyms description.

The concept	Acronym
Ant Colony Algorithm	ACA
Ant Colony Optimization	ACO
ASAWoO Disruption-Tolerant RESTful Support	ADTRS
Artificial Fish	AF
Artificial Fish Swarm Algorithm	AFSA
Adaptive Fibonacci-based Tuning	AFT
Analytic Hierarchy Process	AHP
Artificial Neural Network	ANN
Artificial Potential Fields	APFs
Application Programming Interface	API
Bluetooth Low Energy	BLE
BLE-Advertiser	BLE-A
Back Propagation	BP
Border Router	BR
Bi-objective Shortest Path Optimization	BSPO
Cloud Computing	CC
Content-Centric Networking	CCN
Continuous Hidden Markov Mode	CHMM
CoRE link format	CLF
Constrained-Node Networks	CNNs
Constrained Application Protocol	CoAP
Constrained RESTful Environments	CoRE
CoRE Resource Directory	CoRE RD
Client Request Handler	CRH
Comma Separated Values	CSV
Dynamic Bayesian Networks	DBN
Distributed Geographic Table	DGT
Distributed Hash Table	DHT
Distributed Location Service	DLS
Decision Makers	DM
Domain Name System	DNS

Continued on next page

Table 2: List of acronyms description.

The concept	Acronym
Domain Name System for Service Discovery	DNS-SD
Devices Profile for Web Services	DPWS
Discovery Responses Suppression	DRS
Distributed Service Framework	DSF
Datagram Transport Layer Security	DTLS
Delay or Disruption-Tolerant Networking	DTN
Extensible and Adaptable Discovery Protocol	EADP
Event-Driven SOA	EDSOA
Evolutionary Game theory- Evaporation-based Water Cycle Algorithm	EG-ERWCA
Electronic Product Code	EPC
Efficient XML Interchange	EXI
Genetics Algorithm	GA
Historical Records	HR
Hypertext Transfer Protocol	HTTP
Information-Centric Networking	ICN
Internet Engineering Task Force	IETF
Integer Linear Programming	ILP
Internet Protocol	IP
Input/Output	IO
Internet of Things	IoT
Internet of Vehicles	IoV
JavaScript Object Noration	JSON
JSON for Linked Data	JSON-LD
Low-power and Lossy IoT Networks	LLN
Linear Reward Inaction	LRI
Labeled Transition System	LTS
Machine to Machine	M2M
Multi-Criteria Decision Making Methods	MCDM
Multicast Domain Name System	mDNS
Mobile Edge Computing	MEC
Many-Objective Evolutionary Algorithm Decomposition	MOEA/D
Multi-objective Optimisation Problems	MOP
Message Queuing Telemetry Transport	MQTT
Neural Collaborative Filtering	NCF
Named Data Networking	NDN
Nature Inspired Computing	NIC
Normalized Google Distance	NGD
National Institute of Standards and Technology	NIST
Natural Language Process	NLP
Object Management Group	OMG
Object Name Service	ONS
Order Statistics Local Method	OSLOM
Ontology Web Language	OWL
Web Ontology Language for Web Services	OWL-S
Peer-to-Peer	P2P
Probing and Advertisement Suppression	PAS
Physical Markup Language	PML
Particle Swarm Optimization	PSO
Pointer record	PTR record
Precondition/Effect	PE
Quality of Context	QoC
Quality of Experience	QoE
Quality of Service	QoS
RESTful API Modeling Language	RAML
Resource Constrained Devices	RCD
Resource Description Framework	RDF
RDF Schema	RDFS
REpresentational State Transfer	REST
Radio-Frequency Identification	RFID
Reference Holders	RH
Reinforcement Learning	RL
Routing Protocol for LLNs	RPL
Response Threshold Model	RTM
Round Trip Time	RTT
Service Agent	SA
Semantic Actuator Network ontology	SAN ontology
Service Discovery	SD
Social Internet of Things	SIoT
Serial Line Internet Protocol	SLIP
Systematic literature review	SLR
Social-like Semantic-aware	SLSA
Semantic Matching Engine	SME
Service Oriented Architecture	SOA
Simple Object Access Protocol	SOAP
SPARQL Protocol and RDF Query Language	SPARQL
Service Provision Node	SPN
Social Profile Search	SPS
Service Selection Algorithms	SSA
Semantic Sensor Network ontology	SSN ontology
Selection Stage Optimization	SSO
Transmission Control Protocol	TCP
Thing Ecosystem Description	TED
Thing Description	TD
Technique for Order of Preference by Similarity to Ideal Solution	TOPSIS

Continued on next page

Table 2: List of acronyms description.

The concept	Acronym
Unmanned Aerial Vehicle	UAV
Universal Description Discovery and Integration	UDDI
User Datagram Protocol	UDP
Universal Plug and Play	UPnP
Uniform Resource Identifier	URI
World Wide Web Consortium	W3C
Web service	WS
Web Services Description Language	WSDL
Web Service Semantics	WSDL-S
Web Service Modeling Ontology	WSMO
Web of Things	WoT
Extensible Markup Language	XML
Extensible Messaging and Presence Protocol	XMPP

3.2 Useful concept definitions:

In this section, we define the main concepts and technologies used within the context of service discovery and selection, while discussing their applicability and the challenges needed to address in order to meet the IoT requirements.

- **Smart things:** Smart things represent the physical devices embedded with sensors, actuators, RFID tags, etc. and are capable of connecting to Internet. They can publish and share their data and services on the network and can also interact with the other smart objects anywhere and anytime.
- **RFID:** *Radio frequency identification* is a wireless communication technology that incorporates a radio frequency to uniquely identify an object, animal, or person without requiring a battery in the tags. The RFID system can be divided into two main components: radio signal transponder (tag) and tag reader [21]. The tag consists of two components: a chip to store the unique identity of the object and an antenna to allow the chip to communicate with the tag reader using radio waves. Then, the activated tag sends back the data to the antenna. These data trigger the programmable logical controller to perform a particular action. RFID tags can be active (powered by battery), passive (don't need battery), or semi-passive/active tags (which use board power when needed)[21].
- **IoT:** IoT is a world-wide dynamic network which interconnects the physical and the virtual worlds via Internet for communication and data transmission [15]. It's composed of billions of heterogeneous smart things which are uniquely addressable and based on standard communication protocols. IoT is lacking of a flexible layered architecture as no reference model has been normalized yet. The already existing drafts are based on a three-layer architecture [22] consisting of the Application, Network, and Perception Layers. However, the authors in [21] have illustrated a five-layer architecture, including:
  - **Object layer (or perception layer):** This layer includes the IoT physical devices dedicated to collect and process the sensed information. The latter is then digitized and transferred to the Object Abstraction Layer.
  - **Object abstraction layer:** It is in charge of transferring the data produced by the Objects layer to the Service Management layer through various communication technologies, such as: RFID, 3G, Wi-Fi, Bluetooth, etc.
  - **Service management layer (or middleware/pairing layer):** The role of this layer is to pair a service with its requester based on addresses and names. Also, it processes the received data, makes decisions, and delivers the required services over the network [22].
  - **Application layer:** It provides the services requested by the customers.
  - **Business (management) layer:** It manages the overall IoT system activities and its related elements. Besides, it compares the output of each layer with the expected one to enhance service performances and maintain users' privacy [22]. Moreover, this layer is responsible of building a business model, graphs, etc. based on the received data from the Application layer. Thus, it becomes possible to support decision-making processes based on big data analysis.

  Figure.1 presents the most common IoT architecture models, as defined in [22].

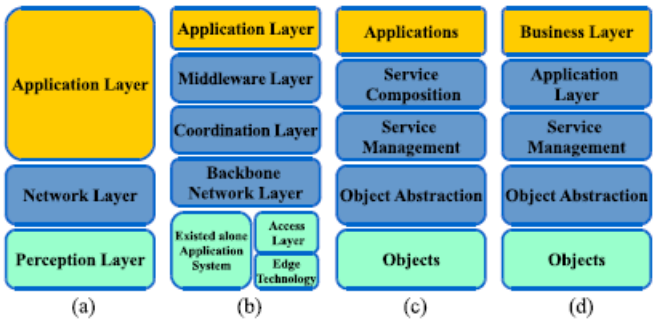

Figure 1: The most common IoT architecture model [21]: (a) Three-layer. (b) Middleware-based. (c) SOA-based. (d) Five-layer.

- **WoT:** The WoT is defined as the application of Web technologies to the IoT in order to access information and services of physical smart things. Therefore, the latter are built according to the REST architecture and accessed using the HTTP protocol via RESTful API.
- **Cloud Computing:** IoT systems frequently involve the use of CC platforms. The most widely agreed proposed definition for this technology was introduced by the NIST [23]: *"A model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction"*. Thus, CC is a particular form of the management of configurable and scalable infrastructures (hardware and software computing resources) based on remote servers defining the "Data Centers" that are centrally deployed on a global scale. It offers a range of on-demand services by usage tranche, for instance: processing power, storage, development environment, bandwidth, etc. over a wide area network, often Internet. The CC has partially resolved most of the IoT issues because it provides a flexible and a robust environment with unlimited virtual resources to the widely-distributed IoT objects which are resource-constrained (i.e., limited storage and processing capabilities, performance, availability, security and reliability issues, etc.).
- **Fog Computing:** The Fog computing is an extension of the classical CC model. It was developed by Cisco to reduce pressure on cloud servers and provide computing, storage, control, and networking capabilities closer to the edge of a network and thus to the IoT devices [24]. As depicted in Figure. 2.a, it consists of multiple fog nodes (e.g. gateways, routers, switches, etc.) distributed over a network. These nodes process the data received from IoT objects locally rather than migrating it to the cloud servers far away from the users, thereby decreasing the latency and the response times [1]. In overall, the fog computing eases location and context-awareness, mobility support, scalability, interoperability, and real-time interactions, while it reduces network traffic and power consumption, etc. The Open Fog Consortium defined the fog computing as [3]: *"A horizontal, system-level architecture that distributes computing, storage, control and networking functions closer to the users along a cloud-to-thing continuum."*. This distributed paradigm acts as an intermediate layer between CC and IoT devices as shown in Figure 2.a. Smart things generate data and then send it to the nearest fog nodes where it will be processed. Thus, the cloud layer is henceforth in charge to manage the massive data requiring extra computing power and storage capabilities, such as: big data management and big data mining [1][24]. The emergence of this new paradigm makes it possible to overcome some of the CC limitations in order to better support IoT application challenges.

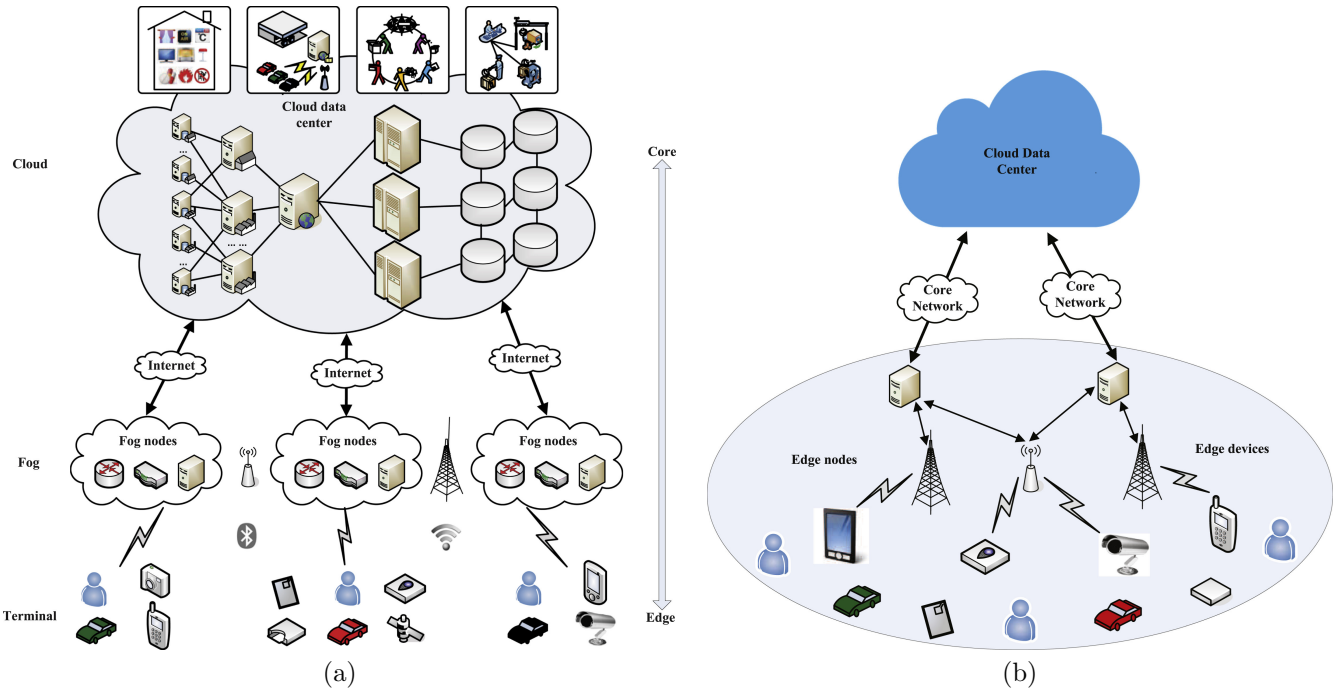

Figure 2: Fog and Edge computing architectures [25].

- **Edge Computing:** Most of the prior literature has treated Fog and Edge computing as synonymous and has used these two words interchangeably. As shown in Figure. 2.b, the edge computing is a paradigm that extends the CC capabilities to the edge nodes (i.e., smart sensors, smart phones, smart vehicles, edge servers, etc.), allowing computation and storage to be performed at the nearest level (within the Base Station of cellular mobile networks) [25]. Therefore, edge devices are called to connect to the cloud data-centers through the core network only when advanced processing is needed. The table 3 compares Edge and Fog paradigms according to different criteria.
- **RDF:** *Resource Description Framework* is a family of W3C specifications designed as a standard model for semantically describing web resources, using a variety of syntax notations and data serialization formats. The resource is represented as a triple (subject, predicate and object), where the subject defines the semantic entity of interest, the object refers to the value of this entity, and the predicate represents the relation between the entity and its value. Moreover, RDF Schema is an

Table 3: Comparison between Fog and Edge paradigms [25].

	Edge Computing	Fog Computing
Architecture	Hierarchical, decentralized, distributed	Hierarchical, decentralized, distributed
Proximity to end devices	Located in end devices	Near (Single network hop or few network hops)
Latency	Low	Low
Bandwidth costs	Low	Low
Resource	More limited	Limited
Computation and storage capabilities	More limited	Limited
Mobility	Supported	Supported
Scalability	High	High
Service	Virtualization	Virtualization

extension of RDF that provides a structured data-modeling vocabulary for RDF data by defining generalization relations and constraints of RDF triples [26].

- **SPARQL**: SPARQL [27] is a language used to express queries across diverse data sources, whether the data is stored natively as RDF or viewed as RDF via a middleware. SPARQL is able to query required and optional graph patterns along with their conjunctions and disjunctions while providing the results as sets or RDF graphs.
- **EPCglobal Network:** EPCglobal is the original organization responsible for the development of EPC, the management of EPC and RFID technology and standards. EPCglobal Network is an IoT standard which can be divided into five components: Electronic Product Code, ID system, EPC Middleware, Discovery Services, and EPC Information Services [21]. The EPC is a unique identification number which is stored on an RFID tag and is used basically in the supply chain management to identify items. The ID system links the EPC identities to a database using an EPC reader through the middleware. The discovery service is a mechanism of EPCglobal to find the required data by the tags using the ONS.
- **Ontologies:** For the semantic Web, ontologies define the concepts and relationships used to describe a particular domain and to capture the semantics of a domain by deploying knowledge representation primitives to understand the relationships between the different concepts. According to Studer *et al.* [28], an ontology is a formal and explicit specification of a shared conceptualization which consists of defining standard vocabularies that will be shared and reused between objects and supporting reasoning mechanisms for inferring intelligent decisions. Also, it facilitates the discovery, integration, manipulation and configuration of IoT resources and their data. According to [29], almost all of the existing approaches using ontologies for representing the IoT domain knowledge take the advantage of reusing the existing ones instead of building them from scratch because building an ontology from scratch is costly and needs great human efforts. Also, the mapping between ontologies sharing components through ontology reuse is easier. Moreover, Lonsdale *et al.* [30] judge that reusing an existing ontology guarantees the quality of the new one because the reused concepts have already been tested and validated. The most reused ontologies are SSN and generic ontologies, such as time and location ontologies, DOLCE, etc [29].
- **WSDL:** WSDL is an XML-based language recommended by the W3C and used for describing web services syntactically. It defines web services as a collection of communication endpoints able to exchange messages in an XML documents, where their locations, operations and properties (i.e., inputs, outputs, condition, etc.) are specified. Thus, a client can locate a web service and invoke any of its publicly available function. Some works have proposed to use the WSDL for describing IoT services. However, the use of this technology is not suitable for the IoT context due to its heavy-weight nature and reliance on XML. In addition, it focuses mainly on modeling the functional aspects, including: the description of the service (inputs, outputs, preconditions, and effects); the service operations (how the service internally operates); and the service invocations (how to call the service). But, some aspects are not directly targeted such as the non-functional requirements: QoS & QoE (e.g., availability, data quality, observation rate). Both of functional and non-functional features are important for IoT object/service discovery and selection. Also, even for traditional web services, the WSDL description is considered insufficient to answer accurately user queries. Therefore, some efforts have been made to extend this concept to support non-functional requirements [31]. Moreover, another extension, called WSDL-S, was proposed to support the semantic information of web services. In WSDL-S, the semantic models are maintained outside of the WSDL documents and are referenced via WSDL extensibility elements [32]. Finally, we note that no standardized version has been presented in the literature to support completely IoT services.
- **QoS:** The QoS is the ability to achieve the desired level of performance regarding a set of parameters (e.g., cost, bandwidth, latency, delay, packet loss ratio, jitter, throughput, etc.). So, the QoS is more focused on non-functional requirements of IoT. QoS attributes are identified and used in the service selection to refine the discovery and select the most suitable services among a set of candidates [33]. Some works [33] [34] [35] have listed and studied the significant QoS metrics in each layer of the IoT architecture. In the same respects, Al Nuaimi *et al.* identified QoS attributes in IoT environment and grouped them according to their similarity into three clusters [34]: "Resources" (allocation, stakeholders and scarcity); "Performances" (power consumption, reliability, trust-management and security...); and "Network" which measures attributes related to the network infrastructure, such as delay-tolerance, packet delivery, message transmission. In [35], Dongre *et al.* investigated the QoS parameters to apply for service composition and selection by considering nine QoS parameters, which are: execution time, response time, availability, reliability, throughput, cost, price, reputation, and latency.
- **QoE:** The QoE can be defined as a method of assessing users satisfaction based on subjective (user opinion, intentions and perceptions) and objective (influenced by technical parameters) psychological measurements for the use of a given service [36]. So, it represents the overall acceptance of an application or a service from users' opinion and can be influenced by expectations and user context [36]. It can be expressed through both qualitative and quantitative parameters. It is noteworthy that so

far no standardization of frameworks for measuring the QoE has been conducted in the IoT context. However, some relevant works, such as [36][37] attempt to model and manage the QoE measurement in the IoT environment.

- **Service Discovery:** SD is the process of finding and discovering the services according to their functional capabilities which are matching the functional properties specified in the user request. In the IoT environment, things are represented as a set of IoT services which are different from traditional ones. They are deployed in resource-constrained devices that are characterized by their mobility and their limited capabilities in terms of energy, computing, communication and storage. Therefore, traditional SD solutions are not suitable for the discovery of IoT services because they demand heavy communication schemes incorporating sleep cycles to save energy, and use bulky formats. Hence, several approaches have been proposed to discover IoT services by considering IoT constraints [38][39][40][41][42][43], but none of them meet all these requirements due to the lack of standards and a unified service/object representation. Indeed, unlike web services, IoT services represent objects' features and thus their description must be lightweight and enriched with some additional parameters, including: location, energy level, QoS support and context management, etc [40][41][44][45][46].
- **Service Selection:** Service Selection is the process of selecting the most appropriate services among the candidate ones based on non-functional properties, such as: price, reliability, security, response time, etc. which are known as the QoS parameters. The quality of an IoT service may change over time because it operates in a constrained environment. Therefore, the service selection method must be adaptable according to user preferences. Thus, a service can be replaced during its life cycle by an alternative providing the same functionalities and with a similar or an improved QoS. This process calls the definition of a unified description method that supports the IoT requirements and the non-functional properties of IoT services, although a great effort has been already made to tackle issue [47][48][49][50][51][52].
- **Service Composition:** Service Composition is the process of dynamically discovering and combining simple and atomic services to form more complex ones to respond to user requirements which cannot be met by considering a simple service. The traditional service composition aims to solve a programmatic issue and focuses on the demonstration of the feasibility and the logicality, while the IoT service composition aims to provide user with the sensory information. Due to the dynamic evolution of user needs and context, the composition must be adaptive and make a real-time update, which improves the user interaction with the environment [53]. Also, it can introduce the non-functional parameters to choose and compose services. However, because of the intermittent connectivity in IoT, services are unstable (i.e., not available all the time); they degrade, vanish and possibly re-appear. This highly increases the probability of errors such as failures and service changes in IoT service composition [14]. Besides, IoT services are mainly deployed in resource-constrained devices with limited capabilities, unlike Internet services which are deployed in computers with rich resources and powerful functions. Moreover, this process must manage the security aspects to resist to malicious attacks. To address these challenges, a number of IoT service composition approaches have been proposed in the literature [54][55][56][57][35].

4 The proposed taxonomy for service discovery and selection approaches

Considering the dynamic nature of the IoT environment, SD techniques that meet QoS, QoC and QoE are becoming fundamental to automatically select and locate the most appropriate devices and services for a given application. Therefore, the efficiency of each IoT system depends mainly on the services provided to the users. According to the architecture design of the SD system, the service delivery passes through many operations, including: service description, service discovery, service selection and service composition. In this section, we put forward a new taxonomy that classifies SD approaches in the context of IoT environment. This classification is depicted in Figure. 3. The first level of our taxonomy classifies the SD approaches into three categories according to the main aspects involved in the SD system:

- The first category deals with 'Description methods'; it identifies and categorizes all the description approaches used in SD.
- The second aspect, called 'Discovery and selection algorithms', identifies and classifies all the SD and selection algorithms defined in the context of IoT environment.
- Finally, the third category called 'Architecture', inventories all the types of IoT architectures adopted in the surveyed papers.

Each category is then refined to classify the different surveyed solutions according to either their conceptual design, or to operational and practical aspects.

- First, the description method class is refined into five main subclasses according to the essence of the information considered in the modeling technique used in the discovery and selection process, we have: Syntactic-based (XML; Records and JSON-based); Semantic-based (based on ontology or RDF); Resource-based (based on CoAP or REST); as well as a hybrid subclass which regroups solutions that combine different description methods.
- As concerns the 'discovery and selection algorithms' category, we further introduce five main sub-classes by recognizing the primary mechanism or paradigm used to characterize the discovery and the selection process, namely: Semantic-based; Nature-inspired; Protocol-based; Context-based; QoS-based. Then, the approaches of each of the previous sub-classes are further characterized to refine their classification taking into account additional criteria.
  - The class of "nature-inspired approaches" identifies all the service selection algorithms that are based on heuristics inspired from natural world. Such techniques can be distinguished by considering the three computing paradigms, which are: Biological; social; and physical.

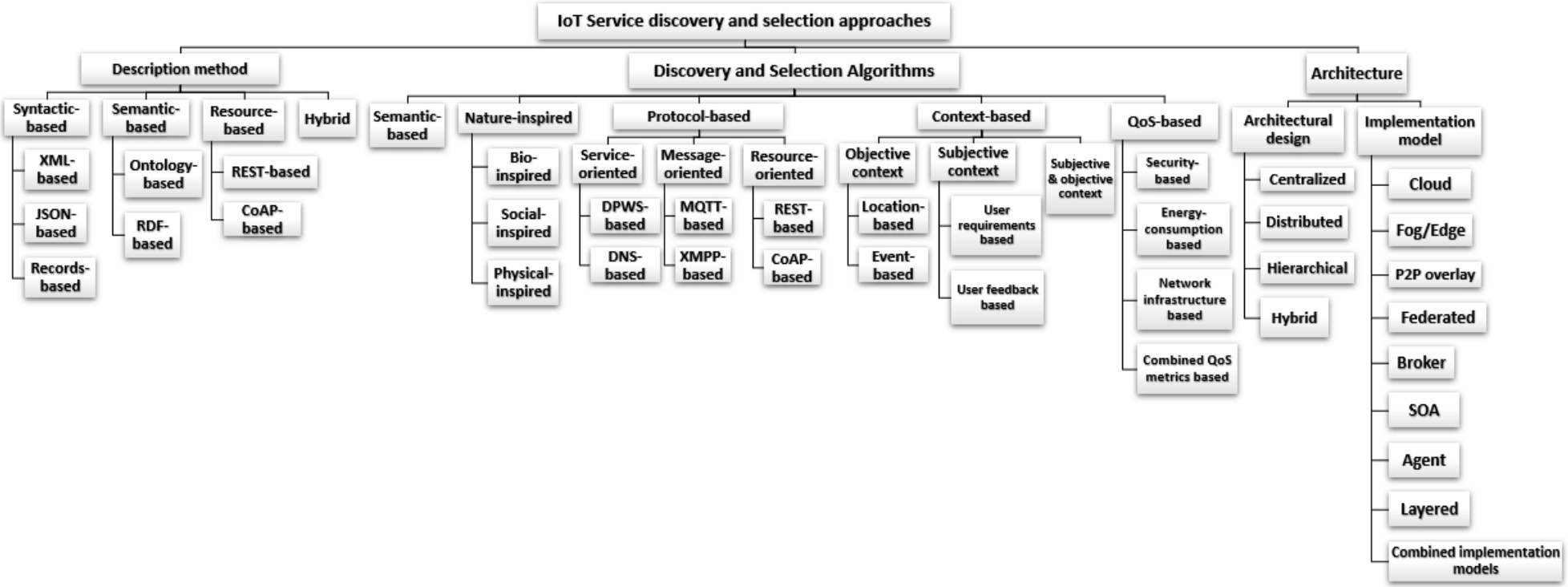

Figure 3: The proposed taxonomy of IoT SD approaches.

