

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.Doi Number

Use-case-centric and industry-agnostic architectural patterns for fog systems

Maximilian Blume¹, Sebastian Lins², and Ali Sunyaev³

¹Karlsruhe Institute of Technology, Karlsruhe, 76131 Germany

²University of Kassel, Kassel, 34125 Germany

³Technical University of Munich, Munich, 80333 Munich

Corresponding author: Maximilian Blume (e-mail: maximilian.blume@partner.kit.edu).

ABSTRACT Fog computing has emerged as a promising paradigm for supporting applications based on edge and cloud computing by providing decentralized and intermediate computing, storage, and networking infrastructure. Fog computing is increasingly recognized as a critical enabler for modern Internet of Things (IoT) applications that require ultra-low latency, local resilience, and improved data privacy. However, designing effective fog systems remains a significant challenge due to the diversity of use case requirements, the immaturity of the domain, and a lack of systematic guidance. We therefore examine how to design effective fog systems. We adopt a use-case-centric and industry-agnostic perspective to develop a comprehensive set of architectural patterns for fog systems. Drawing on a systematic literature review and expert interviews, we identify critical fog use case characteristics, typical system tasks, and intermediary roles that fog nodes must perform. Based on these elements, we derive and refine architectural patterns that guide system architects in selecting and implementing suitable fog designs that are aligned with their specific use case objectives. Our contribution provides a structured framework for the principled design of fog systems and supports both researchers and practitioners in bridging the gap between abstract requirements and concrete system architectures.

INDEX TERMS Architectural Patterns, Edge Computing, Fog Computing, Internet of Things

I. INTRODUCTION

With over 50 billion connected devices worldwide [1], cloud technology based on large central data centers is increasingly struggling to achieve the required latency, resilience, and security for contemporary Internet of Things (IoT) applications [2]. Fog computing has emerged as a novel computing model for coping with these problems and has become indispensable for IoT use cases: it bridges the geographical and logical distances between edge devices and centralized cloud services by introducing an intermediate layer consisting of fog nodes [3]. These fog nodes provide computing, storage, and networking capabilities geographically close to edge devices [4], thereby enabling ultra-low latency and data-volume-intensive applications such as autonomous driving [5] and augmented reality [6]. In addition, practitioners state improvements in privacy and security as well as increased independence from internet connectivity as key drivers for adopting fog computing [7].

Fog nodes can support cloud services and edge devices in varied and cost efficient ways due to their intermediary nature, resulting in many options for designing fog systems. In particular, the initial system architecture (e.g., regarding the physical and virtual design of fog nodes) remains challenging and crucial for architects because the implementation influences

the effectiveness of the entire system, and some decisions are difficult to change once the system is operational [1], [8]. For instance, after the physical distance between edge devices and fog nodes and the required communication technology are determined, hundreds of edge devices will be physically equipped accordingly, and their subsequent adaptation will be costly.

Architects must consider two key driving forces when designing a system with embedded fog nodes. First, the characteristics of the specific use case impact the design of fog systems [9], [10]. For example, different edge device capabilities, volatility of data loads, and expected system dynamics necessitate different fog node designs to achieve high system performance. Second, architects need to determine which tasks fog nodes should perform as intermediaries between edge devices and cloud services to fulfill the use case objectives [11]. For example, a use case may require fog nodes to analyze sensor data in real time to detect abnormalities [12], requiring a fog node architecture that is optimized for low-latency processing support for edge devices. To cope with both driving forces, architects must engage in the use-case-centric design of fog systems.

However, determining how to best design and set up a system with embedded fog nodes constitutes a severe difficulty because architects must consider the complex set of characteristics and tasks that fog nodes need to fulfill depending on the use case (e.g., ultralow latency, resilience, cost efficiency, and security [9], [13]). In addition, the immaturity of the fog computing domain and lack of large-scale implementation examples impose an additional burden for system architects and developers, as an effective architecture must be determined for their system [1], [8]. As the terminology and manifestations of fog tasks and architectures are diverse and best practices are lacking, practitioners struggle to find guidance for designing their systems while considering their specific use cases, and it is difficult to identify comparable architectures [3]. If ineffective design choices are made, the full value of fog systems may remain untapped. Further guidance in designing and setting up fog systems is needed [14].