  - Regarding the context-based class, the latter includes all the techniques that integrate context information (QoC and QoE parameters) when performing the selection process. Such solutions are further characterized by categorizing the context information into objective, subjective aspects, or both. The 'objective aspects' consider either the 'location-based' or the 'event-based', whereas the 'subjective aspects' include 'user requirements' and 'user feedback'.
  - As far as the protocol-based approaches are concerned, they are further described by specifying the type of protocols considered in the SD process, thus defining three groups of techniques: 'Service-oriented' (DPWS, DNS); 'message-oriented' (MQTT, XMPP); and finally 'resource-oriented' (REST, CoAP).
  - In the QoS based class, we inventory all the approaches that aim to improve the QoS of the network. These solutions are further distinguished by recognizing the QoS aspects to be optimized in the selection process, including four clusters: Security-based; energy based; network-infrastructure based, and finally by combining different QoS metrics.

- In the category "architecture", we recognize two principal structuring and organization aspects in the design and the implementation of the SD system. The first aspect considers the architectural design of the solution in the way its different components are communicating and collaborating namely, whether it is centralised; distributed; hierarchical or hybrid. The second aspect focuses on the implementation model; whether the solution is based on: Cloud; Fog/Edge; P2P overlay; Broker; SOA; Agents; Layered; Federated or hybrid.

In the sequel, we describe and discuss the features of each class of our taxonomy while highlighting the most interesting works introduced in the literature for each identified category.

5 Architecture

An architecture of a SD system is a representation that defines the structure, the behavior, the perspectives and the views of the system, in order to meet all the technical and operational requirements, while optimizing the common quality attributes like performance and security. It shows how a system is decomposed, how the processes interact, or the different ways in which system components are distributed across the network. As aforementioned, two aspects are considered: the *Architectural design*, which focuses on the global conceptual representation of the system, especially, its level of decentralization; and the *implementation model* that represents the way this system is implemented and putt into effect. This aspect describes also the elements of the system, how they fit and work together to fulfill its requirements. The majority of the approaches surveyed in this paper has adopted an architecture type, however, some solutions, such as in: [58, 59, 60] did not mention any specific architectural design, and others like in: [61, 62, 52, 63, 64, 65] didn't adopt any particular implementation model of the SD solutions they proposed.

5.1 Architectural design

From the architectural design perspective, an IoT SD and selection system can operate by following either a centralised, a distributed, a hierarchical or a hybrid architecture.

5.1.1 Centralized solutions:

Generally, the centralized systems have a central component (e.g, middleware, server, etc) that is responsible for conducting the discovery and selection process. These central points are paramount in the system as they represent a common entry for users' queries and service providers. Besides, they contain the registers which store the service descriptions and they handle the retrieval requests to discover registered objects and services. In the literature, several SD solutions have adopted a centralized design, out of these, we can quote the works presented in [66], [39], [50] [44], [67]. Low execution time and cost, good stability, high availability are the main advantages of centralized methods. Although this architectural type is easy to set up, affordable to maintain and can be developed quickly, it is prone to failures since it is based on a single central owner. Moreover, such solutions are more subject to a higher security and privacy risks as they provide a longer access time to data. In addition, they cannot scale well as the number of registered services and parallel accesses to the registry are limited. Thus, bottlenecks can appear when the traffic spikes and, as a result, the response time may increase dramatically, especially, when the system is subject to a high load.

5.1.2 Distributed solutions:

To overcome the limitations of the centralized architecture and to bring more flexibility and resilience into IoT environments, the distributed solutions have been put forward. They rely on connections among service providers or between service providers and end-users to solve the SD and selection problems, without referring to any central entity. They achieve thereby to distribute the workload on a number of interconnected cooperating nodes, which are distributed geographically. In this model, the discovery algorithms run locally on these nodes which store indexes, data items of the IoT resources, etc. Each local registry or algorithm can directly be connected to the next register or algorithm. The distributed architecture enables the resource sharing and improves the performances of the systems by making them more efficient, fault-tolerant, secure, extremely scalable and robust. However, such a model is more difficult to deploy and induces higher maintenance costs. Besides, due to the absence of a centric manager that processes user requests and responses, it becomes difficult to identify which node has failed and which one has served the request. Thus, each node must be pinged to check its availability. In the literature, many SD solutions have considered a distributed design, among them we can quote the works presented in [68], [48], [43], [69], etc.

5.1.3 Hierarchical solutions:

In the hierarchical architecture, the involved components in the SD system are organized in a hierarchical structure and follow a master-slave (or parent-child) relation. The low-tier element (child) covers its own local area and tries to resolve service queries within its local network. When it fails, it forwards the query to its parent component which federates many low-tier elements and collects aggregated services description by pulling its following components. The hierarchical techniques were proposed to offer a scalable solution and to minimize the traffic congestion and overhead. Also, they allow exchageability, extensibility and separation of concerns. Among SD approaches using this hierarchical design, we can cite the solution presented in [38], where the IoT environment is structured as a tree hierarchy of smart spaces (e.g., country, region, city, streets, buildings, rooms). Each smart space is managed by a semantic gateway, which is a software component maintaining information about the IoT services in its scope and processing discovery requests. At the lowest level nodes, the actual IoT service descriptions are registered in the semantic gateway while at upper levels only aggregated information is maintained. In another solution, Nguyen *et al.* [70] have proposed to organize a gateway in every area according to a three-tiered hierarchical MEC network topology to reduce computing overhead at the centralized controller. All MECs are divided into smaller groups/clusters based on their location and are managed by a group/cluster-head node (Regional Cloud). However, the high latency induced in hierarchical communications can be problematic in real-time systems. In addition, the failure of a parent node requires downtime and restart (data loss can happen in such cases) [71] [72]. Also, this architectural design doesn't support automated fail-over systems since a child needs to be manually promoted to a parent, if the original one fails.

5.1.4 Hybrid solutions:

Solutions that adopted a hybrid architecture mix two or more architectural designs, or switches from one to the other. In this way, they achieve to combine the desired features of each to increase the efficiency and the performances of the resulting scheme. In the literature, few solutions have considered hybrid architectures. For example, in [73], authors proposed to combine the advantageous characteristics of the centralized and the distributed CoAP models to design a single model for an efficient SD in the context of a constrained IoT network. First, the client request goes through the refinery node for analyzing its type. If the request contains the address of the directory node, it is then forwarded to the Border Router by following the distributed CoAP SD approach. Otherwise, the request is forwarded to the Client Request Handler by considering the centralized CoAP SD approach. In another solution, Dahbi *et al.* proposed a hierarchical distributed architecture for EPCglobal SD [74]. It refers to a set of structured P2P overlay networks, each of which is attached to one sub-region and involves only nodes representing companies that are physically located in this sub-region. Also, it is based on a multi-level tree structure where each node at a given level, except the root, points to one parent node at the immediate upper level, representing the sub-region in which the child node is physically located. Each overlay network in this architecture elects at least one of its nodes as its "entry node" which plays the role of a gateway. This allows a node, located in a given sub-region, to route lookup queries whose the key belongs to a company located in a different sub-region.

The main advantages of using hybrid solutions are: a low processing and response time, a good stability, a high scalability, and good performances etc. However, they may suffer from a high complexity and some management and maintanability issues.

5.2 Implementation model

SD solutions can be further distinguished by the way their architectures are implemented. Several implementation models have been considered such as: cloud, fog, broker, layered, agents, P2P overlay and so forth. The relation between the implementation model and the architectural design is not always a one to one relation, but rather meshy, in the sens that many implementations models can be related to different architectural designs and vice versa. For instance, cloud and broker based models are generally associated with a centralized architecture, unlike models based on Fog/Edge, P2P overlay or federated which rather favor a distributed or sometimes a hierarchical architecture. However, agent-based, layered and SOA models are open to any type of architectural design.

5.2.1 Cloud-based model:

Several works opted for a cloud infrastructure to guarantee scalability and to ease the deployment with a low-cost management. This technology offers the potential to use large amounts of resources, both in terms of data storage as well as in the ability to bring flexible and scalable processing resources for data analysis. Among the works that have adopted the cloud based model, we can quote the approach presented in [75], where a SD system was designed as a client-server model to register and discover services and their privacy properties. The web server (IoT Service Store), used in this solution, runs on a virtual private cloud and communicates with users in a secured manner through the use of the HTTPS protocol. In another solution [76], the authors propose to host the ONS and the IoT service databases in the cloud. Similarly in [77], the cloud is used to process and store an enormous amount of data to the high-end servers and data centers. Each IoT gateway is then connected to a cloud data center by a wired network.

However, since cloud data centres are geographically centralized, they often fail to deal with storage and processing demands of billions of geo-distributed IoT devices/sensors. Hence, congested network, high latency in service delivery and poor QoS are experienced. In addition, virtualizing IoT nodes in the cloud can be problematic when it comes to deal with low-latency applications. Besides, applications with security concerns are often requiring to process the data inside the company and not remotely.

5.2.2 Fog/Edge-based model:

To address the limitations of the cloud model, the Fog/Edge Computing has been considered. This model consists of multiple nodes, deployed over a large geographical area, to extend the cloud infrastructure and services to the edge of the network by being close

to the user, as only the necessary part of the data is sent to the cloud for further analysis [78][79][70]. This technology has a great number of advantages, such as: real-time processing, rapid scalability, context-awareness, location-awareness and resource pooling, etc. In addition, by reducing the number of network hops, it better deals with latency-sensitive applications that require nodes to be in the vicinity to meet their delay requirements. Therefore, to minimize the delays in the service access layer, a judicious placement of the fog nodes is needed. For example, in [77], authors have taken gateways as candidates for fog node deployment to collect data from smart sensors and have made them smarter with fog capabilities for pre-processing and decision-making features. In the proposed solution, the fog tier is responsible for running all the latency-sensitive applications as well as real-time analysis.

However, this implementation model faces some major challenges, which are: the discovery of fog nodes, the data caching and offloading, the service orchestration, etc. Moreover, the deployment strategy of fog nodes remains a serious concern as there exists neither a common Fog/Edge computing architecture nor a generally accepted concept to how and where deploy fog nodes so that they support efficiently the real-time IoT service execution. Generally, Edge devices (i.e., switch, router, gateway, mobile phones, smart car, etc.) are the candidates for deployment of fog nodes but the deployment differs according to the application.

5.2.3 P2P overlay network:

Peer-to-peer overlay is a computer network built on the top of an existing network, usually the Internet, in a distributed manner. This model enables participating peers (devices that are part of P2P networks), to find the other peers not by their IP addresses but by using the specific logical identifiers known to all these peers. The devices do not have specific roles and can be service requester and provider at the same time. Also, they are equal and there are no privileged ones and there exists no primary administrator device in the center of the network. When a client wants to search for a service, it sends a query message to all its directly connected neighbour peers (except the one which delivered this query). These nodes forward the query to their neighbours and so on. If a node receives a query and finds a match in its directory, it responds with a query-hit message containing enough information for the retrieval of the data matching the corresponding query. P2P networks have attracted the attention of many researchers because they provide some desirable features, including: the support of scalability, reliability and mobility, robustness and efficiency, high availability, no central control, etc. They are able to handle joining and leaving peers (at anytime) without requiring any configuration on a central server, thereby coping with the inherent dynamicity of IoT scenarios. In addition, they are self-organizing and resilient to failure of individual peers, so, no administrative overhead. However, security and privacy in huge IoT networks are difficult to guarantee. Moreover, it's not easy to ensure the integrity of packet sent between nodes, since one router node could easily replace one packet with another, and the receiver would never be able to make the difference. A wide range of SD solutions make use of P2P overlays, as for example, the work presented in [80], where authors proposed a decentralized discovery system that uses a structured P2P network based on SkipNet-overlay. This scheme is scalable and provides a controlled data placement. Besides, it guarantees locality based routing by organizing data primarily by string names. This topology is composed of IoT nodes (both service providers and service consumers), which are organized into several doubly-linked lists. Each of these lists is sorted using the nodes' string identifiers and every pair of nodes that are adjacent in these lists forms a bidirectional network connection. In addition, every node in the discovery overlay provides an information repository, where services' descriptions are stored. More recently, Kalkan *et al.* proposed a decentralized framework for SD on top of a structured P2P network based on a DHT allowing each peer to be responsible for a specific part of the content in the network [81]. There are different protocols that construct structured P2P networks (e.g., CAN, Chord, Pastry, Kademlia, etc.), but in this work, authors did not specify any type of these protocols since they only used the lookup function which is provided by any type of DHT. The lookup operation is provided by sending a request to the closest node known by the requester which then forwards it to the next closest node it identifies and so on until it reaches the destination. If the latter does not exist in the network, then the last node receiving the query notifies the requester of the transmission failure. In another work, Cirani *et al.* proposed a SD architecture that relies on two P2P overlays: the Distributed Location Service and the Distributed Geographic Table [82]. The DLS provides a name resolution service to retrieve all the information needed to access a resource (of any kind) identified by a URI, while the DGT builds a distributed geographical knowledge, based on the location of nodes, which can be used to retrieve a list of resources matching geographic criteria.

5.2.4 Federated solutions:

Some solutions organize services in a collection of autonomous directories that cooperate by forming a federation with its own rules, to design what is called "a federated architecture", like the approaches presented in [40] and [45].

The latter rely on a federation of repositories cooperating with each other according to a predefined standard used to perform data and SD tasks. When receiving a discovery request, the Federation Search Handler performs a federation search by simultaneously searching for resource descriptions and forwarding queries to the repositories available at the federation. Therefore, federated search architectures can be used when we need to simultaneously search multiple sources and quickly obtain relevant results using a single search query on a single intuitive search interface. They are suitable for the IoT SD as they can provide several advantages, for instance: high scalability, fault-tolerance, independent control of repositories, and data distribution, etc. Also, they reduce the response time and guarantee a unified access to diverse content sources. However, they suffer from a number of issues, such as: access problem, interface issue, results duplication, management and coordination of the repositories, etc. Besides, a federated search on its own can lead to information overload, making it afterwards time-consuming and difficult for users to locate the most relevant services.

5.2.5 Broker-based model:

To achieve a better decoupling of clients and providers as well as hiding communication-related concerns, a broker component is introduced as an additional layer, to which providers register themselves and make their services available to clients through uniform

interfaces. Therefore, clients are accessing these services by sending requests or subscribing to particular ones. Besides, users don't need to know anything about the implementation details of an object or its physical location. Actually, the broker component is responsible for the coordination and the communication, such as: locating the appropriate provider, forwarding the request and transmitting results and exceptions back to the client. Also, this entity is flexible as it allows dynamic changes, additions, deletions, and the relocation of objects as well as it reduces the complexity of developing distributed applications as they are made transparent to the developer. Broker-based models can be used in systems that consist of multiple remote objects which interact synchronously or asynchronously, as well as for heterogeneous environment. The use of this model has some benefits such as: changeability and extensibility of components, portability and reusability of the broker, interoperability between the different broker systems, location transparency and so on. However, using such a model has some drawbacks: a restricted efficiency (applications using brokers are usually slower and have a high latency), a less fault-tolerance, as well as more modules to deploy, test, debug and maintain.

Many works, adopting this model, were proposed in the literature, as for instance, the work presented in [66], wherein a broker-based SD mechanism with flexible query/service matching capabilities, is proposed. This solution is composed of four basic entities (Clients, Service Providers, Discovery Brokers and Semantic Matching Engines), which interact with each other through the use of well defined interfaces. The Discovery Broker is responsible for holding the information about the available services and for matching incoming queries with these services (by interacting with the SME over an available transport protocol (e.g., UDP, TCP, ICN)). The main features of the Discovery Broker are: (i) Service (un)registering by listening to requests from service providers, and accordingly adds/removes services to/from the local table of available services and forwards part of the received information to the SME to update the services database located at its level; (ii) Service matching; (iii) communicating with service providers and clients using the NDN protocol. In another work, Ahmed *et al.* proposed a secure broker-based SD technique that aims to securely deliver services to consumer taking into consideration services' context and QoS as well as provider trustworthiness [67]. In this solution, a broker is installed on the devices' gateway as a software to gain advantages of edge computing in offloading IoT devices from implementing heavyweight security algorithms. Service consumer submits an encrypted version of his query to a centralized broker, which manages a trust value to the connected objects besides to monitoring devices' QoS. Based on the services' queried QoS, context, and trust records, the broker matches the user query to the most appropriate provider. To guarantee secure communications, the broker distributes session key to the communicating entities during the service delivery. Additional broker-based solutions were proposed in [83][84][85] by using the MQTT protocol. The broker is responsible here for handling and organizing all communications between the various devices. It manages topics and message exchange and notifies the MQTT clients (subscribers) about the publication of messages within specific topics.

5.2.6 SOA-based model:

The adoption of SOA in the IoT enables communication between smart devices by accessing their exposed services. It thereby simplifies the complexity of the discovery process and allows objects to interoperate in a uniform and general manner, independently of their communication standards. This paradigm provides an automated SD and enables the generation of new services obtained by service composition to achieve complex tasks. Many SD solutions have adopted this model, as for example in [38, 39, 86, 44, 56, 87]. There exesit several advantages to use SOA model, as for instance: portability, reusability, interoperability, loose coupling, service autonomy and agility, service interoperability, service discoverability, service composability, etc. However, using it in the IoT context may lead to some issues, such as: the lack of a standard format for IoT service description and of the context-awareness aspect, complex data visualisation and analysis, compatibility, mobility, security and fault management, etc.

5.2.7 Agent model:

Agents are pieces of software running on a device, responsible for the physical data acquisition and have the capability to autonomously learn and make recommendations regarding a particular action, for example: locating and accessing necessary information, resolving inconsistencies in the retrieved information, filtering away irrelevant and unwanted information, integrating information from heterogeneous information sources and so on. The use of software agents in IoT SD and selection ensures a high level of autonomy, intelligence and dynamicity, and minimizes human intervention. Besides, they offer a flexibility in design and implementation and are reliable because they can temporary store/cache monitored logs when network connection is lost, and can also decrease the network overhead. However, agent-based solutions are hard to deploy because they require to install and maintain agents on each target node. Such a procedure is time and resources consuming and impacts negatively the performances of resource-constrained devices. Some SD solutions defined in the literature have adopted this model, as for instance, Krivic *et al.* who proposed an agent-based model for IoT SD and management, which is illustrated in Figure. 4 [88]. In this solution, whenever the M2M/IoT server receives user requests to start or stop certain services, the Server Agent executes the task allocation algorithm ensuring thereby a maximum service lifetime by considering which services can be executed on given M2M/IoT devices according to their current battery level, and how much energy the execution of certain services would consume. Then, the server Agent notifies the Gateway Agent, located at M2M/IoT Gateway, of the tasks that need to start their execution on selected devices. Thereafter, the Gateway Agent forwards the instructions to the corresponding Device Agents, which initiate the execution of the allocated tasks and forward the results to the M2M/IoT Gateway. It's worth to notice that agents don't handle the data, but only the control messages. In the solution defined in [89], the authors considered software agents that reside in each SPN and can interact with the infrastructure of the IoT device associated with the respective SPN. These agents provide SPN functionalities and handle user requests using a service registry that stores service descriptions, and a request table to record information about the processed requests. More works using this model can be identified in the literature, as for instance: [88, 89, 90, 91, 49, 92, 93].

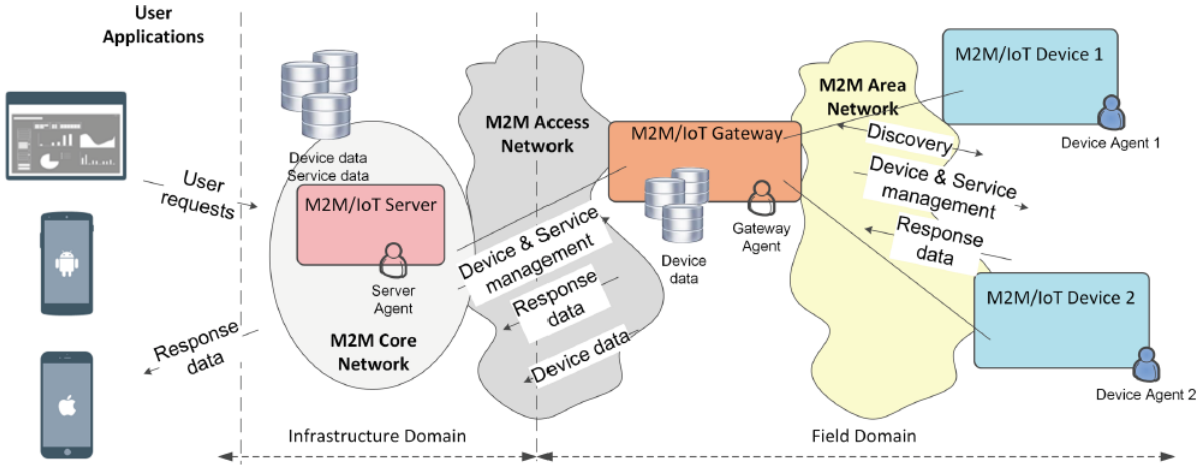

Figure 4: Agent-based model for IoT SD and management [88].

5.2.8 Layered model:

The layered model allows a system to be organized into a set of hierarchical layers by limiting the behavior of components so that each component can't "see" beyond the immediate layer with which it interacts. A layer is a logical structuring of components that are deployed or organized in the same place. Each layer provides a set of services to the layer "above". This model type is mainly used for complex applications like those of the IoT for multiple advantages, as for instance: it provides modularity and clear interfaces, simplicity in the design, reusability, flexibility, scalability and supports portability, etc. However, the performances of the system may get slower as more and more layers are added, thus incurring execution and processing overheads. Many SD solutions have adopted this model, as for example, in: [41], [40], [47], [94], [95], [96], etc.

In [41], a centralized SD algorithm was designed into four-layers structure, including: the interactive interface layer, the parsing annotation layer, the service matching layer, and the data semantic layer. The first layer is an information interaction interface between the SD model and the requester, In the parsing annotation layer, the request service description document is received from the interactive interface layer, and parsed using OWL-S_{IOT} to extract some information, as: Service category, Input/Output, Precondition/Effect and QoE in OWL-Siot description. The service matching layer is divided into the category matching module, the IO matching module, the PE matching module and the QoE matching module. Finally, the data semantic layer stores knowledge and service documents, including an ontology knowledge base and a UDDI semantic services base. In [96], a framework based on CoAP is proposed . It's composed of four layers: Sensor Abstraction Layer, Data Access Layer, Resource Registration and Discovery Layer, and Social-ambient Overlay layer. The first one represents the physical devices present in the real world that are abstracted and exposed to communicate through standard interfaces. The second layer is composed of three modules: (i) a repository, where the resource descriptions are stored; (ii) a triple store for storing RDF; and (iii) a reasoner to match the desired requirement with the most covering available services by inferring logical consequences from a set of axioms based on QoS, location and application entity. In the resource registration and discovery layer, the CoAP CoRE link format is enriched with semantic tags provided by the semantic matchmaker module. In the last layer, a social-aware overlay network is designed to perform semantic-based group composition by the cluster head. The latter is an IoT node selected to perform a faster and a scalable device discovery.

5.2.9 Combined implementation models:

Many works have combined the aforementioned implementation models to provide efficient SD solutions. Among them we can quote the work presented in [97], in which an edge-centric distributed architecture is presented to provide resource discovery and access services to IoT applications. In this solution, IoT gateways are deployed in servers located close to IoT devices and away from the centralized Cloud (Fog nodes), and are federated through a P2P overlay implemented by a DHT. The latter is exploited to share the information on IoT resources available across multiple domains that are expected to inter-work with each other in a distributed manner. The combination of these two technologies (i.e., Fog computing and P2P) has guaranteed the scalability, an easy deployment, a low-cost management, a low latency and a location-awareness. More recently, Ferdousi *et al.* proposed in [73] a cloud-based middleware for CoAP SD (see Figure. 5). This model acts like the existing CoAP distributed approach where it is required, however, for complex requests (where the previous model can be expensive), it calls the middleware to make the discovery mechanism more reliable. The proposed solution is dynamically responsive for a continuously changing IoT network since it introduces a context-based analysis to generate a virtual cloud-based registry dynamically, that enables the scalability, the mobility, the interoperability and resolves proficiently bottleneck problems.

In another work, Khansari *et al.* proposed a cloud-based platform to better manage IoT service selection and composition in fog computing environment while dealing with some network constraints and QoS parameters, such as: bandwidth usage, latency and distributed resource utilization [79]. As depicted in Figure. 6, each group of sensors is connected to a nearby gateway (fog node), which manages the services, controls their availability, while being in charge of running particular workflows composed of

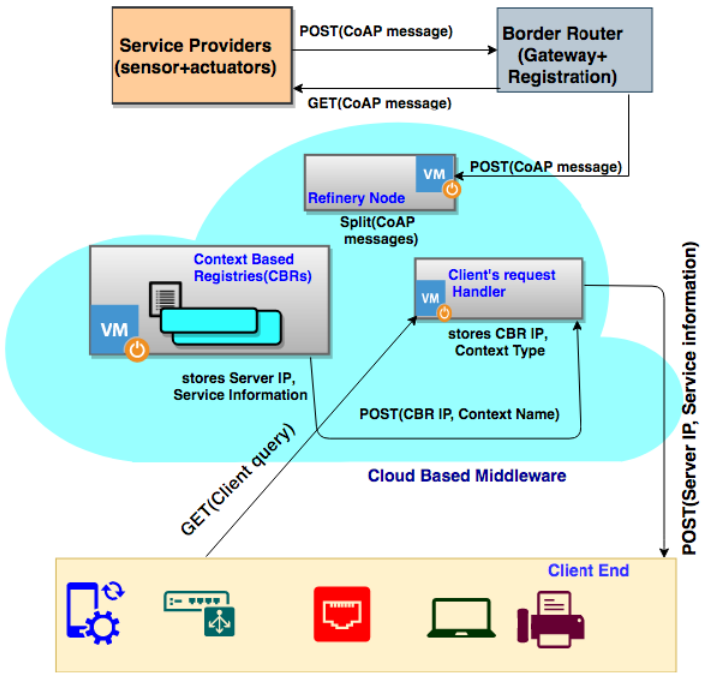

Figure 5: The cloud-based middleware architecture proposed in [73].

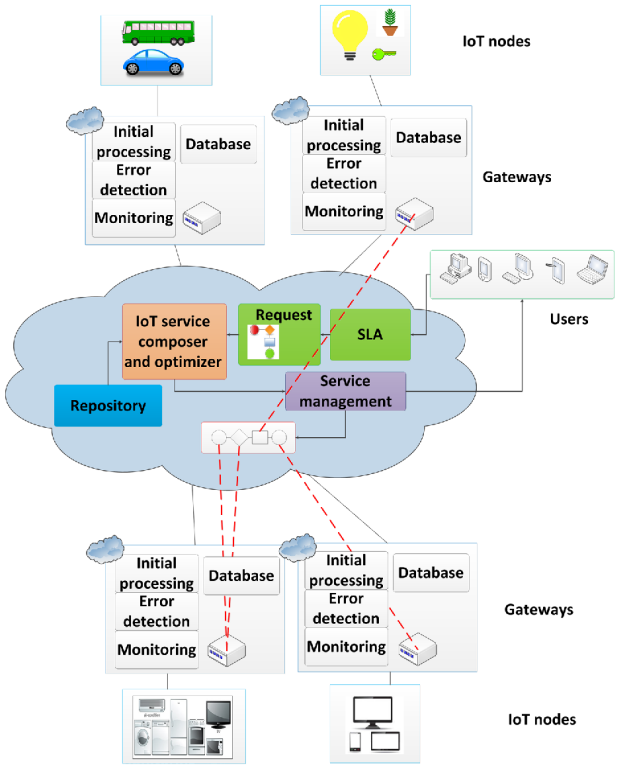

Figure 6: The IoT service management architecture proposed in [79].

these services. This gateway sends the services' information that are currently available in its zone and their properties to the cloud. The data coming from different gateways is aggregated and stored into a repository. In this solution, gateways can process data and make decisions. Therefore, errors and failures can be detected more quickly as each gateway is in charge of the nearby nodes, thereby easing the management of geographically distributed devices and their related services.

6 Service description methods

They represent the definition of what an object or a service provide and how they are accessed and used. The description captures the functional and non-functional characteristics of the object/service in a processable format and in a comprehensive, unambiguous manner, thereby easing its interpretation and automation to become human readable, and fast to describe by users. In the literature, different methods have been defined to describe objects and their services in IoT context. The majority of these approaches has extended already existing solutions to cope with IoT environment. However, among the surveyed approaches few of them did not adopt any specific description method [98], [99], [50], [58], [88]. To the best of our knowledge, five main sub-classes of description methods can be identified that we distinguish according to the language or the framework used in the description process.

6.1 Syntactic-based description methods

Generally, these methods are implemented through a list of key-values attributes. So, the provider and the user must agree on the keywords to use. The discovery mechanism based on this category needs to obtain an exact match between users' requirements and the service description. Although these techniques are easy to understand and to implement, they are not suitable for a heterogeneous environment like IoT. Besides, using keywords doesn't recover accurately all the services answering the query or those that are close to. These methods can be distinguished further according to the language used in the syntactic description. Hence, three subcategories of approaches can be recognized, that are: XML-based, JSON-based and records-based methods.

6.1.1 XML-based approaches:

XML was designed to be human-readable and self-descriptive. It focuses mainly on simplicity, generality, portability and usability across the Internet. However, the main drawback of XML is its parsing complexity and verboseness. Also, it takes up a lot of space to represent data and increases traffic overhead. Moreover, XML can induce heavy system loads in a large scale because XML documents need to be loaded into memory before being processed. Therefore, using XML in the IoT context is not really suitable. Several SD approaches adopted XML as a description tool like in [47], [67], [100], etc. In some other works, XML has been extended to enrich the description of IoT services. For example, the works in [39] and [101] adopted the WSDL language to describe IoT

services while [87] and [102] used DPWS metadata to extend WSDL descriptions to enable a seamless integration of device-provided services. In [76], the service and device information are stored and retrieved in PML format which is a markup language based on XML allowing the description of information about a product using predefined tags.

6.1.2 JSON-based approaches:

JavaScript Object Notation is used to store information in an organized, lightweight and easy-to-access manner. It's more compact and lightweight than XML, and is easy and fast for machines to parse and generate in an automatic way.

JSON is preferred for IoT applications since it can self-describe and is more programmatic. Moreover, it considerably reduces network traffic compared to XML, and is suitable to represent attribute-value pairs. However, the major drawback of JSON is its lack of security. Indeed, using JSON with untrusted services is risky because it increases hacking possibilities. Therefore, it's very important to be aware of the threats the system can undergo when using JSON. Add to that, the simplicity of the data format introduces an additional complexity effort in the implementation because JSON' simple types rarely match the data types often used in IoT programming [103]. Moreover, IoT devices are resource constrained with limited memory capacities whereas JSON is not a space-efficient encoding since all the data is expressed as ASCII strings and every label field must be repeated in its entirety for each occurrence. Among the SD approaches that have considered JSON for service description, we can cite the work presented in [84], which is based on CoRE RD, but its functionalities are based on MQTT protocol instead of CoAP. So, the CoRE Link Format is replaced by a new file format, called Resource Collection, which contains a list of all the components in the network.

6.1.3 Records-based approaches:

Few SD solutions have considered the use of records to specify IoT services features. For instance, the authors in [104] describe the services using four DNS records to enable their discovery. Each application running on a device has to register the service with its name, its IP address (provided by Contiki), and its port (if it wants to be found in the network by other devices). These DNS records can be extended by including a context-based model, as presented in the work [64], to describe and discover services based on their context. The authors in [105], proposed to use a table that contains the status information of the resource capabilities required to instantiate a service. More recently, the approach presented in [70] proposes to use a database of four tables to store service information: (i) the *registered service store* table (having the format "Service ID - Service IP - Port") to store the services registered by the providers; (ii) the *forwarding information base* table (which has the format "Service ID - Next - Port") to store neighboring MECs (in Next) in which an edge computing service is running; (iii) the *pending interest* table in the format "Service ID - Requester" to store requester and service pairs that are waiting for the results of SD, and finally (iv) the *online service store* table to store service information, such as: container ID, local IP address, and the local port of the container running on that node. In the solution presented in [58], a *concrete service* is described by its functional properties (inputs, outputs and actions) and its non-functional properties defined by the QoS attributes and the energy profile. The latter is described by the energy cost that reflects the overall power consumption during the execution of the service, and the autonomy of the device hosting the service.

The use of records allows an easy information retrieval and sharing. Besides, with their stable structure, queries can be developed and reused repeatedly. However, this description model is not suitable to describe IoT services because of the dynamic features of the latter. Although records offer the possibility to represent QoS and context attributes as key-value pairs, this information can be used only after the service resolution. this implies human intervention, which limits the intelligence of the system to make decisions. In addition, these methods do not provide any security capabilities.

6.2 Semantic-based description methods

Unlike the syntactic-based approaches, these methods aim at providing an explicit, an easy and a comprehensible description to represent IoT objects, their services and their data by using semantic web technologies. Semantic methods can be defined as [106]: "an extension of the current web in which information is given a well-defined meaning, while enabling computers and people to work in cooperation". Many models have been developed to add semantics to describe web services (such as OWL-S, WSMO, RDF) so that to support automation of SD, service composition, service invocation, knowledge extraction as well as data integration and interoperability. These models efficiently manage the service and data heterogeneity while they ease data analysis and reasoning. Semantic approaches can be further distinguished into two categories according to the conceptual paradigm used in the description. We identify, on a hand, the RDF-based approaches, and on the other hand, the solutions based on using ontology.

6.2.1 RDF-based approaches:

The RDF technology allows unifying data from different sources as well as correlating and linking the semantic described resources together to form large datasets given in the form of directed graphs. However, such a representation structure produces heavy and redundant RDF documents with a limited expressiveness compared to OWL which has a more enriched vocabulary and supports ontological reasoning. The RDF specification which has been widely used in the context of web service description, has been also adopted in few works to describe IoT devices and their services [107][108][109]. For example, in [68], the Thing Description model, which relays on RDF as an underlying data model, was introduced to semantically describe IoT devices and to ease their integration within an interoperable ecosystem.

6.2.2 Ontology-based approaches:

OWL is an ontology language standardized by the W3C for the semantic web and has been developed to provide vocabulary extension to RDF. It's easier and more expressive compared to the RDFS language. This technology allows the representation of concepts and their relationship with logical reasoning and information systems in an appropriate manner. Furthermore, it is the most widespread and used language for ontology specification. Especially OWL-S which is based on description logic to specify services semantically. Although ontologies allow the representation and the annotation of complex real-world entities as well as their web representations, including: properties, actions, and events, there exist no well defined, standardized and accepted ontologies for all the IoT applications. Furthermore, it lacks interoperability since various ontologies across different domains have to be integrated [29]. Several SD solutions have adopted OWL or one of its different extensions. Among them, the work presented in [40] wherein the authors combine SAN and SSN ontologies to semantically describe resources (i.e., sensors and actuators). The basic GeoVocabulary is considered to define concepts such as latitude, longitude and altitude of a geographic location. Thereby, it becomes possible to relate resources with the geographic location of devices by introducing a new property called "hasGeographicalLocation" to the SAN ontology. Also, they added the OWL-S ontology to this information model to describe services exposed by these resources. This description method has been extended in [45] by adding a QoC criterion to qualify the context information like accuracy, up-to-dateness, and which can be calculated in different ways according to the context within which it is enforced. In another work [41], the authors considered an extension of OWL-S, called OWL-S$_{\text{IoT}}$, which includes further context and QoE ontologies, to enrich service profile description. In the same respect, the authors in [44] specified services using an extension of OWL-S, called OWL-SE [110], to target four aspects of the service profile: inputs, outputs, preconditions and results.

6.3 Resource-based description methods:

These techniques rely on the concept of resource while adopting the REST style to describe the resource features. According to REST principles, resources are the key abstraction of information and can be referenced as a thing, stored on a computer and identified by using unique self-descriptive URIs. These resources are linked to each other and can be accessed via a common uniform interface. This description improves evolutivity and agility as it is easily extensible by adding new representations and new URIs. Moreover, resources are thus interoperable, reusable, indexable, and cacheable (within a specified expiration time). Within this class, we distinguish two types of approaches according to the description language they use: The REST-based and the CoAP-based.

6.3.1 REST-based approaches:

REST is actually the core of the Web and uses URIs for encapsulating and identifying resources on the Web in a convenient and efficient way. The representations of REST resources are exchanged amongst HTTP protocol which has four main operations: GET (to retrieve the resource representation), PUT (to update the state of an existing resource or to create a resource by providing its identifier), POST (to create a new resource), and DELETE (to remove a resource). Thus, the interaction with smart things can almost entirely happen from the browser and applications can be built upon them using well-known Web languages and technologies. The simplicity of this subclass of approaches has made it widely adopted, especially for what it brings to APIs and how it simplifies combination with existing Web infrastructures. REST aims to focus on creating loosely-coupled and interoperable services on the Web so that they can be easily reused. Besides, it ensures scalability, indexing and searching. However, it doesn't support non-functional requirements of IoT services and is heavyweight compared to CoAP, since it uses HTTP as an application layer protocol. The REST description language has been considered in many SD solutions. For example, the author in [95] integrates RESTful services within its SD framework to expose service functionalities to the consumers. In [54], the heterogeneous IoT devices are encapsulated into RESTful Web services, providing uniform interfaces for IoT service invocation, thereby allowing devices to collaborate with each other to form a comprehensive composite business process. In [111], the authors used RAML, which is a standard API definition language, for describing RESTful APIs to make resource and service exposition more simple and practical. More works can be identified in the literature, as for instance, [112, 113, 87, 54, 95, 111].

6.3.2 CoAP-based approaches:

In CoAP, IoT resources are registered in a centralized directory, called CoRE Resource Directory, which implements a set of REST interfaces: for servers, to discover RD and to register their sets of resources; and for clients, to discover these resources. The registration interface allows a CoAP server to register its resources by providing a corresponding description using the CoRE Link Format. This description contains: an identifier (i.e., URI); a resource type; and optionally, a list of client-defined key-value attributes which is useful to characterize the functionalities provided by each resource. CoAP is based on REST but is more lightweight since it's not based on the HTTP protocol. Besides, it has a low overhead and a simple packets structure. In addition, it allows the definition of context and QoS attributes as key-value pairs. However, it supports only static registration and syntactic queries, which is highly inadequate to provide automatic and intelligent resource discovery in IoT. Among the surveyed works, many have adopted CoAP to expose and implement IoT services on resource-constrained devices, as for instance, [114, 115, 73, 97, 82].

6.4 Hybrid description methods:

Many SD solutions have considered to combine the aforementioned description methods to enrich service description. For example, to describe IoT services and their geolocation, Calcina-Ccori *et al.* define JSON-LD, a method for encoding linked data using JSON [116]. The service descriptions are semantically annotated by using the Hydra vocabulary, before being translated into RDF as triples and stored into a knowledge base. Moreover, in [54], the authors consider the encapsulation of IoT devices using

RESTful Web services. The latter are described by an XML configuration files and can only be accessed through their published uniform interfaces. A similar work is proposed by Wang *et al.* that consider a semantic service description model based on OWL-S and RESTful services [112]. This model follows a "profile-model-grounding" design pattern where: the "profile" defines the non-functional service aspects (e.g, service name, category, QoS and location, etc.); the "model" defines the functional aspects (e.g, the operations allowed on the service); and the "grounding" describes service interaction elements and captures the RESTful resources. In another work, Guinard *et al.* designed and implemented two methods for representing devices and their services: DPWS-based method and REST-based method (for devices that don't support DPWS but only HTTP) [87]. In other respects, the authors in [46] extended CoAP with lightweight semantic-rich information by defining appropriate CoRE link format attributes. This extension which has been integrated in the resource directory makes it possible to describe both IoT resources and user request requirements. Similarly in [96], the CoAP protocol was enriched with semantic capabilities to describe devices. To this aim, the following attributes are added to the CoRE Link format: *ep* (It is the name of the physical device); *ae* (It refers to the application entity as described by the "*ro*"); *ro* (the reference ontology which defines the URIs used to describe the resource); *loc* (represents the devices' location in the format (longitude, latitude)); *rt* (denotes the resource type which can define the observation, events and the output generated by the service); and finally *qos* (identifies the QoS of the service specified in "*rt*").

7 Discovery and selection algorithms

In this section, we present and discuss the more interesting algorithms introduced in the literature to find, locate and select objects and their services by considering their functional and non-functional capabilities and while dealing with user requirements as well as IoT constraints. In our review, we have identified five categories of approaches according to the main paradigm or the type of algorithm considered in the SD and selection process. A review of the most important works in each class is given hereafter.

7.1 Semantic-based approaches

Semantic Web technologies have been widely used to tackle some challenging issues of IoT environment. These techniques rely mainly on using ontologies and semantic relationships to estimate the correlation degree among services and objects. Hence, data and service heterogeneity are efficiently managed resulting in a more relevant SD process. In the literature, many interesting approaches within this class can be quoted. Some approaches have considered semantic similarity measurement to elaborate communities of services.

For example, Ben Fredj *et al.* proposed a semantic-oriented IoT SD system based on a hierarchy of semantic gateways to optimize the SD in a dynamic context [38]. Within a gateway, an incremental clustering mechanism was defined to create groups of similar services. This clustering is optimized over time to minimize the discovery cost. In the same context, Chirila *et al.* [39] presented a broker-based architecture enhanced with a semantic clustering engine together with semantic similarity metrics to improve SD and recommendation. This selection approach is based on applying SOA directly on devices by considering both functional and non-functional requirements with a semantic similarity-based clustering and filtering framework to reduce the search space. To select the appropriate service from a cluster, the authors evaluate and compare similarity coefficients in the identified cluster with a given similarity threshold. A similar work is presented in [117], Zhao *et al.* put forward a multidimensional semantic-based model to describe and assess the similarity between IoT services. For this effect, a density-peaks-based clustering algorithm is introduced to gather similar services according to a similarity measurement.

Other semantic SD algorithms rely on using ontologies. For example, the authors in [40] developed a federated discovery service, called ForwarDS-IoT, aiming at providing resources discovery based on multiple attributes, range queries and synchronous/asynchronous operations. When receiving a retrieval request, ForwarDS-IoT performs a federation search by simultaneously searching for resource descriptions stored in the available repositories according to an ontology-based information model. This work was extended to another system, called QoDisco, in which QoC-related information was added to the resource descriptions in the repositories [45]. More recently, Cimmino *et al.* [68] extended the TD model by providing WoT-mappings ontology that translates the data from heterogeneous IoT devices that form the interoperable ecosystem. To this aim, they provided a SPARQL-based mechanism to transparently discover and access IoT devices. The different TDs are then analyzed and their related data are fetched to be translated into RDF using the WoT-Mappings. The evaluation of the approach shows that this solution provides a complete and correct answers without affecting the response time while scaling linearly in large ecosystems.

In other respects, some of the surveyed solutions have proposed to combine different semantic models and mechanisms to improve SD. For example, Quevedo *et al.* developed in [66], a flexible ICN-based SD mechanism on top of the NDN architecture, which leverages the use of the semantic matching mechanism. This solution makes it possible to understand and process context information for turning raw data into useful knowledge and thus ensuring the interoperability among heterogeneous devices and applications. In the work presented in [86], the authors proposed a hybrid algorithm for service matchmaking by combining syntactic and ontology similarity methods while considering QoS attributes to improve the accuracy and the precision rate. More recently [41], Jia *et al.* put forward a centralized four-layer model for IoT SD by considering a multi-stage semantic service matching. For this purpose, they introduced a hybrid service matching degree measurement by calculating the concept logic and the semantic similarity for each layer, separately.

7.2 Nature-inspired approaches

In the last decade, a sustainable effort has been devoted to adopt nature-inspired computing paradigms, such as physical, biological and social. Such techniques enable the development of decentralized service oriented systems, by taking advantage of their inherent ability to operate without a central control as well as their efficient management of resources and their self-organization. In our taxonomy, we identified three categories of nature-inspired approaches, which are: bio-inspired; social-inspired; and physical-inspired.

7.2.1 Bio-inspired approaches

Bio-inspired algorithms aim at solving a given problem using computer models based on biology concepts, which take a set of rules to build thereof unsupervised training deep learning tools, as for instance: Artificial Neural Network, Ant Colony Algorithm, Genetics Algorithm, etc. In the literature, several SD techniques have considered the use of bio-inspired models: Among the first works within this class, we can quote the solution published in [47] by Bao *et al.* who developed an ubiquitous services platform for IoT which adopts an XML-based service description method. This solution provides a Back Propagation service selection algorithm based on ANN for adaptive momentum factor to find the most appropriate service quickly and accurately. In 2016, Rapti *et al.* presented in [89] a decentralized IoT SD model based on RTM which is inspired from a decentralised insect colonies taking advantage of the coordination mechanisms of biological societies. In this work, RTM is mapped into the SD and selection problem to provide a model based on the local view of each SPNs in the network, modeled as agents. More recently, Azizou *et al.* [118] proposed a distributed SD model based on ACA to find the shortest path to the most appropriate service with a low number of hops required. Another interesting and recent work has been presented by Jia *et al.* [48]. The authors proposed a privacy-sensitive service selection algorithm based on AFSA for QoE restrictions. The AFSA method is one of the swarm intelligence approaches that operates based on the population and the stochastic search. In this work, a specific service is regarded as "foods" and Artificial Fish looks for it in its visual field and moves randomly from a certain distance to it. The system is initialized as a set of randomly generated potential solutions, and then performs the search for the optimum one iteratively. A scalable multi-layer QoE quantitative model is designed to enable privacy preferences.

7.2.2 Social-inspired approaches

In this class of methods, a new paradigm is introduced, called SIoT, which applies social networks aspects to IoT, thereby allowing objects to autonomously elaborate their social relationships to form several communities having common interests in order to discover other "friendly" objects and their related services [119]. These devices can form five main relationships which are [120]: *parental object relationship* (objects having the same manufacturer); *co-location object relationship* (objects present in the same area); *co-work object relationship* (between devices that are ready to work jointly); *ownership object relationship* (objects having a common owner); and *social object relationship* which is launched among objects when they have a chance of coming near each other, continuously. In the recent literature, we find different approaches in that regard. One of the first works within this class was presented in [42]. The authors proposed a self-organized decentralized social network within a P2P infrastructure and endowed with a swarm intelligence search strategy. They designed a homophily-based user model to select candidate neighbors in each routing step taking into account social relationships as well as content semantic features. In [80], Li *et al.* developed a decentralized trustworthy context and a QoS-aware SD framework using an ontological model to present and match service with context and QoS requirements. The discovery mechanism consists of two models: a social recommendation-based model and a decentralized locality-preserving model. The proposed system utilizes social links between IoT entities to forward service advertisements/recommendations in the IoT network. Besides, they designed a trust propagation approach that collects feedback from service consumers to improve trustworthy service selection. In [91], Pruthvi *et al.* have designed an SIoT platform for a smart college environment allowing objects to create communities by establishing their own relationships based on rules set by their owners. A social profile for objects is thereby created based on the received data from IoT applications. In another interesting work, [43], Xia *et al.* have introduced a decentralized social-like semantic-aware SD mechanism. This framework is dedicated for a large-scale IoT to design a social network among autonomous devices. In this solution, services are ranked by calculating their correlation degree using the fuzzy logic method. Based on a restricted contact graph, the SLSA implements an adaptive forwarding policy, where each service discovery request is forwarded to a selected subset of neighboring devices in a preferred order. A similar solution is proposed in [94], where a community detection algorithm together with a SD mechanism operate according to user preferences, movement and social similarities.

Moreover, Araùjo *et al.* proposed in [69] an efficient SIoT SD mechanism, called Social Profile Search, based on the information and characteristics contained within social profiles of objects. This solution revisits the breadth-first search algorithm by introducing a new stopping condition once all requested services are found, regardless of the number of visited objects. For a sake of service composition , Chen *et al.* proposed in [54] a social network based approach for IoT device management and service composition, in which IoT devices are represented as RESTful Web services. A 3-dimensional social network model (location, type and correlation) is applied to describe the relationships between devices/services and to enable service selection. To improve trustworthiness during the SD, Nizamkari proposed a scalable graph-based collaborative filtering recommendation system in SIoT that incorporates trust [99]. A node can thereby rate the provisioned services and hence selects the best ones. In a recent work, Khanfor *et al.* developed an automated SD process for mobile crowdsourcing applications within an SIoT [98]. Two community detection algorithms, namely Louvain and OSLOM are proposed and applied to a real-world IoT dataset to form non-overlapping and overlapping IoT devices groups sharing common characteristics and using social relations. This solution provides besides a NLP-based tool to handle crowdsourcing textual requests and accordingly find the list of IoT devices capable of effectively accomplishing the required tasks.

7.2.3 Physical-inspired approaches

Algorithms under this classification are characterized by the fact that they imitate the behavior of physical phenomena, such as: gravitational forces, electromagnetism, electric charges and water movement. So far, few works have considered the use of physical models in the context of SD in IoT environment. Rapti *et al.* were among the first to investigate such a technique. They presented a solution based on APFs exerting forces over user requests [90]. APFs use observations from physics where complex, stable motions that have many properties arise from relatively simple laws. They have been extensively studied for robot navigation and routing in wireless sensor networks. In the proposed model, each device in the network represents an SPN that provides its functionalities through services that are handled internally by software agents. Each service agent generates APFs that exert attractive and

repulsive forces over all user requests in their neighborhood. The resulting force is used to route a user request among the SPNs of the IoT network so that all requested services are delivered to the user on time. The selected services can belong to the same or different network nodes (i.e., SPNs). In the solution proposed in [79], the authors developed an IoT service selection algorithm, called (EG-ERWCA), which is based on the evolutionary game theory combined with a metaheuristic search algorithm (evaporation-based water cycle algorithm) to improve the QoS of the network.

7.3 Context-based approaches

Such a class provides SD mechanisms that take into account the context and the environment within which the solution has to be implemented and deployed. Three categories are further recognized in our classification: *Objective context* that deals with techniques that are centered on the physical environment surrounding the user; the *Subjective context* that refers to the approaches that are mainly centered on users; as well as a *hybrid subclass* which combine the objective and the subjective context aspects.