Given the clear opportunities of fog computing, related research is continuously increasing and provides valuable starting points to support practitioners and researchers. First, there is a growing number of proposals for (industry-)specific fog architectures that provide tangible examples of how a fog system can be implemented in certain circumstances [9]. However, for designing a new fog system, this is only helpful if the new use case generally matches the use case examined by the article, and semantic heterogeneity also makes it difficult to find relevant research articles. Second, related research provides general recommendations on how to improve a fog system to meet a certain requirement (e.g., ensuring ultralow latency [15]) or to meet industry-specific standards (e.g., in healthcare [16]). However, these research endeavors often neglect other characteristics, synergies, or tradeoffs for design decisions, and most focus on the operation of fog systems (e.g., the optimization of task scheduling). Third, system taxonomies and literature reviews provide an overview of design choices for architectures with embedded fog nodes (e.g., [17]) but rarely link these to their impact on how such fog systems fulfill certain requirements.

Overall, the existing research provides valuable input for designing fog systems in very specific ways for particular industries, examples for certain use cases or general design options. However, it does not include a use-case-centric view on how a fog system can be designed to perform tasks with different requirements. Furthermore, it does not consider important use case characteristics, potential design options, or how they can be effectively combined. Accordingly, we set the following research question: How can effective fog systems be designed depending on the use case?

We address this problem by developing architectural patterns for fog systems based on both the literature and diverse interviews with fog computing experts. To take a use-case-centric and industry-agnostic perspective, we first identify fog use case characteristics that can be applied to assess a specific use case. In addition, we determine the tasks and corresponding roles that fog systems usually must fulfill depending on the use case characteristics. We then derive and iteratively refine architectural patterns regarding how to implement individual

tasks and roles depending on the use case. Researchers and practitioners can apply these patterns to design new fog systems starting from a structured categorization of their use case and definitions of the tasks and the role that the fog system is intended to fulfill; they can then obtain the recommended way of implementing the system by applying software design patterns.

II. FOG COMPUTING AND RELATED RESEARCH

A. INTRODUCTION TO FOG COMPUTING

CISCO was one of the first companies to introduce the concept of fog computing to support edge devices with separate physical nodes and provide additional computing, storage, and networking capabilities in geographic proximity [4]. We define the layer of fog nodes together with its relationships to adjacent edge and/or cloud layers and interacting end users as a *fog system*. Then, we interpret fog systems as sociotechnical IT artifacts to highlight that individuals and technologies interact to fulfill the use case objectives [18]. A fog system has entities such as edge devices, fog nodes, or cloud services that can request or fulfill tasks and end users that can only request tasks. Fog nodes themselves can have processing, storage, and networking resources [4]. As edge devices, we include any sensors, mobile devices, vehicles, etc., that have the ability to sense and/or act, are deployed at the edge of a network, and often have direct contact with users.

Fog nodes enhance the cloud infrastructure in a continuum of both varying geographic distances and extents of capabilities because fog nodes can be placed either directly next to edge devices or at greater distances to support a greater number of devices in a specific area [19]. Therefore, fog nodes can process data and workloads from the edge with much lower latency than can the cloud [20]. For instance, network service providers can enhance roadside units across a city with additional processing capabilities to take over the resource-intensive task of sensor fusion and object detection for several vehicles [21]. In addition to improving latency, fog nodes can reduce the data loads that must be sent to the cloud by aggregating and filtering the data received from the edge [22]. For use cases with sensitive data, fog systems can also enhance security and privacy across the system, keeping data and their processing steps close to their origin and usage [23]. Given the clear opportunities of fog systems, determining how to design them in an effective manner has become crucial [24].

B. DRIVING FORCES FOR FOG SYSTEM DESIGN

Reviewing related fog computing research as well as surveying fog systems operated by early adopters reveals two key driving forces for fog system design: (1) the characteristics of a use case and (2) tasks that fog nodes should perform as intermediates in the cloud-edge continuum.

1) USE CASE CHARACTERISTICS AND THEIR IMPACT ON DESIGNING FOG SYSTEMS

As is common for technologies, fog systems are most impactful when their system architecture truly fits the foreseen use case and its contextual conditions. Fog systems are typically applied to use cases in which edge devices with limited capabilities face extensive data loads and processing tasks with low latency requirements in a dynamic environment [25]. However, the specific use case context can vary, having a significant impact on the effectiveness of a fog system architecture [9]. For instance, an architecture that is built for a fixed and direct relation between edge devices and fog nodes and has no interconnections between fog nodes performs very efficiently for constant data loads between edge devices and fog nodes. In contrast, such a design struggles to handle fluctuating loads, which would benefit from interconnected fog nodes that can forward tasks to less utilized fog nodes in times of local peak loads.