7.3.1 Objective context based approaches

These solutions consider objective information which is supplied by the service providers to perform the SD and the selection. Within this class, approaches are further distinguished into those based on location, and those based on event.

a- Location-based approaches: Such techniques consider the geographical coordinates or the relative references of the user to select services. They support indexing structures and ranking methods to speed up the processing of requests and hence produce real-time results. Many interesting works can be quoted within this class. For example, the authors in [116] proposed a location-aware discovery system for IoT based on a decentralized service-oriented platform using the swarm network to enable device communications. For this effect, an ontology for the geolocation information is designed and the service description is expanded to support that information. Wang *et al.* put forward a unified IoT SD framework based on an updated version of the Auto-ID Center's ONS architecture using PML to provide a notice-and-consent agreement for IoT [76]. Furthermore, a mobile application interacting with this framework is designed to notify users of interesting nearby services, or about existing running services that are collecting their data secretly. In the same regard, Wang *et al.* developed a scalable sensor SD architecture using a semantic geospatial indexing technique [112]. This method is resilient to dynamicity and reduces the number of expensive update operations. In other respects, Tao *et al.* introduced in [121] a location-based trustworthy services recommendation system that operates in a cooperative-communication-enabled Internet of Vehicles. This solution aims at satisfying user's requirements by taking into account the space-time characteristics as well as the direct and indirect friendships among users. In another work, Baek *et al.* proposed a spatio-cohesive service selection approach using a RL method [49]. RL is about training a smart agent that learns the optimal policy of taking actions in a certain type of environments to achieve a given goal, by simulating a number of trials and errors. In this solution, the agent hosted on each user's mobile device peers at other devices that are providing services in the surrounding environment. Then, it classifies them into several categories. For each service type, the agent selects one IoT device by applying a neural networks based policy. Hence, all the selected services form a joint selection.

b- Event-based approaches: Such algorithms aim at automatically handling events that may occur in a dynamic environment. Each event is characterized by a type and some parameters. Initially, the sensing service systems collect and analyze the real-time data, get the events, and thereof perform real-time processing as a response by applying user defined event rules. The generation of these events can trigger one or more services to be invoked. Moreover, events are generated by producers and received by the consumers. This can be viewed as a publish/subscribe architecture, where entities can subscribe for some event types and get notified automatically for those events once happened. Let's take an example of a camera' service, which is defined for taking pictures and videos, and the position' sensors which provide the position' service [122]. When position changes are detected, the position' service generates a "position changing event". After receiving this event, the camera service can take pictures and videos. That means "position change event" is driving the camera service execution. Many interesting works can be identified within this class, as for instance, the work in [123] that proposed an event-driven and service-oriented situation-aware IoT service coordination platform. The latter is endowed with a reliable real-time data distribution model which is based on publish/subscribe brokers for efficient topic searching. In [50], the authors introduce a service-oriented, user-centered and an event-aware framework dedicated for events monitoring in ambient environments. This solution enables self-adaptation to cope with unpredictable changes by using service replacement and replanning in case of failure. In [101], Gupta *et al.* have proposed an event-driven SOA-based IoT architecture, which can be seen as an extension of a broker-based architecture to support real-time, event-driven, and active service execution. Moreover, Liu *et al.* [44] have proposed a method of semantic service matching based on word embedding in event-driven SD. Two types of semantic services related to events were introduced: event-recognition and event-handling services.

7.3.2 Subjective context based approaches

Subjective context refers to the need of information provided by the service consumers to conduct the SD and the selection. It focuses on extracting the user's cognitive activities, in order to provide personalized services according to user preferences, feedbacks, task and emotional states [124]. Two types of approaches can be further distinguished within this class, those that are based on user requirements, and those based on the user feedback.

a- User-requirements based approaches: Such solutions are based on user preferences and profiles to select the most appropriate services matching these requirements. Among the works identified within this class, we can quote the one defined in [61] that considers a context-aware SD architecture for IoT. The discovery process adopts DBN by combining the role of the context and the object relationships to design an ontology-based model, making it possible to handle uncertainty and temporal aspects. The

authors in [51] developed an architecture that enables a representation of user preferences and manipulates relevant description of available services. Based on this architecture, user and service profiles are modelled and a service selection algorithm is designed for a context recommendation process. For this effect, a matching score function is introduced to rank candidate services by aggregating the matching scores (syntactically and semantically) of each attribute's values from the service and the user profile. In [125], Jin *et al.* introduced a physical service model to describe and select physical IoT services while dealing with user's requirements. The model identifies three main components (devices, resources, and services) and allows to describe the relationships between them. A physical service selection algorithm is designed taking into account spatio-temporal information to evaluate services, and selection is then performed dynamically based on the user's preferences. In the same regard, Ali *et al.* have proposed a dynamic service selection and execution approach based on user preferences and situation, by using a real world knowledge model to encompass situational and user preference information [126]. More recently, Pattar *et al.* [62] have developed a progressive SD approach together with two strategies that use semantic and proximity measures to match user's requirements with IoT resources characteristics.

b- User-feedback based approaches: Such techniques require the assessment of the performances of services. Hence, SD and selection are mainly based on user experience, comments and feedback. Within this class, we can quote the work proposed in [127] that designs a service composition model with a minimal human intervention. User' feedback is considered to determine whether a discovered plan is correct or not in order to improve the SD performances. More recently, Quan *et al.* put forward an IoT service selection model that considers the aggregation of user's feedback and service objective performance analysis [92]. An RL algorithm is applied to learn all subjective and objective assessments. The solution assumes a dynamic IoT environment by integrating the user's mobility factor that could impact service accessibility. The subjective assessment is performed by calculating four factors: privacy, reliability, availability, and response time. In another work, Abu-Safe *et al.* proposed a service selection model using Likert scale measurements with an improved-PSO that considers user-feedback to rank services [128]. The proposed approach regards the user preferences and their feedback as reputation information characterizing services. Five Likert scale measurements are considered to classify the reputation (Strongly Agree, Agree, Neither, Disagree, and Strongly Disagree). When a customer requests a service, it will be selected based on the related feedback saved in the service registry.

7.3.3 Subjective and objective context based approaches

Such approaches attempt to combine both subjective and objective contextual information when performing SD and selection. Some surveyed solutions have adopted this type of algorithm, for example, in 2014, Win *et al.* designed a matching, ranking, and selection model to meet the distributed requirements of dynamic networks in IoT environments [129]. The proposed model integrates objective and subjective factors to optimize the QoS of the network. They considered a similarity aggregation method to calculate the subjective factors. User appreciations are extracted using the QoS ontology, WordNet, and ontological reasoning. The selection algorithm is based on the ANN back-propagation to determine the objective factors and improve the selection performance rate for an acceptable real-time service selection. In 2017, Mejri *et al.* proposed a solution to address the scalability of service selection in the IoT [63]. For this purpose, they implemented a self-adaptive approach based on a combination of a QoS prediction model, which considered the user context, service context, and network context. The selection algorithm combines ANN and the TOPSIS ranking method to promote the best service to the consumer by optimizing the response time and the reliability. Moreover, Qi *et al.* put forward a Time–Location–Frequency –aware service selection approach based on the weighting model of Historical Records [52]. They first weigh each HR based on its service invocation time and location. Then, they weigh each candidate IoT service based on its invocation frequency and finally, they evaluated each candidate IoT service to return the quality-optimal one to the target user. Recently, Gao *et al.* [130] proposed a holistic model to predict the QoS values in IoT by using fuzzy C-means to cluster contextual information related to users and services. An NCF is designed to learn the in-depth latent features, using objective and subjective information. NCF is used to aggregate both information to perform both the prediction and the selection.

7.4 Protocol-based approaches

Discovery protocols allow devices to automatically become aware of the identity of other devices and their services in the network without the need for human intervention. The traditional SD protocols proposed for wired, powered or high-bandwidth wireless networks are not suitable for the IoT environment. For this purpose, either new protocols are being proposed as IoT standards by different organizations, such as: the OMG, the IETF, etc. or existing solutions are adapted to meet the IoT requirements. The main functionalities of SD protocols are: publication, registration into the directory (if available), discovery and resolution. Moreover, for protocols based on a directory, several maintenance features may be available to update, remove, and validate the entries. The publication process is used by protocols working in a distributed manner, allowing devices to announce and disseminate their own offered services. As regards the registration process, service descriptions provided by devices are stored into the directory to make them available during the discovery process. It can be carried out through stateless mechanisms (the directory listens to service advertisements and populates its service database), or stateful ones (an explicit registration is carried out directly from the device to the directory). Moreover, the role of the discovery process is to locate instances of some relevant services (based on the type, the domain, the location, etc.) and can be performed directly on the neighborhood, on a centralized local directory, or at a higher scale on a global directory. Finally, the *resolution* relates to translating the discovered service instance to an accessible end-point address or host name. According to [131], three main categories of protocol based approaches in IoT environments can be identified: Service-oriented, message-oriented and resource-oriented.

7.4.1 Service-oriented approaches

This class of approaches allows objects to expose their functionalities as services, which can be searched and accessed through the use of multicast protocols (by the device name, type, scope, etc.). Devices matching the probe send a unicast response. When a device joins the network, it announces itself by sending a multicast "*Hello*" message. Similarly, when leaving the network, it announces this through a "*Bye*" message. Within this class, we report two clusters of approaches, DPWS and DNS based approaches.

a- DPWS-based approaches: DPWS is a Web service standard that provides the plug-and-play functionality for networked devices with minimal interactions with the running services. This protocol is based on existing standards, including: XML, WSDL, SOAP, HTTP, etc. and it offers a publish/subscribe eventing mechanism under the Web service-Eventing specification. DPWS enables devices to be dynamically located when they connect to the network, while it provides mechanisms to retrieve their metadata and discover the embedded services as well as their functionalities. Two types of services are recognized in DPWS: A hosting service and a hosted service. The latter represents the service offered by the device, functionalities of which are described using WSDL. The former denotes the device that hosts services that can be invoked and processed. In the literature, DPWS has been considered in few works: Among the first published works, we quote the one in [87], by Guinard *et al.* who implemented a process, called Real-World SD and a provisioning process to assist the developers at the design time of the SD approach. DPWS is considered for discovering and registering devices and their services through a RESTful API. Son Han *et al.* have extended DPWS to operate in a REST proxy architecture so that to reduce the latency as well as to simplify the global topology [102].

b- DNS-based approaches: DNS protocols are based on a distributed database that stores service description records that can be perused by querying a DNS server. In the DNS-SD approaches, the providers register services in a DNS server by including their description (service name, type, domain, etc). These protocols adopt a request/response functioning, where service consumers are called to query the DNS server for a given service description, which returns the corresponding records. The Multicast DNS-SD is a distributed extension of the DNS protocol that adopts the same service description structure as DNS-SD. It proactively discovers and advertises new services that are made available in the cyberspace. Such solutions require a minimal configuration, and can operate in infrastructure-less environments while being resilient to network failures. Some interesting works can be quoted within this class: Klauck and Kirsche were among the first to address such a solution by introducing a lightweight SD, called "*Bonjour Contiki*" or "*uBonjour*" [104]. The latter assumes a DNS-SD/mDNS protocol and operates on Contiki OS embedded devices to enable self-configuration and a standardized SD. First, a device initiates the SD by sending a pointer record to mDNS containing the name of the searched service. If the service is found, uBonjour posts an event to inform the processes about the IP address and the port of the service. In 2014, Djamaa and Richardson developed an efficient, extensible and adaptable discovery protocol for CNNs to overcome mDNS limitations by coupling DNS-SD with EADP [93]. EADP adopts a fully-distributed push-pull approach to service three main components: (i) the *user agent* responsible for discovering services via multicast; (ii) the *service agent* responsible for registering and advertising services' description; and (iii) the *state maintenance mechanism* responsible for managing local directories and reacting to network dynamics. To enable scalability in large networks, Stolikj *et al.* have extended the mDNS/DNS-SD protocol with a context-based model for describing and discovering services [64]. In this solution, any given service can be associated with a set of context tags which represent atomic descriptors of the context properties. These tags are pre-configured at commissioning time, or are learned from the environment at run-time. Therefore, discovering a service consists of sending one or more queries, listing a combination of wanted/unwanted context tags, the service type and the logical domain. In the same regard, Mahyoub *et al.* have proposed an optimized, distributed and autonomous discovery solution for smart devices. The purpose is to reduce the computation and traffic effort induced by mDNS/DNS-SD protocols [65]. For this effect, three approaches are designed: Probing and advertisement suppression; Discovery responses suppression; and the selection stage optimization.

7.4.2 Message-oriented approaches

Such solutions are based on exchanging asynchronous messages between the distributed devices to discover services. The reliability and the QoS of the exchanged messages are controlled by centralized entities, called brokers. MQTT and XMPP are the most famous protocols considered in this context.

a- MQTT-based approaches: MQTT is an OASIS standard messaging protocol for IoT which runs on top of TCP (that is encrypted by SSL/TLS to provide a secure communication). It is designed as an extremely lightweight publish/subscribe messaging transport that is ideal for connecting remote devices with a small code footprint, minimal network bandwidth, high latency and low reliability. In MQTT, clients connect to the broker to publish messages to a particular topic, or to subscribe to already published ones. When publishing a message to a particular topic, all the subscribers are notified. This protocol enables one–one, one-to-many and many-to-many communication possibilities. In addition, the reliability of messages in MQTT is taken care by three QoS levels: QoS level 0 means that the broker/client will deliver the message once with no required confirmation, QoS level 1 means that the broker/client will deliver the message at least once with confirmation required, and finally, QoS level 2 means that the broker/client will deliver the message exactly once. In the literature, several MQTT based SD approaches can be identified. Venanzi *et al.* were among the first to investigate such a solution. In [78], they revisited their previous work in [132] that introduces an MQTT-driven node discovery protocol to evaluate the impact of the dynamicity of the advertiser nodes on the device discovery success and the sustainability of the battery-powered IoT nodes. The MQTT broker serves as a fog node to trigger turning on/off the BLE interfaces of the surrounding objects by monitoring their trajectories. Sahlmann *et al.* have proposed an extension of MQTT, called NETCONF-MQTT, to bridge and interconnect constrained networks with the public network [85]. Device capabilities are described using OWL (A oneM2M ontology which improves interoperability), and are published via the MQTT broker to the network. The NETCONF-MQTT bridge generates dynamically YANG data models from the semantic description of the device capabilities. Recently, Kim *et al.* proposed a context-aware autonomous SD in oneM2M architecture to enable auto-configuration

by implementing an MQTT asynchronous request/response scheme [83]. In this solution, devices are asked to register their services in the resource directory through the gateway which autonomously interacts with them using semantic Web technology. Once registered, the devices are recognized as oneM2M resources and are able to access other devices and discover their services using MQTT. Likewise, Pareira *et al.* developed a distributed MQTT-based resource discovery architecture for M2M communications, that enables zero-configuration networking by providing plug and play capabilities for network devices [84].

b- XMPP-based approaches: XMPP is an XML-based communication protocol for instant messaging which supports publish/subscribe model and allows servers with different architectures to communicate. It runs over TCP and uses TLS to encrypt the communication channel. XMPP supports asynchronous communication by exchanging the data using XML streams between the server and the client, where the XMPP addressing is used to locate devices in the network. In XMPP, instant messages can be sent between the devices on the Internet, independently to the operating system they are running on. In addition, all the XMPP's clients, servers and services, present in the network, are addressable. Many XMPP extensions have been implemented in the context of IoT to deal with different issues: Efficient XML Interchange Format, sensor data, service provisioning, data transfer control, service discovery. Among these extensions, we can quote the work of Wang *et al.* [100] which proposed a lightweight XMPP for resource-constrained IoT devices by optimizing the energy consumption and the computation effort. In this solution, the publisher can adjust the data information published in the server to deal efficiently with event-driven and periodic data publications.

7.4.3 Resource-oriented approaches

In the resource-oriented class, SD techniques are based on REST. Such solutions provide a resource monitoring mechanism and device discovery functions so that to minimize human intervention. Such a class of approaches can either rely on using the REST protocol or CoAP for data transfer or control.

a- REST-based approaches: REST architectural design allows developing RESTful Web services that communicate and access resources by their URIs over HTTP protocol and its methods. Many REST based SD solutions can be identified in the literature. Khodadadi *et al.* were among the first to design a Rest based SD solution [111]. They proposed a framework to define, discover and compose things and their services that are made accessible using RAML. Besides, to provide complex service functionalities, service composition is enabled by chaining loT service calls together.

b- CoAP-based approaches: CoAP is a REST-based application layer protocol that was designed to interoperate with HTTP and the RESTful Web through simple proxies. Thus, it can easily be integrated within the web. Also, it shares the same methods as HTTP and uses URIs to invoke services. This protocol supports both request/response and resource/observe (a variant of publish/subscribe) models. CoAP has a special feature in addition to HTTP known as "Observe" resource, which allows clients subscribing to a particular resource to receive an asynchronous notification message each time its value changes. Hence, CoAP clients can retrieve the resource representation and keep it updated by the server over a period of time. Moreover, CoAP is a binary protocol that runs over UDP to save some bandwidth and to ease implementation, while it uses DTLS to provide security (e.g, privacy for datagrams, confidentially, integrity and authentication). Since UDP is an unreliable transport protocol, CoAP implements four types of messages: *Confirmable*, *Non-Confirmable*, *Acknowledgement* and *Reset*. Besides, this protocol allows UDP broadcasting and multicast to be used for addressing. Furthermore, CoAP endpoints have the capability of caching the responses. Therefore, previous responses can be reused to respond to another similar request in the future, thereby reducing the response time and the traffic overhead. In the literature, COAP has been adopted in many SD approaches. To the best of our knowledge, Cirani *et al.* can be recognized among the first to address such a solution [82]. The authors introduced a P2P-based architecture for self-configurable, scalable, and reliable large-scale SD using IoT gateways. The latter which are designed as peers collaborate to provide an efficient name resolution for CoAP services. This technique which is characterized as using zero-configuration operations, supports a local-scale discovery. In the same regard, Tanganelli *et al.* have designed an edge-centric distributed architecture to enable SD. The latter is based on a DHT which is maintained by IoT gateways and deployed on fog nodes [97]. In [46], Yachir *et al.* have proposed to extend CoAP with lightweight semantic-rich information by defining appropriate CLF attributes describing both IoT resources and user requests to facilitate semantic resources discovery. An IoT resource includes the physical device and its hosted software services (device and service descriptions). A framework integrating the proposed semantic-enhanced CoAP model was designed and implemented. In [115], Djamaa *et al.* proposed a new CoAP-based discovery mechanism built on the newly standardized FETCH method while developing its specifications, its rules and its semantics to overcome the limits of the COAP GET method. In this solution, the parameters of FETCH are transmitted in the payload of the request rather than in the context of its URI. In addition, the authors proposed a mechanism to exclude resources that are not needed and include other desired features to the response. Finally, the work includes a resource discovery layer to the reliable and secure openThread networking protocol. In [73] Ferdousi *et al.* have investigated the use of a cloud-based middleware, called "LOAMY", for CoAP-based IoT SD, which is dedicated for interoperable mobile constrained IoT networks. LOAMY works as the existing CoAP distributed approach where it is required. But, for a complex request where the existing discovery mechanism can be expensive, it requires the middleware to make the discovery mechanism more reliable. In a recent work, Vandana *et al.* enrich CoAP with semantic aspects to improve resource discovery [96]. The CoRE Link Format attributes are redesigned so that the user request and the resource description can be handled semantically. Moreover, to better cope with a distributed topology, a social framework was proposed to capture the social relationship between devices.

7.5 QoS-based approaches

Such solutions take into account QoS aspects that characterize the performances of the services, devices as well as the network infrastructure when processing the service discovery and selection. In the literature, a sustained effort has been made to provide

the best QoS in IoT. According to the QoS aspects managed during the SD process, we identify four sub-classes of QoS-based approaches based on: Security, energy consumption, network infrastructure or by combining different QoS metrics.

7.5.1 Security-based approaches

Such solutions are mainly focusing on improving security aspects such as: authentication, trust, privacy, object access control, data integrity and so on. Therefore, the selected elements (services or objects) must provide the best response to the security requirements as defined by the user and the running application. Datta *et al.* were among the first to deal with security in SD in the context of IoT [95]. They developed a RESTful SD framework dedicated for smart home systems that provides secure solutions for authentication as well as access control for thing discovery. Thereafter, Dahbi *et al.* have put forward a secure and distributed SD architecture to supply chain domain in IoT [74]. This solution is based on DHT mechanisms and P2P systems to better cope with ONS and EPCglobal technologies. To tackle authentication, data integrity and confidentiality issues, two security schemes are developed to detect suspicious lookup queries. The first is based on a statistical Gaussian model and the second is a CHMM based probabilistic model. In 2018, Lee *et al.* proposed in [75] a web-based system making it possible to discover nearby IoT services by guaranteeing user' privacy. For this effect, they designed an architecture to describe user privacy properties and profile. This solution thereby enables controlling the collection and usage of sensor data according to each user's privacy preferences and collaboratively assessing its potential privacy risks. In [67], Ahmed *et al.* proposed a centralized trustworthy SD solution to securely deliver services to consumer. By considering a trust-management model as well as services' context and QoS, session keys are generated and distributed to consumers to secure the SD process. In the same regard, Kurte *et al.* introduced a SOA based DSF for an open-source platform that supports the development and the deployment of trustworthy and privacy-preserving distributed IoT applications [133]. This solution provides common protocol and infrastructure for a secure service specification, registration, discovery, publishing and subscription, over insecure public networks. More recently, Kalkan and Rasmussen proposed a distributed trust framework on top of a structured P2P network based on DHT, which enables trusted communications among identified or unknown devices during SD [81]. A device in the system can be a provider and a requester of a service at the same time. Each device is monitored by other nodes, called Reference Holders, that are responsible for holding a trust value for this device. The proposed framework provides trust aggregation using static weighted sum, service provision as well as feedback aggregation.

7.5.2 Energy consumption based approaches

The main goal of such techniques is to select services that guarantee an optimal energy consumption to ensure a high availability of battery-constrained objects and therefore of their services. Huang *et al.* were among the first to investigate energy issue in SD [55]. They proposed a very interesting approach to map and co-locate neighboring virtual services on the same physical devices. The selection process has then to consider this information so that to reduce and balance the energy consumption during communication to prolong the network lifetime. Heuristic algorithms were employed to co-locate services by considering a single-hop and multi-hop networks by modeling the problem as a quadratic programming problem and solving it with the ILP model. In 2015, Na *et al.* proposed a service selection approach based on a dynamic evolutionary game for multiple IoT applications to balance the energy consumption over the IoT devices [134]. They used evolutionary equilibrium to evaluate the payoff functions of each service selection behavior based on a global view of competitions and interactions. In 2016, Khanouche *et al.* developed an energy-centered and QoS-aware service selection algorithm for IoT services composition [58]. This approach uses a MOP algorithm and operates in two phases : 1) First of all, services that are offering the required QoS level are pre-selected using a lexicographic optimization strategy and QoS constraints relaxation technique ; and 2) The most appropriate services among the preselection set are considered for composition using the concept of relative dominance of services in the sense of Pareto basing on energy profile, QoS attributes and user's preferences. In other respects, Sharma *et al.* proposed an efficient energy framework for device discovery in 5G-based IoT and FANETs [135]. This solution considers XML charts to perform device discovery on the basis of networks state cost and the available energy. In [114], Albalas *et al.* proposed an adaptive Fibonacci-based tuning protocol for service and resource discovery in IoT, taking into account SA battery level in a centralized CoAP. This solution achieves to reduce the power consumption of nodes and prolong the network lifetime. Likewise, Lin *et al.* have designed a sensor selection algorithm to operate in a large-scale environment [59]. The algorithm considers mainly energy and distance parameters to select sensors among candidate ones. The energetic profile of sensor integrates the available battery energy, the capacity of energy harvesting (e.g., wind and solar), and the green index to rate the total pollution level of the sensor.

7.5.3 Network-infrastructure based approaches

These solutions are dedicated for resource constrained networks, in terms of bandwidth availability, storage capacities, latency, etc. The main goal of such algorithms is to improve the QoS of the network while reducing the cost of the SD. Among these solutions, we can quote the work of Antonini *et al.* in [136]. The authors proposed a lightweight multicast forwarding mechanism for SD in LLNs with low memory footprint and zero-configuration operations. This solution considers RPL with a filtered local flooding to adjust to duty-cycled devices, while it introduces Bloom filters to detect duplicate packets and prevent forward loops. A similar work is proposed in [105], that considers a distributed service registries enabling a fast and flexible SD method in mobile IoT environments using hierarchical Bloom filters. The latter makes it possible to reduce the configuration cost and the network traffic exchanged among registries in discovering the available services. In 2017, Sun *et al.* designed a framework making it possible the integration and the co-operating of smart IoT functionalities so that to satisfy different user requirements specified on more than one smart device [55]. The proposed solution considers the energy consumption, the spatial constraints (location and communication radius), as well as the temporal constraints to specify the exact time duration required to meet the user's requirements. More recently, Nguyen *et al.* have presented decentralized and improved CCN-based MEC service deployment/discovery protocols and platform

[70]. The service deployment protocol assists service providers in self-deploying their services on MECs through RESTful API whenever a service is requested. Besides, MECs are put into multiple clusters/groups based on geographic location to decentralize the network control. The RTT from mobile IoT devices to neighboring MECs is assessed to select the best destination node with respect to the QoS. This solution achieves in overall to reduce the traffic overhead and the transmission delay. In the same regard, Maiti *et al.* have introduced in [77] a deployment strategy of latency-aware fog smart gateways for IoT services which optimizes the number of fog nodes so that to reduce the latency induced by traffic aggregation and processing.

7.5.4 Combined QoS metrics based service selection

This class of approaches is designed to select the best services and objects in terms of their QoS performances described by their non-functional parameters (metrics). The selection is generally conducted in the purpose to compose services or combine objects to design a more complex framework to improve further the capabilities and the overall performances. The QoS metrics involve all the performance aspects of the network as for instance: energy consumption, security, reliability, throughput, etc. Hence, the selection process will balance between all the defined metrics (which could be conflicting) to elect the optimal services among the candidate ones. The QoS metrics should be assessed for every selectable element and be continuously updated to guarantee the most accurate result, each time a selection request is processed. several SD solutions have considered such algorithms, which require generally to use MOP methods or ranking methods as for instance MCDM. In 2014, Perera *et al.* proposed in [137] a solution to rank large-scale sensors with the same functionalities, but with different QoS features such as: accuracy, reliability, battery lifetime, etc, They designed and implemented an ontology-based context-aware sensor SD model to promote the most appropriate sensors according to user requirements. This model uses a comparative-priority based weighted index to remove sensors with a lower weighted context property calculated on user preferences. Then the set of candidate sensors is further pruned by removing the sensors placed away from the user. Finally, by specifying an acceptable range of context property values using semantic relational operators, the premium set of candidate sensor is set up by considering user's priorities. The ranking process is then run to select the appropriate sensors that satisfy the user's requirements on multiple IoT distributed servers. In 2018, Alsaryrah *et al.* designed a selection algorithm to promote an appropriate set of smart objects by considering the traditional QoS and the energy cost [56]. They defined a MOP model that aims to minimize the traditional QoS metrics (execution time, network latency, and cost) as well as the energy consumption, for the set of objects to promote. For this purpose, they proposed a BSPO algorithm combining four pruning techniques: by the cycle, nadir point, efficient set, and label. More recently, Hosseinzadeh *et al.* presented a hybrid ANN-based PSO algorithm to evaluate the optimal QoS-based service composition for Cloud-Edge computing in IoT environment [57]. This algorithm operates in two stages: in the first stage, a global search process is performed to predict the optimal position for existing particles to prevent premature convergence. Then in the second stage, a PSO based optimal service composition is processed by considering user' QoS requirements. Three QoS factors were considered comprising the response time, availability, and prices. Baranwal *et al.* proposed in [60] a multi-criteria group decision-making framework to select and rank IoT services by identifying various QoS metrics related to each of the IoT components (computing, communication or thing). The proposed model uses fuzzy TOPSIS to handle rank reversal problem and judgments of Decision Makers in linguistic term and integrates an ordered weighted averaging aggregation operator with it, to model risk attitude of DMs and to aggregate their preferences for the final ranking. The same authors extended their solution to consider a combination of TOPSIS with AHP [138]. AHP was used to calculate the weights for the QoS criteria and TOPSIS used to rank the service providers. They describe a QoS parameter according to three IoT components, (things, communication entity, and computing entity). Hence, nine QoS parameters were identified: operating temperature range, resolution, accuracy, delay, jitter, pricing, availability, throughput, and response time. They demonstrate the effectiveness of the proposed approach along with the sensitivity analysis for showing the robustness of the proposed framework.

8 Research questions, discussion and evaluation

In the previous section, we surveyed the most recent and interesting works in the literature that were dedicated to design and develop SD and selection approaches in the IoT context. The taxonomy provided in section 4 endeavors to classify these solutions according to different concepts and paradigms. In this section, we conduct a comparative study of the different classes of solutions, considering several aspects and criteria. Then, we report the advantages, the limitations of each class of our taxonomy. Concretely, according to the literature, this study aims to answer the following research questions:

1. What are the advantages and the limitations of using each class of methods relatively to each aspect of the SD system: architecture, description, and SD algorithms ?
2. Which are the most dominant classes in the literature, and how the tendency is evolving in the last three years ?
3. Which descriptive methods, architectural designs and implementation models are the most considered to operate with each class of SD and selection algorithms ?
4. Which QoS metrics and performances are improved by each class of SD and selection algorithms?

We believe that this study will provide researchers and interested individuals with the stepping stone into understanding the full picture of the existing SD and selection approaches in the IoT.

8.1 The considered QoS criteria & IoT requirements

The following parameters are considered to compare the performances of the different reviewed approaches. This set is an aggregation of different parameters proposed in the literature [33] [139] [21]. Some parameters are classical and are already considered in

the context of web service selection: *Accuracy, Precision, Scalability, Trust, Overhead, User preferences,* and *User feedback.* Others have been redefined to meet the IoT requirements: *Availability, Interoperability, Security & privacy, Reliability, Autonomy, Delay, Network QoS performances, Load balancing,* and *Fault-tolerance.* Finally, the parameters: *Mobility, Energy consumption, Dynamicity,* and *Context-awareness,* have been introduced to measure some performances related to IoT context.

- **Interoperability:** A discovery system must be able to operate despite the heterogeneity of data, protocols, systems and applications, etc.
- **Scalability:** The system must be able to add a new device, service or operation and to handle growing amounts of requests without affecting the quality and the performances of existing services. Therefore, Scalability is the ability to manage a large volume of connected IoT resources.It can be defined as producing maximum throughput in a minimum response time.
- **Mobility:** When IoT objects are mobile, they are led to change frequently their location. Therefore, a discovery system must deal with such a dynamic scenario.
- **Dynamicity:** It refers to the ability of a system to consider the flexibility of the IoT network, changes in network topologies, IoT resources and so on.
- **Autonomy:** Such a property is ensured by integrating a certain degree of intelligence while minimizing human intervention.
- **Fault-tolerance :** A discovery system should support fault detection and fault recovery.
- **Availability:** It refers to the probability that a service or a device is accessible and available, anywhere and anytime, when required for use, under normal operating conditions. It must be ensured at both levels: software and hardware. In case of downfall, the system should recover as soon as possible by providing a better operational solution.
- **Energy consumption:** IoT objects have to deal with limited power resources while they should be active continually to sense real-time data and react to received requests. Therefore, the design of a SD system should take into account this issue.
- **Load balancing:** It refers to efficiently distributing the incoming traffic across a group of back-end servers to maximize the speed and the capacity utilization and to ensure that no server is overworking, which could degrade the overall performances.
- **Reliability:** It guarantees a proper functioning of the SD system based on its specification and aims to increase the success rate of the IoT service delivery.
- **Accuracy:** It determines the relevance of the SD system and its degree of correctness. It can be defined as the maximum uncertainty between the actual value measured and the standard value set at output parameters.
- **Precision:** It is the probability that all the selected services, extracted over the total number of services in the repository, are correctly relevant.
- **Context-awareness & user preferences:** It denotes the ability of a system to gather information about its environment at any given time and to adapt its behaviors accordingly, by considering in addition the user preferences.
- **User feedback:** It denotes the information collected from users that measure their appreciations towards a provided service. Such a parameter is used to elaborate the user experience to advertise existing services to potential consumers.
- **Security & privacy:** A discovery system must implement mechanisms to prevent, detect and deal with fraudulent service customers and providers. It must ensure a secure service delivery in terms of: confidentiality, authentication, data integrity, privacy and so on.
- **Trust:** It measures the fulfillment of the requester towards the service provider. The trust guarantees to the requester that the service provider is acting in a way that it is supposed to be honest, secure and reliable when processing its request.
- **Delay:** This parameter assesses the time needed to process a given request.
- **Overhead:** Discovery and selection methods may induce an overhead at different levels, including: network, communication, processing, exchanged packets, control and management, maintenance, etc.
- **Network QoS performances:** A discovery and selection approach must take care of the network resources by optimizing their management, and while considering the network constraints and requirements, such as: bandwidth availability, network capacity and latency, packets and messages loss, throughput, etc.

8.2 SD and selection approaches comparison and evaluation

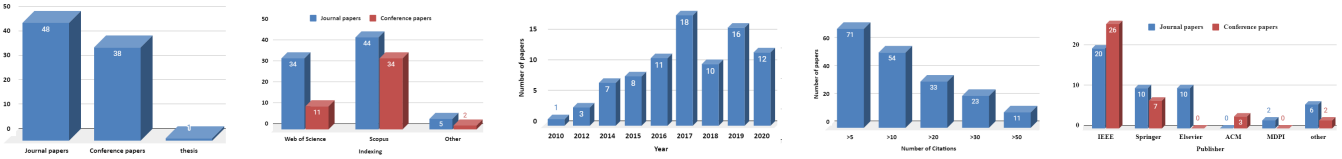

Figure 7: Profile of papers considered in the study.

In this section, we compare the different classes of our taxonomy by providing some useful statistics. Then, we evaluate all the surveyed SD and selection approaches considered in our survey according to the QoS metrics discussed previously. For this effect,

87 approaches have been considered, profile of which is shown in Figure 7. Three criteria have been considered in the selection of these papers, the first one deals with their relevance in terms of number of citations and the reputation of the publication. The second criterion considers the novelty of the papers. The third one assumes the existence of at least two papers within each class of our taxonomy. The profile shows that 56 out of the 87 approaches have been published within the last 5 years. All the considered papers are recognized by well known indices, and particularly, we have 82 papers that are either indexed in Scopus or in Web of science. Moreover, 54 papers have already more than 10 citations, and 23 papers more than 30 citations.

The expertise that we conduct in the sequel makes it possible to draw out the benefits and the gaps of each class of our classification and locate the context and the requirements under which each can operate. The three essential aspects for designing an SD solution are thus addressed, which are: Architecture, Description methods, and Discovery and selection algorithms.

8.2.1 Architectural aspect

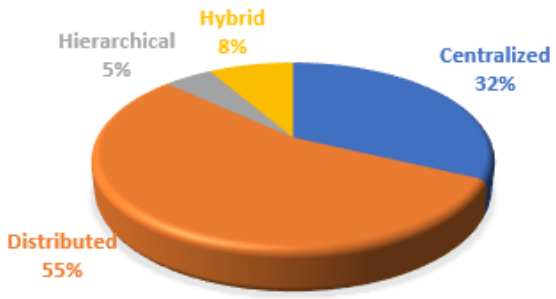

Figure 8: Statistics of the adopted architectural design in the surveyed works.

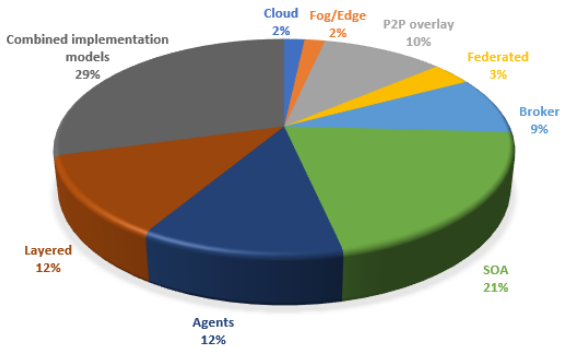

Figure 9: Statistics of the adopted implementation models in the surveyed works.

The *architectural* aspect aims at defining the structure and the implementation of the SD system. We notice that regarding the *architectural design*, as illustrated in Figure. 8, the distributed type is the most widely adopted (55% of the studied articles), due to its efficiency, fault-tolerance, scalability and robustness. Also, it overcomes the limitations of the centralized type which was adopted by 32% out of the surveyed approaches. The hierarchical and the hybrid architectural designs are the less considered with a percentage of 5% and 8% respectively. Concerning the *implementation model*, as shown in Figure. 9, we remark that the SOA model is so far the most commonly adopted, since it can reduce the complexity of the device discovery process and allows objects to inter-operate in a uniform and general manner by exposing their features as services, being thereby discoverable, composable and reusable. Moreover, we observed that the implementation models considered in the different solutions are highly connected with the environment and the context of the application. For instance, the use of a broker-based model is suitable for heterogeneous environments because it allows multiple remote objects to interact (synchronously or asynchronously) with a location transparency. The layered model is more recommended for big applications because it provides modularity, flexibility, changeability and so forth. These two models have been used by 9% and 12% of the reviewed approaches respectively. Moreover, the CC paradigm is the less used (2% out of the discussed solutions) when designing the global IoT infrastructures as it is essentially based on a centralized architecture, so a single point of failure exposed to security risks. A few of the reviewed works (2%) have adopted Fog/Edge Computing paradigms to improve latency, mobility, location-awareness, rapid scalability, and minimizes network traffic overhead. P2P networks have attracted the attention of 10% of the researchers because they support scalability, reliability, mobility and dynamicity. They are robust and efficient since they rely on a distributed architectural design, so no central control. To bring dynamicity, flexibility, autonomy and intelligence to the SD system, the agent-based model has been used by 12% of the surveyed works. Furthermore, only 3% have adopted the federated architecture to simultaneously querying multiple sources and quickly obtaining relevant results, thus ensuring scalability, data distribution and an independent control of repositories. Finally, the combination of several implementation models is considered in 29% of the surveyed papers, mainly to increase the benefits and the performances of the SD systems at the expenses of the design complexity and the maintainability costs.

8.2.2 Descriptive aspect

Description methods represent the way an IoT service/object profile is described by capturing its functional and non-functional characteristics in a machine-interpretable manner. Following our investigations, we found out that, despite their limitations, the syntactic-based approaches are so far the most frequently adopted (43% of the studied papers), because they are easy to understand and to manage, with 30% out of these approaches are records-based, 12% are XML-based, and only 2% have considered JSON as a description method (see Figure. 10). However such methods are less adopted in recent papers, which rather promote other approaches. To better cope with IoT requirements, semantic-based techniques are gaining more interest mainly to provide an explicit, an easy and a comprehensible description (31% out of these articles) either by using ontologies (28%) or RDF (2%). Moreover, resource-based methods are considered by 14% of the studied solutions (9% CoAP and 5% REST) to improve evolutivity, agility, reusablity, etc. Likewise, (12%) of the surveyed articles, mainly the very recent ones, are considering the combination of more than one technique to enrich the service description and to provide more flexibility in the SD. However, no standardized description method has been defined up to now, and the heterogeneity in the way IoT resources are described remains the norm, so far.

Figure 10: Statistics of the adopted description methods in the surveyed works.

8.2.3 Discovery and selection aspect

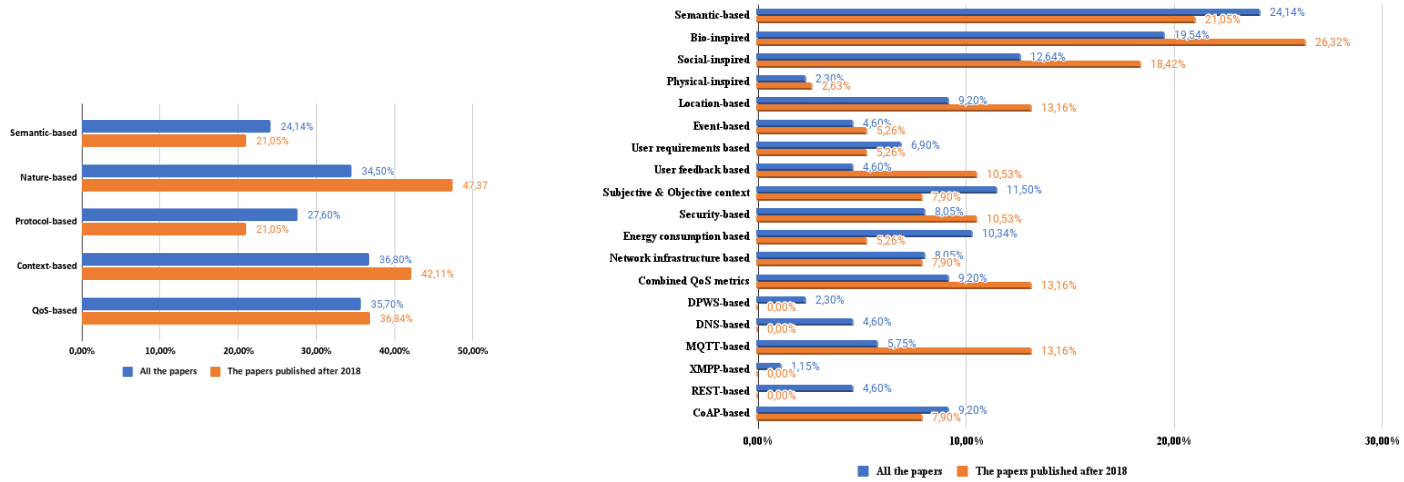

Figure 11: Statistics of the adopted SD and selection algorithms in the surveyed works.

Regarding *service discovery and selection algorithms*, all the existing approaches are designed to deal with specific network requirements by considering different conceptual and computational models. Obviously, none of the reviewed approaches can be promoted as the ideal one, that could satisfy all the user needs and be deployed under any network constraints. Indeed, each network designer has to consider the class of solutions that fits the best with its network environment and the requirements called to be satisfied, according to the type of data provided to the SD application (Semantic, QoS, QoE, etc), and the type of technology used to deploy the solution (type of protocol). It is noteworthy that all the existing approaches are intrinsically hybrid as they can be put in different classes following the paradigms they use and the perspective view of the reader. As shown in Figure. 11, 34% of the surveyed articles are using nature-based concepts, while 36% can be considered as QoS-based. Besides, 28% of the solutions are protocol-based, whereas 37% can be categorized as context-based. Finally, semantic mechanisms are adopted in (24%) of the reviewed papers. However, if we restrict the statistics to the papers published since 2018, we observe that protocols and semantics based solutions are loosing much terrain (with around 21% each) to the benefit of respectively nature based with 47% which are leading the poll, followed by context based with 42% and then QoS based with 37%.

- **Semantic approaches:** Many existing SD and selection algorithms (around 24% of all the studied papers and 21% since 2018) are integrating semantic information provided from different levels (User profiles, Objects, Services, Network), so that to be able to efficiently and accurately respond to all the system requirements. Semantic-based approaches have been introduced to support efficient pre-processing and querying management to produce accurate and precise search results which can be modeled in natural language. They effectively manage data and services heterogeneity to ensure interoperability. For example, the approaches [41][45][86][39] have integrated semantic data capturing context-awareness and user preferences aspects. In other respects, the solutions defined in [41][66][86][68] use semantics to improve the accuracy and the precision of the services. However, using semantic based algorithms requires to transform the raw IoT data into a compatible format so that it can be exploited by the semantic core system. Such a process incurs additional costs, like delay and traffic overhead that should be taken into account when dealing with constrained networks. Moreover, using a semantic search engine calls the design of a middleware, thus yielding performance issues due to the huge amount of data to process. It is noteworthy that so far there exist no harmonisation nor a standardization of semantic ontologies, as each SD algorithm is considering its own solution and design. Therefore, to enable interoperability, the big challenge is to remove the heterogeneity on ontologies across different domains. This is the reason why semantics based approaches are not attracting more interest in recent published SD solutions.

  Moreover, according to our study, we found out that this category of approaches is essentially based on the adoption of

semantic or hybrid description methods, relying mainly on the use of ontologies with almost 71.5% of the papers. Regarding the architectural design and the implementation model, there is a no likely preference.

- **QoS-based SD approaches:** They are considered when it comes to improve the quality, the availability and the reliability of the delivery and the deployment of the required services. Thus, to meet some IoT requirements, they consider a number of QoS parameters. For example, the approaches [58], [135], [114], [134], [88], [140] and [59] were designed to minimize the energy consumption and to extend the network lifetime, which thereby improves the service availability of energy-constrained objects. Other solutions [77], [136], [105], [113], [55] and [70] aim to optimize the infrastructure management of constrained networks, in order to improve data routing and minimize the network overhead. Security-based approaches are considered to provide an enhanced security when processing SD, by mainly guaranteeing privacy, trust, authentication, and data integrity [67], [75], [74], [95], [133] [81]. The combined QoS metrics based methods can be seen as hybrid solutions that aggregate different QoS metrics measuring different network constraints and user requirements, to promote the optimal services. Several works in the literature can be identified within this class [137], [56], [57], [60], [80], [138]. According to our review, among all the surveyed papers, the most commonly used QoS parameter is the energy consumption with a percentage of 10.34%. Then, the approaches that combine multiple QoS metrics are considered by 9.20% of the reviewed papers. The less investigated paradigms within this class are: the security aspect and those which are related to the network infrastructure (around 8%, each one) (see Figure. 11). However, if we restrict the study to papers published since 2018 we found out that security aspects and combining QoS metrics are attracting more interest with 11% and 13% respectively, while the energy based is loosing ground with only 5%. Moreover, it should be noticed that 48.5% of the surveyed solutions pertaining to the QoS class did not adopt or mention any description method, and for the rest, we noted that the records-based description method is the most widely adopted in this class (around 29%). As far as the architectural design is concerned, we observed that the distributed architecture is the most frequently used (45.16% out of the QoS class papers). Regarding the implementation model, SOA, P2P overlay and the combined models are the most applied within this class with percentages of 25.8%, 12.9% and 12.9% respectively. Note that the most combined implementation models are mainly the cloud and the Fog/Edge computing paradigms.
- **Nature inspired approaches:** NIC has been considered in many SD algorithms and is still extensively researched, especially when it comes to deal with complex and conflicting requirements in big data. The success of nature-inspired approaches is due to their significant ability to imitate the best features of nature that evolved by natural selection over millions of years. These solutions have been attracting considerable attention for their good performances, and for the ability to adapt and scale in response to the unpredictable load patterns of the distributed IoT applications. In the IoT environment, there exist many common significant challenges to be addressed, such as: the increased complexity with large scale networks; the dynamic nature of IoT; the network resource constraints; the heterogeneous architectures; the absence or the impracticality of the centralized control and infrastructure; the need for survivability; and the unattended resolution of potential failures. These issues can be successfully dealt by using nature-inspired approaches for many reasons, including: self-adaptivity to varying environmental conditions; inherent resilience to failures and damages caused by internal or external factors. Furthermore, NIC provides self-organization in a fully distributed fashion, survivability despite harsh environmental conditions due to its inherent and sufficient redundancy, scalability, effective management of constrained resources, and finally the ability to achieve complex behaviors on the basis of a usually limited set of basic rules. Besides, NIC can be used in QoS-aware service composition because it requires less computation time for large scale problems, especially in dynamic environments. It can be adopted also in unstructured decentralized P2P networks to find data (by using ant-based approaches).
  According to our study, nature-inspired techniques are more and more considered in the SD solutions (35% of all surveyed papers and 47% since 2018). They are mainly adopting a distributed architectural design with almost 63.34% out of the papers within this class. However, no specific description method nor an implementation model, are favoured. Biological systems are the most considered within the NIC class and they are gaining more interest (19.54% out of all the surveyed articles, and 26% since 2018) and serve as the source of the intelligence of nature-inspired approaches. They have been adopted in many solutions to develop decentralized SD systems and to efficiently manage network resources by enabling self-organization and self-adaptivity. Different biological computing models have been considered, as for instance: ANN [47], insect colonies [89], ACA [118], AFSA [48], the evolutionary game theory [134], etc. Regarding social networks, they are gaining more ground in the design of SD and selection algorithms (almost 13% out of all the studied papers, and 18% since 2018), mainly to improve the trustworthiness, the dynamicity, the autonomy and the scalability of the system. The social approach to the IoT is perceived by many as the solution to meet the needs of users, designers, and developers so that to tackle difficulties that the IoT is raising. In recent years, SIoT has emerged as a new paradigm of IoT that gains many success by enabling objects to have their own social networks; and humans to impose rules to protect their privacy and only access the result of autonomous inter-object interactions occurring on the objects' social network. In SIoT, devices interact and establish relationships with each other to achieve a common goal, such as energy, utility services, and transportation. SIoT methods make it possible to pass through the heterogeneity of IoT devices, by promoting autonomous services that can collaborate on behalf of their owners. The SIoT design concept was considered in several works mainly to enable interoperability by building scalable communities of services [91], [43], [69], [94], [98], [42] [54]. Besides, this model has yielded good results when used to promote trust-management and network navigability, as in [99]. Besides, it has been combined with a great success with semantic approaches in [42], [43] and with RESTful mechanisms in [54] to locate objects and services for composition purpose. However, many security attacks can affect the SIoT system, including: Bad-mouthing, Slandering attack, Self-promoting, Whitewashing, On–Off Selective, Opportunistic service, Discriminatory and Ballot stuffing. In addition, SIoT systems may face unauthorized access and suspicious behavior due to a lack of authentication infrastructure. Therefore, there appears an increase in the need to design efficient mechanisms to verify the authenticity and the confidentiality of the information in a unauthorized access to the data and network resources. This allows predicting the behavior of objects and separating malicious objects from reliable ones, for example, by using machine learning, decision tree and deep learning approaches. Moreover, due to the existence of

multiple data sources from a large number of IoT objects, data ownership is a concern and needs to be addressed. In addition, SIoT suffers from traffic congestion issues due to the large scale of SIoT.

As concerns physical inspired models, they appear to be the less researched in the design of SD and selection algorithms, mainly because they are more complex and less intuitive. Indeed, they are considered only in two works all published after 2018 (2.3% of the studied papers) mainly to improve the QoS of the network and to achieve a load balancing [90][79].

- **Protocol based approaches:** Several protocols offering SD capabilities have been proposed as standards for the IoT. These protocols aim to meet the requirements of resource-constrained devices and allow them to communicate with each other and to automatically discover services without human intervention. Besides, they endeavor to handle dynamic scenarios by providing components that guarantee an effective management of mobile services. Moreover, these protocols have to overcome challenges faced in the IoT environments, for instance: latency, power consumption, interoperability, semantic-awareness, asynchronous notification, standardization and so forth. However, no one of the existing protocols can fit all the IoT scenarios due to the varying requirements and the supported features. Hence, the choice of the discovery protocol to consider depends mainly on the intended running applications. Therefore, it is essential to study these protocols and identify their features in order to opt for the one that better fits the requirements of the SD system. In our taxonomy, discovery protocols are classified into three categories: those oriented services (25% out of the papers within this class); those based on messages (25%), and finally, resources based which are the most considered (50%) among the protocols based approaches.

  Regarding service oriented protocols, DNS-based solutions were proposed to provide an autonomous discovery systems requiring a minimal human intervention. They use mDNS-SD to announce new services and DNS-SD to attend user queries. These two protocols operate in infrastructure-less environments while being resilient to network failures. In addition, DNS has the advantage of wide adoption in traditional networks, which has the potential to offer immediate integration. Among these solutions, the ones presented in [93],[104], [64] and [65] saved the energy, improved the scalability and decreased the overhead. Moreover, DNS-SD/mDNS support the context information in the service description using the TXT records as the work presented in [64]. This information can be used after the service resolution, which implies human intervention after SD. However, the main drawback of using such solutions is the need for caching DNS entries especially for resource-constrained devices. This issue can be solved by timing the cache for a specific interval [21]. Also, they don't provide any security capabilities and do not integrate the QoS information in the discovery process. This is why they are less adopted nowadays in the context of IoT SD systems. DPWS protocol which is the other identified service-oriented protocol, appears to be suitable for resource-constrained devices. It supports eventing mechanisms and improves dynamicity and flexibility. However, it introduces a considerable overhead and an increased latency due to the use of SOAP, and has a limited network range. To deal with these issues, the works in [87] and [102] leverage the use of REST with DPWS to achieve interoperability and decrease overhead and latency. As concerns resource-oriented protocols, the REST protocol has a good interoperability and it is supported by all commercial M2M cloud platforms. It can be easily implemented in mobile or any embedded device and in any language as it uses HTTP library. However, REST supports request/response model which is not suitable for the IoT environment and relies on HTTP protocol which is stateless (i.e., one client request at time, then the connection is closed). So, it has to setup and tear-down the connection each time, and that can take up a ton of data. In addition, HTTP protocol is not lightweight because there's a lot of headers and encoding per request. Unlike REST, the CoAP protocol which becomes the leading solution, has a very light and simple packet structure with binary data representation. Thus, it has a lower overhead as well as a reduced bandwidth requirements and parsing complexity. Besides, it provides multicast support, reliability, security, and simplicity for constrained environments. However, the use of DTLS implies more overhead, bandwidth and energy consumption in the communication. Moreover, there is a complexity for mapping protocols and a lack of appropriate key management mechanisms for the support of secure CoAP multicast communications. Also, CoAP is still maturing (few existing libraries and solution supports).

  Regarding message-oriented protocols, the MQTT protocol was designed for constrained devices and networks and can be used in low-bandwidth, high latency or unreliable networks. It is extremely lightweight, ideal for low energy consumption and easy to implement. However, MQTT clients have their connections to brokers remain open all the time, and require the use of TCP/IP which induces much packets loss. Besides, the resource topic names often require lengthy strings which makes MQTT practically inoperative for IEEE 802.15.4. To overcome these limitations, MQTT-SN was defined to run over UDP and to consider additional features to the broker for indexing the resource topics. Thus, the header complexity is reduced by replacing topic strings by topic IDs to identify content. MQTT-SN still supports all QoS levels but does not inherit any reliability property from the transport layer. The XMPP protocol, which is the alternative solution to MQTT, can be used as an application layer protocol for IoT to achieve real-time application transmission. It has a good scalability, and is highly extensible. Moreover, this protocol offers a rich variety of open source software for servers, clients, and libraries that support several operating systems, thus reducing development costs. However, using streaming XML induces a considerable network traffic overhead, and high memory, CPU, and bandwidth usage. Although TLS/SSL security makes the XMPP standard reliable to use with TCP mechanisms, it does not support QoS options which complicate IoT communications and makes it impractical. Also, the use of TCP may cause some overhead. According to [141], the solutions that are based on CoAP, XMPP and MQTT protocols achieve a better interoperability with other protocols.

  According to our review, the most protocols considered within this class are CoAP with 33.34%, MQTT with 20.84% and REST with 16.67% while DNS, DPWS and XMPP are the less adopted. However, if we restrict the statistics to papers published after 2018, we find out that protocols oriented service (DPWS, DNS), are no longer adopted (0%) to the profit of other categories. Particularly, MQTT is the only protocol adopted in messages oriented solutions and becomes the leading solution with 62% of the papers within the protocol based class. The other dominant solution is COAP (38% of protocol based solutions) which has completely replaced REST (0%) when it comes to adopt resource oriented protocols.

Moreover, the DNS protocol relies essentially on the use of record-based description method and favours a distributed architectural design, without referring to any specific implementation model. On the other hand, DPWS-based techniques essentially adopt an XML-based description method, a centralized architectural design and SOA as an implementation model. In addition, solutions based on the MQTT protocol dot not favour any description method, but require the use of a broker and the adoption of either a centralized architecture or a distributed one. Regarding the CoAP-based techniques, we noticed that 75% have opted for a resource-based description method, while the rest (25%) integrates semantic information through the use of ontologies with these methods (a hybrid approach). Besides, 50% out of the COAP solutions adopt a distributed approach and the remaining has considered either a centralized or a hybrid architecture, 25% each, without favouring any specific implementation model. Finally, REST and XMPP protocols use mainly REST-based and XML-based description methods respectively, without a likely preference on an architectural design or an implementation model.

- **Context based approaches:** One of the most important requirements to guarantee for an IoT SD system is the context-awareness that lies on the ability of a system to provide relevant services to users based on their surrounding environment (Objective context), preferences and feedbacks (Subjective context). It can be used for recommendation systems as well as applications that need to control, choose, and activate an object depending on certain status, environmental and other desired requirements. Discovery solutions should consider the context-awareness for many reasons. First, by acquiring, analyzing, and interpreting relevant context information, services and their provision can become smarter. Second, it ensures the dynamicity of IoT services and enables user-centric approaches. Also, these solutions aim to better meet the needs and expectations of the users by continually evaluating and analyzing the current context. However, most of the existing works don't handle all the context attributes. Also, context acquisition, modeling and reasoning remain a significant challenge to overcome. In addition, the Quality of Context, which denotes a set of parameters used to describe the quality of certain context information elements, is not well investigated. Besides, semantic interoperability across all software architectures is required and a seamless integration of existing computational artifacts (hardware and software) and communication infrastructures is demanded in order to successfully share the context information between highly adaptive services across heterogeneous devices on large-scale networks that consider this information relevant for their purposes. One can see that the QoS-based approaches and the context-based ones are unquestionably related. The key difference between them is that the QoS-based solutions focus on the overall performances of the discovery system and their delivered services. They aim to select the most suitable service, among similar ones, based on its non-functional parameters: delay, availability, bandwidth, energy consumption, etc. without considering the users' perception. While the context-awareness enables the system to adapt its behavior by collecting information about the user, its state and social situation as well as its surrounding environment, at any time. Therefore, these two classes complete each other to ensure the user satisfaction and especially to improve the QoE. Regarding objective solutions, the location-based techniques were introduced to improve mobility, dynamicity and context-awareness. They are strongly coupled with query routing mechanisms and support indexing and ranking methods which reduce the search space and hence the response time like in [112, 121]. However, these approaches utilize large storage when indexes are built and the processing of queries that require information about remote locations take more time to resolve. Also, they may suffer from identification, naming and security problems. Among these works, those which improve the scalability and interoperability requirements [112, 121, 49] and those which solve some security issues, such as: privacy [76] and trust [121]. On the other hand, in event-based solutions, service' triggers are events that occur in a dynamic environment, thereby enhancing the autonomy, the dynamicity and the context-awareness. These solutions were introduced to handle millions of events at once and act in real-time. They adopt a publish/subscribe model. However, if the broker fails or gets overloaded, the overall system may downfall or suffer from performance issues. Moreover, the QoS can be easily affected by the environment such as: the number of connected users and the mobility of services.

  Subjective solutions, as for instance, user feedback-based techniques [127][92][128] were introduced to accurately represent the performances of a service and predict the execution plan since a user under the same environment can easily get a similar experience for a particular service. These solutions use reputation mechanisms which consist in processing previous user feedbacks (i.e., a set of historical interactions) to select the most appropriate services. They also consider user experience, comments and feedback to increase the trust value. Both of user requirement-based and user feedback-based approaches aim at ensuring the user satisfaction, improving QoE, reducing the search space and thus the response time.

  Finally, the hybrid class attempt to combine objective and subjective context attributes to provide an efficient and a real-time SD and selection process. It improves the dynamicity, the scalability, the context-awareness and thus the QoE.

  According to our review, context based solutions are gaining more interest the last years (42% of the surveyed solutions since 2018). Particularly, objective context solutions are the most investigated with 38% (44% since 2018) out of the context-based solutions. Within this class, the location-based are attracting the interest of researches with 25% in overall to 31% since 2018. The event-based solutions have been considered in 12.5% of context based papers with a no significant development since 2018. On the other hand, subjective context based approaches are gaining ground moving from 31% in overall to 36% since 2018, mainly because of user feedback based approaches which are gaining a huge interest moving from 12.5% in overall to 25% since 2018. However, user requirements based solutions are less investigated moving from 18.75% to 12.5% since 2018. Finally, hybrid solutions are paying the price as they have been less researched since 2018, with 19% against 31% in overall.

  Regarding the descriptive aspect, 37.5% out of the selected context-based solutions did not adopt or mention any description method, while 25% have used ontologies. As regards the architectural design, the distributed architecture is favoured (around 53%), then, comes the centralized architecture with 37% and the remaining papers used a hybrid approach (almost 9.5%). We did not notice any likely preference for the implementation model.

Table 4 provides a full comparison of the all surveyed works according to our classification, and by associating with each approach the considered description method, architecture and the implementation model.

Table 4: Comparison of the reviewed works.

Main paradigm			App	Description method	Discovery and selection algorithms	Architectural design	Implementation model
Semantic			[38]	OWL-S	Semantic gateways, dynamic clustering of services, semantic similarity measurement	Hierarchical	SOA
			[39]	WSDL	semantic similarity measurement, clustering mechanism, NGD metric	Centralized	Combined (SOA, Broker)
			[117]	Ontology	semantic similarity measurement, density-peaks-based clustering	Centralized	/
			[40]	extended OWL-S	ForwarDS-IoT, Federated search	Distributed	Federated
			[45]	extended OWL-S	QoDisco, Federated search	Distributed	Federated
			[68]	RDF	eWoT, SPARQL	Distributed	/
			[66]	Simple records	ICN, calculating Semantic similarity and the distributional profiles via SME	Centralized	Broker
			[86]	OWL-S	A hybrid algorithm for service matchmaking combining syntactic and ontology similarities	Centralized	SOA
			[41]	OWL-SIoT	Multi-stage semantic service matching, a hybrid service matching degree and semantic similarity measurement	Centralized	layered
Nature	Bio		[47]	XML-based	Model based on ANN	Centralized	Layered
			[89]	/	Model inspired from RTM	Distributed	Combined (Agents, SOA)
			[118]	/	Model based on ACA	Distributed	Agents
			[48]	/	Model based on AFSA	Distributed	/
	Social		[42]	Semantic	Swarm intelligence search strategy, homophily-based user model	Distributed	P2P overlay
			[80]	Ontology	a social recommendation-based model and a decentralized locality-preserving model, signature matching, semantic similarity, SkipNet-based overlay structure	Distributed	P2P overlay
			[91]	/	Objects create their own relationships based on rules defined by the owner, SIoT, object profiling	Distributed	/
			[43]	OWL	Correlation degree, semantic similarity and relativity, Social network among devices based on restricted contact graph	Distributed	/
			[94]	/	community detection algo based on preference, social and movement similarities. Intra/inter-community search	Distributed	layered
			[69]	/	SIoT, Object profiling	Distributed	/
			[54]	Hybrid (XML, REST)	three searching algos based on 3-dimensions: location, type, correlation	Distributed	Combined (Layered, Broker, SOA)
			[99]	/	Graph-based collaborative filtering recommendation system, similarity measurement, predicted rating, SIoT	Distributed	/
			[98]	/	Community detection algos (Louvain and OSLOM), SIoT, NLP	Distributed	/
	Physical		[90]	/	Model based on APFs	Distributed	Agents
			[79]	/	evolutionary game theory combined with evaporation-based water cycle algorithm	Hierarchical	Combined (fog-cloud)
Context	Objective context	Location	[116]	Hybrid (JSON-LD)	SPARQL, swarm network, semantic matching score	Distributed	Combined (SOA, Broker)
			[76]	XML-based (PML)	Model based on user's location and preferences, ONS/PML	Centralized	combined(SOA, Cloud)
			[112]	Hybrid (OWL-S, REST)	Geospatial indexing, SPARQL, semantic search and matching	Distributed	/
			[121]	/	Services recommendation method by defining a location-based service preference similarity based on space-time characteristics and direct and indirect friendships	Distributed	/
			[49]	/	spatially-cohesive service selection mechanism based on RL	Distributed	Agents
		Event	[50]	/	publish/subscribe, grouping services, selection model based on user specification and QoS	Centralized	SOA
			[101]	WSDL	MQTT, Broker	Centralized	Combined (Broker, SOA)
			[44]	OWL-SE	semantic service matching based on word embeddings	Centralized	SOA
			[123]	/	publish/subscribe	Distributed	Combined (SOA, Broker)
	Subjective context	User requirements	[61]	Ontology	Context-aware discovery based on the collected context information, uncertainty and temporal aspects of context, Dynamic Bayesian networks	Centralized	/
			[51]	Records	user and service profiles, matching score function, semantic similarity	Hybrid	Combined (Broker, layered)
			[125]	Ontology	PSS, user preferences, absolute dominance relationship	Distributed	/
			[126]	Ontology	Model based on user situation and preferences	Distributed	Layered
			[62]	Ontology	A progressive search algo for Semantic similarity and proximity measures	Centralized	/
		User feedback	[127]	Hybrid	Service composition model based on consumers' feedback for planning algorithm with minimal human intervention	Hybrid	/
			[92]	/	LRI algo considering the similarity between users	Distributed	agents
			[128]	/	Model based on Likert scales measurement with Improved-PSO	Distributed	SOA
	Subjective and objective context		[129]	Ontology	FQSA, ANN-BP, similarity aggregation method, QoS ontology, user feedback and preferences	Centralized	SOA
			[63]	/	ANN-BP and TOPSIS	Hybrid	/
			[52]	Records	TLF_SSA based on the weighting model of historical records	Centralized	/

Continued on next page

Table 4: Comparison of the reviewed works.

Main paradigm			App	Description method	Discovery and selection algorithms	Architectural design	Implementation model
			[130]	Records	NCF and Fuzzy clustering algo, new Euclidean distance calculation	Distributed	Layered
QoS	Security		[95]	REST	Discovery layer, search engine, indexing API	Centralized	Layered
			[74]	Records	A probabilistic security schemes to detect suspicious SD lookup queries in the EPCglobal network, ONS, DHT	Hybrid	P2P overlay
			[75]	Records	Web-based IoT service management system	Centralized	Cloud
			[67]	XML	Genetic and PSO matching algorithms, Services profiles repository, session key generator, certificate/encrypt	Centralized	Broker
			[133]	Records	Kademlia DHT	Distributed	P2P overlay
			[81]	/	DHT, trust management model, user feedback	Distributed	P2P overlay
	Energy Consumption		[140]	/	two-stage algo "dual", energy sentient methodology, ILP	Distributed	SOA
			[134]	/	a dynamic evolutionary game, evolutionary equilibrium	Centralized	SOA
			[58]	Records	lexicographic optimization strategy and QoS constraints relaxation technique, relative dominance of services in the sense of Pareto	/	/
			[135]	/	Energy, offloading and fault-tolerance models, XML charts	Distributed	/
			[114]	CoAP	Adaptive Fibonacci-based Tuning Protocol, Fibonacci sequence, CoAP	Centralized	Agents
			[59]	/	MOEA/D	/	/
			[88]	Records	Agents	Distributed	Agents
	Network infrastructure		[136]	/	Multicast forwarding mechanism, zero-config, bloom filters	Distributed	/
			[105]	Records	Bloom filters	Distributed	/
			[113]	REST	unicast, anycast, multicast and broadcast service invocations using HTTP or CoAP	Distributed	Combined (Cloud, Broker)
			[55]	Records	PSO, ACO and GA, energy model	Distributed	SOA
			[70]	Records	Modal based on Forwarding Information Base and round trip time	Hierarchical	Combined (Cloud, Edge)
			[77]	/	Fog node selection based on randomized, greedy, kMedian,k- Means and simulated annealing techniques	Distributed	Fog
	Combined QoS metrics		[137]	Ontology	CASSARAM, SPARQL, Context information	Distributed	Broker
			[56]	/	Pulse algo, energy profile model, a Pareto-optimal solution for QoS and energy-aware IoT composition based on user preferences	Centralized	SOA
			[57]	/	ANN-PSO	hybrid	Combined (Cloud-Edge)
			[60]	/	Fuzzy and TOPSIS, OWA operator	/	/
			[138]	/	AHP and TOPSIS	Hierarchical	/
Protocol	Service	DPWS	[87]	Hybrid (DPWS, REST)	DPWS + REST	Centralized	SOA
			[102]	DPWS	DPWS, REST proxy	Centralized	SOA
		DNS	[104]	DNS-records	mDNS/DNS-SD, zero-conf	Distributed	/
			[93]	DNS-records	EADP/DNS-SD	Distributed	Agents
			[64]	DNS-records	mDNS/DNS-SD extended to support context	Distributed	/
			[65]	Simple records	mDNS/DNS-SD	Distributed	/
	Message	MQTT	[78]	/	MQTT	Distributed	Combined (Fog, Cloud, Broker)
			[85]	Ontology	NETCONF-MQTT	Centralized	Broker
			[83]	Simple records	MQTT, auto-configuration	Centralized	Broker
			[84]	JSON	MQTT, zero-configuration	Distributed	Combined (Cloud, Broker)
		XMPP	[100]	XML-based	XMPP, publish/subscribe	Centralized	/
	Resource	REST	[111]	RAML	Matching between user request and flow templates Syntax-based searching methods	Centralized	Combined (SOA, layered)
		CoAP	[82]	CoAP	CoAP, zero-conf local SD	Distributed	P2P overlay
			[97]	CoAP	CoAP, CoRE/RD interface	Distributed	Combined (Fog, P2P overlay)
			[46]	Hybrid (CoAP + OWL)	CoAP, SPARQL	Centralized	/
			[115]	CoAP	FETCHIoT, CoAP	Hybrid	/
			[73]	CoAP	CoAP, LOAMY	hybrid	Combined (Cloud, Broker)
			[96]	Hybrid (CoAP + ontologies)	SPARQL, semantic group composition based on social-awareness on COAP	Distributed	layered

/ : Not available

8.3 QoS metrics evaluation and performance comparison

To evaluate the performances of the surveyed solutions, we further conducted an analysis based on the metrics stated in section 8.1 (see Table 5). It appears thereof that none of them has dealt with all the requirements, although, a considerable effort has been put in designing the different solutions while minimizing cost and ensuring a number of IoT requirements. This assessment is based on objective and subjective statements made in the 87 reviewed papers. Clearly, we fist identified objectively all the metrics that have been evaluated during the experiments of each solution. For the remaining aspects, we subjectively evaluate them depending, for ones, on the targeted goals by each solution, and for the others, according to some criteria, as for instance, the type of the SD algorithm, the type of architecture design, the implementation model as well as the description method adopted in the solution.

Figure. 12 synthesises this study. Hence, we notice that scalability, interoperability, as well as context-awareness and user preferences are the most improved metrics by the surveyed solutions with a percentage of 67%, 49% and 54% respectively. Other researchers were interested in improving the dynamicity (33%) and the autonomy (39%), and in reducing the delay (38%). However,

the remaining metrics, such as: mobility, security, user feedback, availability, etc. are the less researched in the surveyed papers.

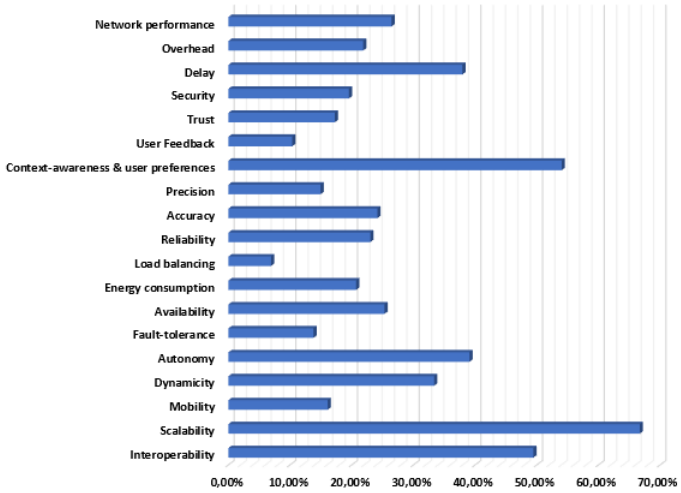

Figure 12: Statistics of the improved metrics in the surveyed works.

By refining the statistics to each class, we notice that semantic-based algorithms improve considerably the interoperability (90.48% of the solutions within this class), scalability (66.67%), accuracy (42.9%) and precision (47.62%). Besides, 61.9% of semantic based solutions integrates the context information and the user preferences by using ontologies.

Regarding the bio-inspired approaches, they significantly improve the scalability (58.82% of the papers within this category), the reliability (52.94%) and the accuracy (45.06%), while the social-inspired solutions are scalable and aim at enhancing the interoperability (73%), the dynamicity (73%), the autonomy (73%) as well as the trust value (72.73%). For the physical-inspired techniques, the availability, the delay and the load balancing are improved, thus ensuring scalability.

As concerns the security-based solutions, they mainly improve the security, and the trust (57.14%), while the energy consumption based ones aim at reducing the energy consumption value and thus enhancing the availability (almost 89%). Regarding the network infrastructure-based techniques, they mainly increase the network performances by optimizing the delay parameter (57.14%), enabling scalability (57.14%) and handling the mobility issue (almost 72%). Moreover, the combined QoS selection algorithms tend to integrate the context information (62.5%) and several QoS parameters to improve the scalability (75%), the availability (50%) and to reduce the delay (50%).

Regarding protocols based solutions, we find out that DPWS-based methods are interoperable, scalable, dynamic and support the context and event awareness, while the DNS-based ones are scalable (75%), autonomous and enjoy less overhead. Concerning the MQTT-based approaches, they are interoperable, autonomous and reduce the overhead (60%). For the CoAP-based algorithms, they are interoperable and scalable (62.5% each) as well as allows reducing the delay (50%) and enhancing the network performances (37.5%). Besides, XMPP and REST protocols tend to improve the interoperability and the scalability. Finally, location-based approaches are interoperable, scalable, dynamic (75% each) and support mobility (37.5%) as well as context-awareness.

As regards the event-based methods, they are dynamic, autonomous and context-aware. Regarding the subjective context based solutions, they are based on either user preferences or feedbacks which can improve the trust value (40%), and reduce the search space and thus the response time (50%). Finally, the hybrid context based methods use several context attributes which can ensure some QoS metrics such as: scalability (60%), dynamicity (40%), reliability (60%), accuracy (40%), delay (50%).

Table. 6 summarizes the findings of our study by providing synthetic information about the different classes and paradigms considered in our taxonomy together with the percentages of the adoption of each aspect relatively to the surveyed works.

Accordingly, the continuing trend is toward using nature inspired and QoS, as well as context algorithms in the SD. Semantics aspects may gain more interest if the issues related to ontology design and homogenisation are resolved. As concerns the use of protocols, it appears that trend is negative, and only MQTT and COAP are still attracting interest. As regards the architectural aspect, the distributed design is the most researched with a preference to use SOA or to combine different implementation models. However, it appears that the big issue in designing an SD and selection system is the lack of a well accepted standard for resource description. Indeed, due to the lack of the latter, existing solutions are forced to adopt obsolete methods like syntactic ones for their simplicity, which however do not fit systematically all the IoT requirements.

Table 5: Comparison of the reviewed works according to QoS metrics and IoT requirements.

Category	App	IoT requirements & QoS metrics																	
		Interoperability	Scalability	Mobility	Dynamicity	Autonomy	Fault tolerance	Availability	Energy consumption	Load balancing	Reliability	Accuracy	Precision	Context	User feedback	Trust	Security	Delay	Overhead	Network performance
Semantic	[38]	↑	↑	✓	↑	–	–	–	–	–	–	–	–	–	–	–	–	–	↓	–
	[39]	↑	–	–	–	–	–	–	–	–	–	–	–	✓	–	–	–	–	–	–
	[117]	↑	–	–	–	–	–	–	–	–	–	↑	↑	–	–	–	–	–	–	–
	[40]	↑	↑	–	–	↑	↑	↑	–	–	–	–	–	–	–	–	–	–	–	–
	[45]	↑	↑	–	–	↑	↑	–	–	–	–	–	–	✓	–	–	–	–	↓	–
	[68]	↑	↑	–	–	–	–	–	–	–	–	↑	↑	–	–	–	–	–	–	–
	[66]	↑	–	✓	–	–	–	–	–	–	–	↑	↑	–	–	–	–	↓	↓	–
	[86]	↑	–	–	–	–	–	–	–	–	–	↑	↑	✓	–	–	–	–	–	–
	[41]	↑	↑	–	–	–	–	–	–	–	–	–	↑	✓	–	–	–	↓	–	–
Bio	[47]	↑	↑	–	↑	↑	–	↑	–	–	↑	↑	–	✓	–	–	↑	–	–	–
	[89]	↑	↑	–	↑	↑	↑	–	–	–	–	–	–	–	–	–	–	–	–	↑
	[118]	–	↑	–	↑	↑	↑	–	–	–	↑	↑	–	–	–	–	↑	↓	–	–
	[48]	–	↑	–	↑	↑	↑	–	–	–	–	↑	↑	✓	–	–	↑	↓	–	–
Social	[42]	↑	↑	–	↑	↑	↑	–	–	–	↑	↑	↑	✓	–	↑	–	–	↓	↑
	[80]	↑	↑	–	–	–	–	–	–	–	↑	–	–	✓	✓	↑	↑	↓	–	–
	[91]	↑	↑	–	↑	↑	–	–	–	–	–	–	–	✓	–	↑	–	–	–	–
	[43]	↑	↑	–	↑	↑	–	–	↓	–	↑	↑	–	✓	–	↑	–	–	–	↑
	[94]	–	↑	–	↑	↑	–	–	–	–	↑	–	–	✓	–	–	–	↓	–	–
	[69]	↑	↑	–	↑	↑	–	–	–	–	–	–	–	✓	–	↑	–	↓	–	–
	[54]	↑	↑	–	↑	↑	–	–	–	–	–	–	–	✓	–	–	–	–	–	–
	[99]	–	↑	–	↑	↑	–	–	–	–	–	–	↑	✓	–	↑	–	–	–	–
	[98]	–	↑	–	↑	↑	–	–	–	–	↑	↑	–	✓	–	↑	–	–	–	–
Physical	[90]	–	↑	–	↑	↑	↑	↑	–	✓	–	–	–	–	–	–	–	↓	–	–
	[79]	–	↑	✓	–	–	–	↑	↓	✓	↑	–	–	–	–	–	–	↓	–	↑
Location	[116]	↑	–	–	–	–	–	–	–	–	–	–	–	✓	–	–	–	–	–	–
	[76]	–	–	✓	↑	–	–	–	–	–	–	–	–	✓	–	–	↑	–	–	–
	[112]	↑	↑	–	↑	–	–	–	–	–	–	↑	↑	✓	–	–	–	↓	–	↑
	[121]	↑	↑	✓	–	–	–	–	–	–	–	–	–	✓	–	↑	–	↓	–	–
	[49]	↑	↑	✓	↑	↑	–	–	–	–	–	–	–	✓	–	–	–	–	–	–
Event	[50]	–	↑	–	↑	↑	↑	↑	–	–	↑	–	–	✓	–	–	–	↓	–	–
	[101]	↑	–	–	↑	↑	–	–	–	–	–	–	–	✓	–	–	–	↓	↓	–
	[44]	↑	–	–	↑	↑	–	–	–	–	–	↑	↑	✓	–	–	–	–	–	–
	[123]	↑	↑	–	↑	↑	–	–	–	–	–	↑	–	✓	–	–	–	↓	–	–
User requirements	[61]	↑	–	–	–	–	–	↑	–	–	–	–	–	✓	–	–	–	–	–	–
	[51]	–	↑	–	–	–	–	–	–	–	–	–	↑	✓	–	–	–	↓	↓	↑
	[125]	–	–	–	–	–	–	–	–	–	↑	–	–	✓	–	–	–	–	–	–
	[126]	↑	–	–	–	–	–	–	–	–	–	–	–	✓	–	–	–	–	–	–
	[62]	–	↑	–	–	–	–	–	–	–	–	–	↑	✓	–	–	–	↓	–	–
User feedback	[127]	–	–	–	–	↑	–	–	–	–	↑	↑	↑	–	✓	↑	–	↓	–	–
	[92]	–	↑	–	↑	↑	–	–	–	–	–	–	–	–	✓	↑	–	–	–	–
	[128]	–	↑	–	–	–	–	↑	–	–	↑	–	–	–	✓	↑	–	↓	–	–
Subj & obj context	[129]	–	↑	–	–	–	–	–	–	–	↑	↑	–	✓	✓	–	↑	–	–	–
	[63]	–	↑	–	↑	–	–	–	–	–	↑	↑	–	✓	✓	–	–	↓	–	–
	[52]	–	–	–	–	–	–	–	–	–	–	↑	–	✓	✓	↑	–	–	–	–
	[130]	–	–	–	–	–	–	–	–	–	↑	↑	–	✓	–	–	–	↓	–	–
Security	[95]	↑	–	–	–	–	–	–	–	–	–	–	–	–	–	–	↑	–	–	↑
	[74]	–	↑	–	–	↑	↑	↑	–	–	–	–	–	✓	–	–	↑	–	–	↑
	[75]	–	–	–	–	–	–	–	–	–	–	–	–	✓	–	–	↑	–	–	–
	[67]	–	–	–	–	–	–	–	↑	–	–	–	–	✓	✓	↑	↑	↑	–	↑
	[133]	–	↑	✓	–	–	↑	↑	–	–	–	–	–	–	–	↑	↑	–	–	–
	[81]	–	↑	–	–	–	–	–	–	–	–	↑	–	–	✓	↑	↑	–	↓	↑

Continued on next page

Table 5: Comparison of the reviewed works according to QoS metrics and IoT requirements.

Category	App	IoT requirements & QoS metrics																	
		Interoperability	Scalability	Mobility	Dynamicity	Autonomy	Fault tolerance	Availability	Energy consumption	Load balancing	Reliability	Accuracy	Precision	Context	User feedback	Trust	Security	Delay	Overhead	Network performance
Energy Consumption	[140]	–	↑	–	–	–	–	↑	↓	✓	–	–	–	–	–	–	–	–	–	–
	[134]	–	–	–	–	–	–	↑	↓	✓	–	–	–	–	–	–	–	–	–	↑
	[58]	–	↑	–	–	–	–	↑	↓	–	–	–	–	✓	–	–	–	–	–	–
	[135]	–	↑	–	–	–	–	↑	↓	✓	↑	–	–	–	–	–	–	↓	↓	↑
	[114]	–	–	–	–	–	–	↑	↓	–	–	–	–	–	–	–	–	–	↓	↑
	[59]	–	–	–	–	–	–	↑	↓	✓	↑	–	–	–	–	–	–	–	–	–
	[88]	↑	↑	–	–	↑	–	↑	↓	–	–	–	–	–	–	–	–	–	–	–
Network infrastructure	[136]	↑	–	–	–	↑	–	–	↓	–	–	–	–	–	–	–	–	↓	–	↑
	[105]	–	↑	✓	–	–	–	↑	–	–	–	–	–	–	–	–	–	–	↓	↑
	[113]	↑	–	✓	↑	–	↑	–	–	–	–	–	–	–	–	–	–	–	–	↑
	[55]	–	–	–	–	–	–	–	↓	–	–	–	–	✓	–	–	–	–	–	↑
	[70]	–	↑	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	↓	↓	↑
	[77]	–	↑	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	↓	–	↑
Combined QoS metrics	[137]	↑	↑	–	–	–	–	–	–	–	–	–	–	✓	–	–	–	–	–	–
	[56]	–	↑	–	–	–	–	↑	↓	–	–	–	–	–	–	–	–	↓	–	↑
	[57]	–	↑	–	–	↑	–	↑	–	–	–	↑	–	–	–	–	–	↓	–	–
	[60]	–	–	–	–	–	–	–	–	–	–	–	–	✓	–	–	–	–	–	–
	[138]	–	–	–	–	–	–	–	–	–	–	–	–	✓	–	–	–	–	–	–
DPWS	[87]	↑	↑	–	↑	–	–	–	–	–	–	–	–	✓	–	–	↑	–	↓	–
	[102]	↑	↑	–	↑	–	–	–	–	–	–	–	–	✓	–	–	↑	↓	↓	–
DNS	[104]	↑	–	–	–	↑	–	–	–	–	–	–	–	–	–	–	–	–	↓	↑
	[93]	–	↑	–	–	↑	–	–	↓	–	↑	–	–	–	–	–	–	↓	↓	–
	[64]	–	↑	–	–	↑	–	–	–	–	–	–	–	✓	–	–	–	–	↓	–
	[65]	–	↑	–	–	↑	–	–	↓	–	–	–	–	–	–	–	–	–	↓	↑
MQTT	[78]	–	↑	✓	↑	–	–	–	↓	–	–	–	–	✓	–	–	–	–	–	–
	[85]	↑	–	–	–	↑	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	[83]	↑	–	–	–	↑	–	–	↓	–	–	–	–	✓	–	–	–	–	↓	–
	[84]	↑	↑	–	–	↑	–	–	–	–	–	–	–	–	–	–	–	↑	↑	↓
XMPP	[100]	↑	↑	–	–	–	–	↑	↓	–	–	–	–	–	–	–	–	↓	↑	–
REST	[111]	↑	↑	–	–	–	–	↑	–	–	–	–	–	–	–	–	↑	–	–	–
CoAP	[82]	↑	↑	✓	↑	↑	↑	↑	–	–	↑	–	–	–	–	–	↑	–	↓	–
	[97]	–	↑	–	–	–	–	–	–	–	–	–	–	–	–	–	–	↓	–	–
	[46]	↑	↑	–	–	–	–	–	↓	–	–	↑	–	–	–	–	–	↓	–	–
	[115]	–	–	–	–	–	–	–	–	–	↑	–	–	–	–	–	–	↓	–	–
	[73]	↑	↑	✓	↑	–	–	–	–	–	–	–	–	✓	–	–	↑	↑	–	–
	[96]	↑	↑	–	–	–	–	–	–	–	–	–	–	–	–	–	–	↓	–	↑

↑ : Increased ↓ : Decreased – : Not supported ✓ : Supported

Table 6: Summary of the classes and paradigms considered in our taxonomy.

Main paradigm	Sub paradigms		% of papers using the paradigm	Description method	Architectural design	Implementation model	Main Improved Metrics
Semantic			24.14%	Semantic or Hybrid (71%)	No preference	No preference	Interoperability, Scalability, Accuracy, Precision, Context-awareness
QoS	Energy consumption		10.34%	-48.5% do not mention any method -29.03% use records-based methods	Distributed (45.16%)	-SOA 25.8%, -P2Poverlay 12.9%, -Combined 12.9%	Energy consumption, Availability
	Security		8.05%				Security and Trust metrics
	Network Infrastructure		8.05%				Delay, Network performances, Scalability, Mobility
	Combined QoS		9.20 %				Context-awareness, Scalability, Availability, Delay
NIC	Bio		19.54%	No preference	Distributed (63.34%)	No preference	Scalability, Reliability, Accuracy
	Social		12.64%				Interoperability, Scalability, Dynamicity, Autonomy, Trust
	Physical		2.30%				Availability, Delay, Load balancing
Protocol	Service	DNS	4.60%	Record-based	Distributed	No preference	Scalability, autonomy, overhead
		DPWS	2.30%	XML-based	Centralized	SOA	Interoperability, Scalability, Dynamicity, Context-awareness
	Resource	REST	4.60%	REST-based	No preference	No preference	Interoperability, scalability.
		COAP	9.20%	-Resource-based 75% -Hybrid 25%	Distributed 50%, Hybrid 25%, Centralized 25%	No preference	Interoperability, scalability, delay, network performances
	Message	MQTT	5.75%	No preference	Centralized or Distributed	broker	Interoperability, Autonomy Overhead
		XMPP	1.15%	XML-based	No preference	No preference	Interoperability, Scalability.
Context	Objective	Location	9.20%	-37.5% do not mention any method, -25% ontologies	Distributed 53.13%, Centralized 37.5%, Hybrid 9.38%	No preference	Interoperability, Scalability, Dynamicity, Mobility, Context-awareness
		Event	4.60%				Dynamicity, Autonomy, Context-awareness.
	Subjective	User feedback	4.60%				Trust, Delay, User-feedback and preferences
		User requirements	6.90%				
	Hybrid		11.50%				Scalability, Dynamicity, Reliability Accuracy, Delay

9 Conclusion

With the advent of IoT, we observed a proliferation of connected devices distributed over different physical locations, called smart spaces. These objects are thereby capable to expose their functionalities as services of different features and performances, to be selected and invoked. However, IoT is growing fast and millions of Internet-connected objects offering a wide range of services, are being integrated every year. Therefore, discovering and selecting the most appropriate services to be invoked become a challenging issue to overcome.

In this paper, we, first, identified and compared the most recent surveys and taxonomy papers that have dealt with service discovery and selection in the context of IoT. Accordingly, we have proposed a more comprehensive and complete classification with a clear hierarchy by considering the main paradigms used in this process. Then accordingly, we surveyed the most recent and important solutions considering three perspectives: Description methods, SD and selection algorithms, and architectural aspects. We conducted a comparative study of the different classes of the surveyed solutions, considering several aspects and performance metrics. Then, we focused on the advantages and drawbacks of each class presented in the taxonomy to highlight some challenges and future research questions and directives. Some of the future research directions in this area include:

1. The lack of a well established standard to describe IoT resources appears to be one of the the big problems that hinders the development of SD and selection systems. This new standard must remove heterogeneity to promote the interoperability. It should enable the description of all the QoS and the QoE parameters to integrate the IoT requirements. Such a model can inherit some functionalities of existing methods to provide a rich service description framework. It would be interesting to consider an extensible meta-model that integrates all the existing description methods with the possibility to extend its functionalities to new features.
2. It would be interesting to explore solutions that call the deployment of dynamic and adaptable indexing and ranking methods to speed up the query execution time and produce real-time accurate results. Indeed, a distributed multilevel index integrating different context information as location, and energy can improve greatly the performances, as for instance, the delay, the mobility, the energy consumption, the dynamicity, and the context-awareness. A multi-criteria ranking method can be integrated at the lower level of the index to select the best candidate services according to some QoS metrics to improve the accuracy of the SD system.
3. Integrating effective security mechanisms is a big challenge to address in the design of SD systems. We believe that considering the SIoT concept combined with the blockchain paradigm can help to improve access control to objects and ensure privacy, confidentiality, trust management and data integrity.
4. The lack of a standardized architecture for the IoT is sill a pebble in the shoe that limits the development of interoperable IoT systems. However, the trend is towards adopting a distributed architecture in designing SD systems to fit the nature of IoT networks. The combination of different architecture designs as well as adopting a hybrid implementation model can help to improve the performances and to enable the interoperability the scalability, the context awareness, and the autonomy, which are the key performance factors for SD systems.
5. Finally, the SD and selection mechanisms in the IoT environment require further investigation to integrate user preferences and feedbacks to improve the QoE and the QoS. SIoT and deep learning based approaches worth to be further explored for that matter.

References

[1] S. P. Singh et al. "Fog computing: from architecture to edge computing and big data processing". In: *J. Supercomput.* 75.4 (2019), pp. 2070–2105. DOI: 10.1007/s11227-018-2701-2.

[2] Z. Mahmood. *Fog Computing*. 2018. DOI: 10.1007/978-3-319-94890-4.

[3] OpenFog Consortium Architecture Working Group. *OpenFog Reference Architecture for Fog Computing*. 2017.

[4] M. R. Bouakouk, A. Abdelli, and L. Mokdad. "Survey on the Cloud-IoT paradigms: Taxonomy and architectures". In: *IEEE ISCC* (2020).

[5] Y. Hammal et al. "Formal techniques for consistency checking of orchestrations of semantic Web services". In: *J. Comput. Sci.* 44 (2020), p. 101165. DOI: 10.1016/j.jocs.2020.101165.

[6] L. Mokdad, J-M. Fourneau, and A. Abdelli. "Performance Evaluation of the QoS Aware Web Service Composition with Communities of Consumers". In: *2019 IEEE Global Communications Conference, GLOBECOM 2019, Waikoloa, HI, USA, December 9-13, 2019*. IEEE, 2019, pp. 1–6. DOI: 10.1109/GLOBECOM38437.2019.9014302.

[7] A. Abdelli, L. Mokdad, and Y. Hammal. "Dealing with value constraints in decision making using MCDM methods". In: *J. Comput. Sci.* 44 (2020), p. 101154. DOI: 10.1016/j.jocs.2020.101154.

[8] S. Abdellatif, O. Tibermacine, and A. Bachir. "Service discovery in the internet of things: a survey". In: *International Symposium on Modelling and Implementation of Complex Systems*. Springer. 2018, pp. 60–74.

[9] S. Pattar et al. "Searching for the IoT Resources: Fundamentals, Requirements, Comprehensive Review, and Future Directions". In: *IEEE Communications Surveys & Tutorials* 20.3 (2018), pp. 2101–2132. DOI: 10.1109/COMST.2018.2825231.

[10] K. Khalil et al. "Resource discovery techniques in the internet of things: A review". In: *Internet Things* 12 (2020), p. 100293. DOI: 10.1016/j.iot.2020.100293.

[11] M. Aziez, S. Benharzallah, and H. Bennoui. "A full comparison study of service discovery approaches for internet of things". In: *Int. J. Pervasive Comput. Commun.* 15.1 (2019), pp. 30–56. DOI: 10.1108/IJPCC-04-2019-0038.

[12] M. S. Roopa et al. "Social Internet of Things (SIoT): Foundations, thrust areas, systematic review and future directions". In: *Comput. Commun.* 139 (2019), pp. 32–57. DOI: 10.1016/j.comcom.2019.03.009.

[13] M. Achir, A. Abdelli, and L. Mokdad. "A taxonomy of service discovery approaches in IoT". In: *8th International Conference on Wireless Networks and Mobile Communications, WINCOM 2020, Reims, France, October 27-29, 2020*. Ed. by M. Ayaida et al. IEEE, 2020, pp. 1–6. DOI: 10.1109/WINCOM50532.2020.9272512.

[14] A. Huf and F. Siqueira. "Composition of heterogeneous web services: A systematic review". In: *Journal of Network and Computer Applications* 143 (2019), pp. 89–110. DOI: 10.1016/j.jnca.2019.06.008.

[15] A. Chowdhury and S. A. Raut. "A survey study on Internet of Things resource management". In: *Journal of Network and Computer Applications* 120 (2018), pp. 42–60. DOI: 10.1016/j.jnca.2018.07.007.

[16] G. Shinde and H. Olesen. "A survey on service discovery mechanism". In: *Intelligent Computing and Information and Communication*. Springer, 2018, pp. 227–236.

[17] H. Zorgati, R. Ben Djemaa, and I. A. Ben Amor. "Service discovery techniques in Internet of Things: a survey". In: *2019 IEEE International Conference on Systems, Man and Cybernetics, SMC 2019, Bari, Italy, October 6-9, 2019*. IEEE, 2019, pp. 1720–1725. DOI: 10.1109/SMC.2019.8913969.

[18] Y. Li et al. "Service selection mechanisms in the Internet of Things (IoT): a systematic and comprehensive study". In: *Clust. Comput.* 23.2 (2020), pp. 1163–1183. DOI: 10.1007/s10586-019-02984-4.

[19] B. Pourghebleh, V. Hayyolalam, and A. A. Anvigh. "Service discovery in the Internet of Things: review of current trends and research challenges". In: *Wirel. Networks* 26.7 (2020), pp. 5371–5391. DOI: 10.1007/s11276-020-02405-0.

[20] Aghabi N. Abosaif and Haitham S. Hamza. "Quality of service-aware service selection algorithms for the internet of things environment: A review paper". In: *Array* 8 (2020), p. 100041. DOI: 10.1016/j.array.2020.100041.

[21] A. I. Al-Fuqaha et al. "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications". In: *IEEE Commun. Surv. Tutorials* 17.4 (2015), pp. 2347–2376. DOI: 10.1109/COMST.2015.2444095.

[22] R. Khan et al. "Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges". In: *10th International Conference on Frontiers of Information Technology, FIT 2012, Islamabad, Pakistan, December 17-19, 2012*. IEEE Computer Society, 2012, pp. 257–260. DOI: 10.1109/FIT.2012.53.

[23] P. Mell and T. Grance. "The NIST definition of cloud computing v15". In: *http://csrc. nist. gov/groups/SNS/cloud-computing/* (2009).

[24] N. Abdullah Bugshan. "An overview of current security threats and existing solutions in Fog Computing". In: *IRJCS Issue 04* Volume 6 (2019).

[25] P. Hu et al. "Survey on fog computing: architecture, key technologies, applications and open issues". In: *Journal of Network and Computer Applications* 98 (2017), pp. 27–42. DOI: 10.1016/j.jnca.2017.09.002.

[26] M. E. Adam. "Usages of Semantic Web Services Technologies in IoT Ecosystems and its Impact in Services Delivery: A survey". In: *International Journal of Computer (IJC)* 36.1 (2020), pp. 53–72.

[27] *SPARQL Query Language for RDF (SPARQL)*. URL: https://www.w3.org/TR/rdf-sparql-query/. (accessed: 21.01.2021).

[28] R. Studer, V.R. Benjamins, and D. Fensel. "Knowledge Engineering: Principles and Methods". In: *Data Knowl. Eng.* 25.1-2 (1998), pp. 161–197. DOI: 10.1016/S0169-023X(97)00056-6.

[29] A. Rhayem, M. Ben Ahmed Mhiri, and F. Gargouri. "Semantic Web Technologies for the Internet of Things: Systematic Literature Review". In: *Internet Things* 11 (2020), p. 100206. DOI: 10.1016/j.iot.2020.100206.

[30] D. Lonsdale et al. "Reusing ontologies and language components for ontology generation". In: *Data Knowl. Eng.* 69.4 (2010), pp. 318–330. DOI: 10.1016/j.datak.2009.08.003.

[31] C. Dai and Z. Wang. "A flexible extension of WSDL to describe non-functional attributes". In: *2010 2nd International Conference on E-business and Information System Security*. IEEE. 2010, pp. 1–4.

[32] A. Adala, N. Tabbane, and S. Tabbane. "A Framework for Automatic Web Service Discovery Based on Semantics and NLP Techniques". In: *Advances in Multimedia* 2011 (2011), 238683:1–238683:7. DOI: 10.1155/2011/238683.

[33] M. Singh and G. Baranwal. "Quality of service (qos) in internet of things". In: *2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU)*. IEEE. 2018, pp. 1–6.

[34] E. Al Nuaimi and N. Al Darmaki. "Managing QoS in IoTs: a survey". In: *Proceedings of the Second International Conference on Internet of things and Cloud Computing, ICC 2017, Cambridge, United Kingdom, March 22-23, 2017*. Ed. by H. Hamdan et al. ACM, 2017, 77:1–77:7. DOI: 10.1145/3018896.3018977.

[35] Y. Dongre and R. Ingle. "An Investigation of QoS Criteria for Optimal Services Selection in Composition". In: *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*. IEEE. 2020, pp. 705–710.

[36] M. Suryanegara et al. "A 5-Step Framework for Measuring the Quality of Experience (QoE) of Internet of Things (IoT) Services". In: *IEEE Access* 7 (2019), pp. 175779–175792. DOI: 10.1109/ACCESS.2019.2957341.

[37] D. H. Shin. "Conceptualizing and measuring quality of experience of the Internet of Things: Exploring how quality is perceived by users". In: *Inf. Manage.* 54.8 (2017), pp. 998–1011. DOI: 10.1016/j.im.2017.02.006.

[38] S. Ben Fredj et al. "Efficient semantic-based IoT service discovery mechanism for dynamic environments". In: *25th IEEE Annual International Symposium on Personal, Indoor, and Mobile Radio Communication, PIMRC 2014, Washington DC, USA, September 2-5, 2014*. IEEE, 2014, pp. 2088–2092. DOI: 10.1109/PIMRC.2014.7136516.

[39] S. Chirila, C. Lemnaru, and M. Dınsoreanu. "Semantic-based IoT device discovery and recommendation mechanism". In: *IEEE 12th International Conference on Intelligent Computer Communication and Processing, ICCP 2016, Cluj-Napoca, Romania, September 8-10, 2016*. IEEE, 2016, pp. 111–116. DOI: 10.1109/ICCP.2016.7737131.

[40] P. Gomes et al. "A Federated Discovery Service for the Internet of Things". In: *Proceedings of the 2nd Workshop on Middleware for Context-Aware Applications in the IoT, M4IoT@Middleware 2015, Vancouver, BC, Canada, December 7-11, 2015*. ACM, 2015, pp. 25–30. DOI: 10.1145/2836127.2836129.

[41] B. Jia and W. Li & T. Zhou. "A Centralized Service Discovery Algorithm via Multi-Stage Semantic Service Matching in Internet of Things". In: *2017 IEEE International Conference on Computational Science and Engineering, CSE 2017, and IEEE International Conference on Embedded and Ubiquitous Computing, EUC 2017, Guangzhou, China, July 21-24, 2017, Volume 1*. IEEE Computer Society, 2017, pp. 422–427. DOI: 10.1109/CSE-EUC.2017.82.

[42] B. Yuan, L. Liu, and N. Antonopoulos. "Efficient service discovery in decentralized online social networks". In: *Future Generation Computer Systems* 86 (2018), pp. 775–791. DOI: 10.1016/j.future.2017.04.022.

[43] H. Xia et al. "An efficient social-like semantic-aware service discovery mechanism for large-scale Internet of Things". In: *Computer Networks* 152 (2019), pp. 210–220. DOI: 10.1016/j.comnet.2019.02.006.

[44] F. Liu et al. "Event-Driven Semantic Service Discovery Based on Word Embeddings". In: *IEEE Access* 6 (2018), pp. 61030–61038. DOI: 10.1109/ACCESS.2018.2876029.

[45] P. Gomes et al. "A semantic-based discovery service for the Internet of Things". In: *Journal of Internet Services and Applications* 10.1 (2019), 10:1–10:14. DOI: 10.1186/s13174-019-0109-8.

[46] B. Djamaa, A. Yachir, and M. A. Richardson. "Hybrid CoAP-based resource discovery for the Internet of Things". In: *J. Ambient Intell. Humaniz. Comput.* 8.3 (2017), pp. 357–372. DOI: 10.1007/s12652-017-0450-3.

[47] J. Bao, Y. Ding, and H. Hu. "A new service selection algorithm in USPIOT". In: *2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE)*. Vol. 2. IEEE. 2012, pp. 22–26.

[48] B. Jia et al. "A Privacy-sensitive Service Selection Method Based on Artificial Fish Swarm Algorithm in the Internet of Things". In: *Mobile Networks and Applications* 26.4 (2021), pp. 1523–1531. DOI: 10.1007/s11036-019-01488-0.

[49] K. D. Baek and I-Y. Ko. "Spatio-Cohesive Service Selection Using Machine Learning in Dynamic IoT Environments". In: *Web Engineering - 18th International Conference, ICWE 2018, Cáceres, Spain, June 5-8, 2018, Proceedings*. Ed. by T. Mikkonen, R. Klamma, and J. Hernández. Vol. 10845. Lecture Notes in Computer Science. Springer, 2018, pp. 366–374. DOI: 10.1007/978-3-319-91662-0_30.

[50] A. Yachir et al. "Event-Aware Framework for Dynamic Services Discovery and Selection in the Context of Ambient Intelligence and Internet of Things". In: *IEEE Trans Autom. Sci. Eng.* 13.1 (2016), pp. 85–102. DOI: 10.1109/TASE.2015.2499792.

[51] L. Manqele et al. “Preference-based Internet of Things dynamic service selection for smart campus”. In: *AFRICON 2015, Addis Ababa, Ethiopia, September 14-17, 2015*. IEEE, 2015, pp. 1–5. DOI: 10.1109/AFRCON.2015.7332047.

[52] L. Qi et al. “”Time-Location-Frequency”-aware Internet of things service selection based on historical records”. In: *International Journal of Distributed Sensor Networks* 13.1 (2017). DOI: 10.1177/1550147716688696.

[53] P. Asghari, A.M. Rahmani, and H.H.S Javadi. “Service composition approaches in IoT: A systematic review”. In: *Journal of Network and Computer Applications* 120 (2018), pp. 61–77. DOI: 10.1016/j.jnca.2018.07.013.

[54] G. Chen et al. “A Social Network Based Approach for IoT Device Management and Service Composition”. In: *2015 IEEE World Congress on Services, SERVICES 2015, New York City, NY, USA, June 27 - July 2, 2015*. Ed. by L-J. Zhang and R. Bahsoon. IEEE Computer Society, 2015, pp. 1–8. DOI: 10.1109/SERVICES.2015.9.

[55] M. Sun et al. “Energy-Efficient Composition of Configurable Internet of Things Services”. In: *IEEE Access* 5 (2017), pp. 25609–25622. DOI: 10.1109/ACCESS.2017.2768544.

[56] O. Alsaryrah, I. Mashal, and T. Chung. “Bi-Objective Optimization for Energy Aware Internet of Things Service Composition”. In: *IEEE Access* 6 (2018), pp. 26809–26819. DOI: 10.1109/ACCESS.2018.2836334.

[57] M. Hosseinzadeh et al. “A Hybrid Service Selection and Composition Model for Cloud-Edge Computing in the Internet of Things”. In: *IEEE Access* 8 (2020), pp. 85939–85949. DOI: 10.1109/ACCESS.2020.2992262.

[58] M. E. Khanouche et al. “Energy-Centered and QoS-Aware Services Selection for Internet of Things”. In: *IEEE TRANS ON AUTOMATION SCIENCE AND ENGINEERING* 13.3 (2016), pp. 1256–1269. DOI: 10.1109/TASE.2016.2539240.

[59] C-C. Lin, D-J. Deng, and L-Y. Lu. “Many-Objective Sensor Selection in IoT Systems”. In: *IEEE Wirel. Commun.* 24.3 (2017), pp. 40–47. DOI: 10.1109/MWC.2017.1600409.

[60] G. Baranwal, M. Singh, and D. P. Vidyarthi. “A framework for IoT service selection”. In: *The Journal of Super-computing* 76.4 (2020), pp. 2777–2814. DOI: 10.1007/s11227-019-03076-1.

[61] Q. Wei and Z. Jin. “Service discovery for internet of things: a context-awareness perspective”. In: *Proceedings of the Fourth Asia-Pacific Symposium on Internetware, Internetware 2012, QingDao, China, October 30-31, 2012*. Ed. by H. Mei et al. ACM, 2012, 25:1–25:6. DOI: 10.1145/2430475.2430500.

[62] S. Pattar et al. “Progressive Search Algorithm for Service Discovery in an IoT Ecosystem”. In: *2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), iThings/GreenCom/CPSCom/SmartData 2019, Atlanta, GA, USA, July 14-17, 2019*. IEEE, 2019, pp. 1041–1048. DOI: 10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00180.

[63] M. Mejri and N. Ben Azzouna. “Scalable and self-adaptive service selection method for the Internet of Things”. In: *International Journal of Computer Applications* 167.10 (2017), pp. 43–49.

[64] M. Stolikj et al. “Context based service discovery in unmanaged networks using mDNS/DNS-SD”. In: *IEEE International Conference on Consumer Electronics, ICCE 2016, Las Vegas, NV, USA, January 7-11, 2016*. IEEE, 2016, pp. 163–165. DOI: 10.1109/ICCE.2016.7430565.

[65] M. Mahyoub and A. Mahmoud & T. Sheltami. “An optimized discovery mechanism for smart objects in IoT”. In: *2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE. 2017, pp. 649–655.

[66] J. Quevedo et al. “On the application of contextual IoT service discovery in Information Centric Networks”. In: *Computer Communications* 89-90 (2016), pp. 117–127. DOI: 10.1016/j.comcom.2016.03.011.

[67] A. H. Ahmed, N. M. Omar, and H. M. Ibrahim. “Secured Service Discovery Technique in IoT”. In: *JCM* 14.1 (2019), pp. 40–46. DOI: 10.12720/jcm.14.1.40-46.

[68] A. Cimmino, M.Poveda-Villalón, and R. Garcıa-Castro. “eWoT: A Semantic Interoperability Approach for Heterogeneous IoT Ecosystems Based on the Web of Things”. In: *Sensors* 20.3 (2020), p. 822. DOI: 10.3390/s20030822.

[69] I. Araújo et al. “Service Discovery Based on Social Profiles of Objects in a Social IoT Network”. In: *Computational Science and Its Applications - ICCSA 2019 - 19th International Conference, Saint Petersburg, Russia, July 1-4, 2019, Proceedings, Part V*. Ed. by S. Misra et al. Vol. 11623. Lecture Notes in Computer Science. Springer, 2019, pp. 400–414. DOI: 10.1007/978-3-030-24308-1_33.

[70] T-D. Nguyen, E-N. Huh, and M. Jo. “Decentralized and revised content-centric networking-based service deployment and discovery platform in mobile edge computing for IoT devices”. In: *IEEE Internet of Things Journal* 6.3 (2018), pp. 4162–4175.

[71] H. Benkaouha, L. Mokdad, and A. Abdelkrim. “2PACA: Two phases algorithm of checkpointing for Ad hoc mobile networks”. In: *2013 9th International Wireless Communications and Mobile Computing Conference, IWCMC 2013, Sardinia, Italy, July 1-5, 2013*. Ed. by R. Saracco et al. IEEE, 2013, pp. 1359–1364. DOI: 10.1109/IWCMC.2013.6583754.

[72] H. Benkaouha et al. “Towards Improving Failure Detection in Mobile Ad Hoc Networks”. In: *2015 IEEE Global Communications Conference, GLOBECOM 2015, San Diego, CA, USA, December 6-10, 2015*. IEEE, 2015, pp. 1–6. DOI: 10.1109/GLOCOM.2014.7417004.

[73] R. Ferdousi and P. K. Mandal. "LOAMY: A cloud-based middleware for CoAP-based IoT service discovery". In: *2019 Second international conference on advanced computational and communication paradigms (ICACCP)*. IEEE. 2019, pp. 1–6.

[74] A. Dahbi. "Supply Chain Discovery Services in an Internet of Things Environment". 2017.

[75] H. Lee et al. "IoT Service Store: A Web-based System for Privacy-aware IoT Service Discovery and Interaction". In: *2018 IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2018, Athens, Greece, March 19-23, 2018*. IEEE Computer Society, 2018, pp. 107–112. DOI: 10.1109/PERCOMW.2018.8480260.

[76] E. Wang and R. Chow. "What can i do here? IoT service discovery in smart cities". In: *2016 IEEE International Conference on Pervasive Computing and Communication Workshops, PerCom Workshops 2016, Sydney, Australia, March 14-18, 2016*. IEEE Computer Society, 2016, pp. 1–6. DOI: 10.1109/PERCOMW.2016.7457097.

[77] P. Maiti et al. "An effective approach of latency-aware fog smart gateways deployment for IoT services". In: *Internet Things* 8 (2019). DOI: 10.1016/j.iot.2019.100091.

[78] R. Venanzi et al. "MQTT-Driven Node Discovery for Integrated IoT-Fog Settings Revisited: The Impact of Advertiser Dynamicity". In: *IEEE Symposium on Service-Oriented System Engineering, SOSE 2018, Bamberg, Germany, March 26-29, 2018*. IEEE Computer Society, 2018, pp. 31–39. DOI: 10.1109/SOSE.2018.00013.

[79] M. E. Khansari and S. Sharifian. "A modified water cycle evolutionary game theory algorithm to utilize QoS for IoT services in cloud-assisted fog computing environments". In: *The Journal of Supercomputing* 76.7 (2020), pp. 5578–5608. DOI: 10.1007/s11227-019-03095-y.

[80] J. Li et al. "A Decentralized Trustworthy Context and QoS-Aware Service Discovery Framework for the Internet of Things". In: *IEEE Access* 5 (2017), pp. 19154–19166. DOI: 10.1109/ACCESS.2017.2756446.

[81] K. Kalkan and K. B. Rasmussen. "TruSD: Trust framework for service discovery among IoT devices". In: *Computer Networks* 178 (2020), p. 107318. DOI: 10.1016/j.comnet.2020.107318.

[82] S. Cirani et al. "A Scalable and Self-Configuring Architecture for Service Discovery in the Internet of Things". In: *IEEE Internet Things J.* 1.5 (2014), pp. 508–521. DOI: 10.1109/JIOT.2014.2358296.

[83] G. Kim et al. "An MQTT-based context-aware autonomous system in oneM2M architecture". In: *IEEE Internet of Things Journal* 6.5 (2019), pp. 8519–8528.

[84] E. M. Pereira et al. "MQTT-RD: A MQTT based Resource Discovery for Machine to Machine Communication". In: *Proceedings of the 4th International Conference on Internet of Things, Big Data and Security, IoTBDS 2019, Heraklion, Crete, Greece, May 2-4, 2019*. Ed. by M. Ramachandran et al. SciTePress, 2019, pp. 115–124. DOI: 10.5220/0007716201150124.

[85] K. Sahlmann, T. Scheffler, and B. Schnor. "Ontology-driven Device Descriptions for IoT Network Management". In: *2018 Global Internet of Things Summit, GIoTS 2018, Bilbao, Spain, June 4-7, 2018*. IEEE, 2018, pp. 1–6. DOI: 10.1109/GIOTS.2018.8534569.

[86] J. Bao and J. Xia. "A hybrid algorithm for service matchmaking based on ontology approach". In: *2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*. IEEE. 2017, pp. 2420–2424.

[87] D. Guinard et al. "Interacting with the SOA-Based Internet of Things: Discovery, Query, Selection, and On-Demand Provisioning of Web Services". In: *IEEE Transactions On Services Computing* 3.3 (2010), pp. 223–235. DOI: 10.1109/TSC.2010.3.

[88] P. Krivic, P. Skocir, and M. Kusek. "Agent-Based Approach for Energy-Efficient IoT Services Discovery and Management". In: *Agents and Multi-Agent Systems: Technologies and Applications 2018, Proceedings of the 12th International Conference on Agents and Multi-Agent Systems, KES-AMSTA 2018, Gold Coast, QLD, Australia, 20-22 June, 2018*. Ed. by G. Jezic et al. Vol. 96. Smart Innovation, Systems and Technologies. Springer, 2018, pp. 57–66. DOI: 10.1007/978-3-319-92031-3_6.

[89] E. Rapti et al. "A Bio-Inspired Service Discovery and Selection Approach for IoT Applications". In: *IEEE International Conference on Services Computing, SCC 2016, San Francisco, CA, USA, June 27 - July 2, 2016*. Ed. by J. Zhang, J. A. Miller, and X. Xu. IEEE Computer Society, 2016, pp. 868–871. DOI: 10.1109/SCC.2016.126.

[90] E. Rapti et al. "Decentralized service discovery and selection in Internet of Things applications based on artificial potential fields". In: *Service Oriented Computing and Applications* 11.1 (2017), pp. 75–86. DOI: 10.1007/s11761-016-0198-1.

[91] M. Pruthvi, S. Karthika, and N. Bhalaji. "A Novel Framework for SIoT College". In: *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*. IEEE. 2019, pp. 1–4.

[92] H. Quan, R. Takahashi, and F. Yoshiaki. "Dynamic Service Selection based on User Feedback in the IoT Environment". In: *2019 International Conference on Computer, Information and Telecommunication Systems, CITS 2019, Beijing, China, August 28-31, 2019*. Ed. by M. S. Obaidat et al. IEEE, 2019, pp. 1–5. DOI: 10.1109/CITS.2019.8862134.

[93] B.Djamaa et al. "Towards efficient distributed service discovery in low-power and lossy networks". In: *Wirel. Networks* 20.8 (2014), pp. 2437–2453. DOI: 10.1007/s11276-014-0749-3.

[94] A. M. Kowshalya, X. Z. Gao, and M. L. Valarmathi. "Efficient service search among Social Internet of Things through construction of communities". In: *Cyber-Physical Systems* 6.1 (2020), pp. 33–48.

[95] S. K. Datta. "Towards securing discovery services in Internet of Things". In: *IEEE International Conference on Consumer Electronics, ICCE 2016, Las Vegas, NV, USA, January 7-11, 2016*. IEEE, 2016, pp. 506–507. DOI: 10.1109/ICCE.2016.7430707.

[96] C. P. Vandana and A. A. Chikkamannur. "S-COAP: Semantic Enrichment of COAP for Resource Discovery". In: *SN Computer Science* 1.2 (2020), pp. 1–7.

[97] G. Tanganelli and C. Vallati & E. Mingozzi. "Edge-Centric Distributed Discovery and Access in the Internet of Things". In: *IEEE Internet Things Journal* 5.1 (2018), pp. 425–438. DOI: 10.1109/JIOT.2017.2767381.

[98] A. Khanfor et al. "Automated Service Discovery for Social Internet-of-Things Systems". In: *IEEE International Symposium on Circuits and Systems, ISCAS 2020, Sevilla, Spain, October 10-21, 2020*. IEEE, 2020, pp. 1–5. DOI: 10.1109/ISCAS45731.2020.9181080.

[99] N. S. Nizamkari. "A graph-based trust-enhanced recommender system for service selection in IOT". In: *2017 International Conference on Inventive Systems and Control (ICISC)*. IEEE. 2017, pp. 1–5.

[100] H. Wang et al. "A Lightweight XMPP Publish/Subscribe Scheme for Resource-Constrained IoT Devices". In: *IEEE Access* 5 (2017), pp. 16393–16405. DOI: 10.1109/ACCESS.2017.2742020.

[101] P. Gupta et al. "Event-Driven SOA-based IoT Architecture". In: *International Conference on Intelligent Computing and Applications*. Vol. 632. Springer. 2018, pp. 247–258.

[102] N. Son Han, S. Park, and G. M. Lee & N. Crespi. "Extending the devices profile for web services standard using a REST proxy". In: *IEEE Internet Computing* 19.1 (2014), pp. 10–17.

[103] M. Butcher. "JSON, HTTP, and the Future of IoT Protocols". In: *dzone's 2015 guide to the internet of things* (2015).

[104] R. Klauck and M. Kirsche. "Bonjour Contiki: A Case Study of a DNS-Based Discovery Service for the Internet of Things". In: *Ad-hoc, Mobile, and Wireless Networks - 11th International Conference, ADHOC-NOW 2012, Belgrade, Serbia, July 9-11, 2012. Proceedings*. Ed. by X-Y. Li, S. Papavassiliou, and S. Rührup. Vol. 7363. Lecture Notes in Computer Science. Springer, 2012, pp. 316–329. DOI: 10.1007/978-3-642-31638-8_24.

[105] H-J. Jo, J-H. Kwon, and I-Y. Ko. "Distributed Service Discovery in Mobile IoT Environments Using Hierarchical Bloom Filters". In: *Engineering the Web in the Big Data Era - 15th International Conference, ICWE 2015, Rotterdam, The Netherlands, June 23-26, 2015, Proceedings*. Ed. by P. Cimiano et al. Vol. 9114. Lecture Notes in Computer Science. Springer, 2015, pp. 498–514. DOI: 10.1007/978-3-319-19890-3_32.

[106] T. Berners-Lee, J. Hendler, and O. Lassila. "The semantic web". In: *Scientific american* 284.5 (2001), pp. 34–43.

[107] L. C. De Biase et al. "The semantic Mediation for the Swarm: An adaptable and organic solution for the Internet of Things". In: *IEEE International Conference on Consumer Electronics, ICCE 2017, Las Vegas, NV, USA, January 8-10, 2017*. IEEE, 2017, pp. 78–79. DOI: 10.1109/ICCE.2017.7889237.

[108] V. Charpenay and S. Käbisch. "On Modeling the Physical World as a Collection of Things: The W3C Thing Description Ontology". In: *The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings*. Ed. by A. Harth et al. Vol. 12123. Lecture Notes in Computer Science. Springer, 2020, pp. 599–615. DOI: 10.1007/978-3-030-49461-2_35.

[109] S. Käbisch and D. Anicic. "Thing description as enabler of semantic interoperability on the Web of Things". In: *Proc. IoT Semantic Interoperability Workshop*. 2016, pp. 1–3.

[110] C. Qu et al. "An OWL-S based specification model of dynamic entity services for Internet of Things". In: *J. Ambient Intell. Humaniz. Comput.* 7.1 (2016), pp. 73–82. DOI: 10.1007/s12652-015-0302-y.

[111] F. Khodadadi and A. V. Dastjerdi & R. Buyya. "Simurgh: A framework for effective discovery, programming, and integration of services exposed in IoT". In: *2015 International Conference on Recent Advances in Internet of Things (RIoT)*. IEEE. 2015, pp. 1–6.

[112] W. Wang, G. Cassar S. De, and K. Moessner. "An experimental study on geospatial indexing for sensor service discovery". In: *Expert Systems with Applications* 42.7 (2015), pp. 3528–3538. DOI: 10.1016/j.eswa.2014.11.058.

[113] N. Le Sommer et al. "A Disruption-Tolerant RESTful Support for the Web of Things". In: *4th IEEE International Conference on Future Internet of Things and Cloud, FiCloud 2016, Vienna, Austria, August 22-24, 2016*. Ed. by M. Younas, I. Awan, and W. Seah. IEEE Computer Society, 2016, pp. 17–24. DOI: 10.1109/FiCloud.2016.11.

[114] F. albalas, W. Mardini, and M. Al-Soud. "AFT: Adaptive Fibonacci-based Tuning Protocol for Service and Resource discovery in the Internet of Things". In: *Second International Conference on Fog and Mobile Edge Computing, FMEC 2017, Valencia, Spain, May 8-11, 2017*. IEEE, 2017, pp. 177–182. DOI: 10.1109/FMEC.2017.7946427.

[115] B. Djamaa et al. "FetchIoT: Efficient Resource Fetching for the Internet of Things". In: *Proceedings of the 2018 Federated Conference on Computer Science and Information Systems, FedCSIS 2018, Poznań, Poland, September 9-12, 2018*. Ed. by M. Ganzha, L. A. Maciaszek, and M. Paprzycki. Vol. 15. Annals of Computer Science and Information Systems. 2018, pp. 637–643. DOI: 10.15439/2018F278.

[116] P. C. Calcina-Ccori et al. "Enabling Semantic Discovery in the Swarm". In: *IEEE Trans. Consumer Electron.* 65.1 (2019), pp. 57–63. DOI: 10.1109/TCE.2018.2888511.

[117] S. Zhao, L. Yu, and B. Cheng & J. Chen. "IoT Service Clustering for Dynamic Service Matchmaking". In: *Sensors* 17.8 (2017), p. 1727. DOI: 10.3390/s17081727.

[118] Z. B. Azizou, A. Boudries, and M. Amad. "Decentralized service discovery and localization in Internet of Things applications based on ant colony algorithm". In: *International Journal of Computing and Digital Systems* 9.5 (2020), pp. 941–950.

[119] Z. Zhang et al. "Peer discovery for D2D communications based on social attribute and service attribute". In: *Journal of Network and Computer Applications* 86 (2017), pp. 82–91. DOI: 10.1016/j.jnca.2016.11.006.

[120] A. Paul et al. "Smartbuddy: defining human behaviors using big data analytics in social internet of things". In: *IEEE Wireless Communications* 23.5 (2016), pp. 68–74. DOI: 10.1109/MWC.2016.7721744.

[121] M. Tao, W. Wei, and S. Huang. "Location-based trustworthy services recommendation in cooperative-communication-enabled Internet of Vehicles". In: *Journal of Network and Computer Applications* 126 (2019), pp. 1–11. DOI: 10.1016/j.jnca.2018.10.023.

[122] L. Lan et al. "An Event-driven Service-oriented Architecture for the Internet of Things Service Execution." In: *International Journal of Online Engineering* 11.2 (2015).

[123] B. Cheng et al. "Situation-Aware IoT Service Coordination Using the Event-Driven SOA Paradigm". In: *IEEE Trans. Netw. Serv. Manag.* 13.2 (2016), pp. 349–361. DOI: 10.1109/TNSM.2016.2541171.

[124] J-y. Hong, E-h. Suh, and S-J. Kim. "Context-aware systems: A literature review and classification". In: *Expert Systems with Applications* 36.4 (2009), pp. 8509–8522. DOI: 10.1016/j.eswa.2008.10.071.

[125] X. Jin et al. "A fast and scalable approach for IoT service selection based on a physical service model". In: *Information Systems Frontiers* 19.6 (2017), pp. 1357–1372. DOI: 10.1007/s10796-016-9650-1.

[126] S. Ali, M. G. Kibria, and I. Chong. "WoO enabled IoT service provisioning based on learning user preferences and situation". In: *2017 International Conference on Information Networking, ICOIN 2017, Da Nang, Vietnam, January 11-13, 2017*. IEEE, 2017, pp. 474–476. DOI: 10.1109/ICOIN.2017.7899538.

[127] C. Cabrera et al. "Services in IoT: A Service Planning Model Based on Consumer Feedback". In: *Service-Oriented Computing - 16th International Conference, ICSOC 2018, Hangzhou, China, November 12-15, 2018, Proceedings*. Ed. by C. Pahl et al. Vol. 11236. Lecture Notes in Computer Science. Springer, 2018, pp. 304–313. DOI: 10.1007/978-3-030-03596-9_21.

[128] A. N. Abu-Safe and S. E. Elrofai. "QOS–aware meta-heuristic services selection algorithm and likert scale measurement for IOT environment". In: *International Journal of Computer Science Trends and Technology* 8.1 (2020), pp. 1–8.

[129] N.N.H Win, J.M BAO, and G. CUI. "Flexible user-centric service selection algorithm for internet of things services". In: *The Journal of China Universities of Posts and Telecommunications* 21 (2014), pp. 64–70.

[130] H. Gao et al. "Context-Aware QoS Prediction With Neural Collaborative Filtering for Internet-of-Things Services". In: *IEEE Internet Things J.* 7.5 (2020), pp. 4532–4542. DOI: 10.1109/JIOT.2019.2956827.

[131] K. Fysarakis et al. "Which IoT Protocol? Comparing Standardized Approaches over a Common M2M Application". In: *2016 IEEE Global Communications Conference, GLOBECOM 2016, Washington, DC, USA, December 4-8, 2016*. IEEE, 2016, pp. 1–7. DOI: 10.1109/GLOCOM.2016.7842383.

[132] R. Venanzi et al. "MQTT-Driven Sustainable Node Discovery for Internet of Things-Fog Environments". In: *2018 IEEE International Conference on Communications, ICC 2018, Kansas City, MO, USA, May 20-24, 2018*. IEEE, 2018, pp. 1–6. DOI: 10.1109/ICC.2018.8422200.

[133] R. Kurte, Z. Salcicy, and K. I-K. Wang. "A Distributed Service Framework for the Internet of Things". In: *IEEE Transactions on Industrial Informatics* 16.6 (2020), pp. 4166–4176. DOI: 10.1109/TII.2019.2948046.

[134] J. Na et al. "An Evolutionary Game Approach on IoT Service Selection for Balancing Device Energy Consumption". In: *12th IEEE International Conference on e-Business Engineering, ICEBE 2015, Beijing, China, October 23-25, 2015*. Ed. by Y. Li et al. IEEE Computer Society, 2015, pp. 331–338. DOI: 10.1109/ICEBE.2015.63.

[135] V. Sharma, F. Song, and I. Youa & M. Atiquzzaman. "Energy efficient device discovery for reliable communication in 5G-based IoT and BSNs using unmanned aerial vehicles". In: *Journal of Network and Computer Applications* 97 (2017), pp. 79–95. DOI: 10.1016/j.jnca.2017.08.013.

[136] M. Antonini et al. "Lightweight multicast forwarding for service discovery in low-power IoT networks". In: *22nd International Conference on Software, Telecommunications and Computer Networks, SoftCOM 2014, Split, Croatia, September 17-19, 2014*. Ed. by N. Rozic and D. Begusic. IEEE, 2014, pp. 133–138. DOI: 10.1109/SOFTCOM.2014.7039103.

[137] C. Perera et al. "Sensor search techniques for sensing as a service architecture for the internet of things". In: *IEEE Sensor J* 14.2 (2014), pp. 406–420.

[138] M. Singh and G. Baranwal & A. K. Tripathi. "QoS-Aware Selection of IoT-Based Service." In: *Arabian Journal for Science & Engineering (Springer Science & Business Media BV)* 45.12 (2020), pp. 10033–10050.

[139] L. Li, S. Li, and S. Zhao. "QoS-Aware Scheduling of Services-Oriented Internet of Things". In: *IEEE Trans. Ind. Informatics* 10.2 (2014), pp. 1497–1505. DOI: 10.1109/TII.2014.2306782.

[140] Z. Huang et al. "Co-locating services in IoT systems to minimize the communication energy cost". In: *J. Innov. Digit. Ecosyst.* 1.1-2 (2014), pp. 47–57. DOI: 10.1016/j.jides.2015.02.005.

[141] Y. Fathy, P. M. Barnaghi, and R. Tafazolli. "Large-Scale Indexing, Discovery, and Ranking for the Internet of Things (IoT)". In: *ACM Computing Surveys* 51.2 (2018), 29:1–29:53. DOI: 10.1145/3154525.