However, prior research rarely characterizes the use case regarding the specific contextual conditions that significantly influence the effectiveness of architectural design decisions. Existing descriptions of use case characteristics are often limited to the physical and communication architecture (e.g., [26]) or the general services required (e.g., [27]). We argue that a nuanced characterization of fog use cases in terms of physical, operational, and strategical aspects supports industry-agnostic knowledge exchange and lays the foundation for the effective design of fog systems with architectures that fit their use cases.

2) TASKS PERFORMED BY FOG NODES TO ACHIEVE USE CASE OBJECTIVES

The determination of what the entire fog system and each individual node will do in a specific use case has an equally significant impact on its effectiveness. When placed between edge devices and the cloud, a fog node can take over tasks from both layers. Tasks describe categories of (recurring) activities that must be performed by one or several fog nodes to fulfill the purpose of the system in the context of a specific use case [28]. In that sense, tasks can include both direct processing support for the edge or cloud and supporting activities such as ensuring a balanced load across the system. For instance, a fog node task can include the real-time analysis of sensor data to detect abnormalities [12] or the authentication of new fog nodes joining the system [29]. Tasks can be inherently defined in nodes when the fog system is set up (e.g., the task in which routing tables are constantly optimized [30]) or flexibly submitted by entities (e.g., users can request sensor data aggregation [31]). We define a job as a specific instantiation of a task, that is, a runtime manifestation of a fog node actively processing data, engaging in communication, or triggering an action from other entities in the system. A fog system can consist of one or more physically separated fog nodes that can individually fulfill either all or individual tasks.

Similar to the characteristics of a use case, the tasks to be fulfilled by the fog system significantly influences the validity of potential fog system architectures. Returning to the aforementioned example of fluctuating loads, assigning the task of orchestration to the fog system to homogeneously distribute all loads and ensure their timely fulfillment opens up the

architectural option of having a centralized fog orchestration node. Therefore, we argue for the importance of a generic description of fog system tasks as the foundation for choosing tasks to fulfill specific use case objectives and building architectural decisions on these objectives.

Existing research on fog tasks either takes a detailed technical perspective on specific tasks (e.g., [17], [32]), defines high-level categories of tasks (e.g., [3], [33]), or focuses on an anecdotal description of specific application scenarios for a certain industry (e.g., [34], [35], [36]). However, prior research has not generalized tasks across industries and use cases. While prior research has contributed valuable details on the potential implementation of fog systems, a technical perspective instead of a use-case-centric perspective has been adopted, and the effective sets of tasks that a fog system can, cannot, or should perform depending on the use case have not been discussed. In addition, research often focuses on specific industries when fog systems are implemented, thereby impeding cross-industry knowledge exchange for use cases that share the same characteristics. We argue that the structuring of tasks for fog systems must be led by generic and industry-agnostic use cases to provide guidance for implementing effective fog system architectures and evolving them across industries.

C. VALUE OF ARCHITECTURAL DESIGN PATTERNS

Considering both driving forces and to answer our research question on how to design effective fog systems, we require a structured approach to documenting architectural details. Software design patterns (SDPs) are helpful tools for this purpose because they consist of a generic description of a recurring problem together with a proven solution [37], [38]. SDPs can be distinguished into varying levels of abstraction, among which architectural patterns are the most frequently analyzed [38]. Architectural patterns describe “[...] a fundamental structural organization or scheme for software systems and provide a set of predefined subsystems, specify their responsibilities, and include rules and guidelines for organizing the relationships between them” [32, p.12]. Considering the current maturity of the fog computing domain and our objective of developing guidance on how to design fog systems, we focus on architectural patterns.

In the context of fog systems, researchers have drafted initial architectural patterns. For instance, Seitz et al. (2018) proposed patterns for a specific use case in smart buildings and focused on the decoupling of edge devices from centralized fog nodes to enable real-time applications in IoT and cyber production system environments [39]. While they contributed a valuable architectural pattern for the specific low-latency, real-time, and high-availability IoT use case, there was less emphasis on the abstraction of this use case across industries. Related research (e.g., [40], [41], [42]) has provided comprehensive suggestions for fog implementations but has rarely compared design options; the broader context of a use case and consideration of the applicability of different architectural patterns for the use case have been neglected. We therefore argue that the design of an effective system for a

specific use case requires generic architectural patterns to choose from. Related fog computing research and surveys of fog systems are needed.

III. RESEARCH METHOD

To develop architectural patterns guiding the effective implementation of fog systems, we applied a multimethod approach comprising descriptive literature reviews [43] augmented by expert interviews [44]. The data collection and analyses were performed iteratively. First, we reviewed the literature on fog system implementations extensively to derive and synthesize fog use case characteristics and tasks (Section III.A). We then developed our set of SDPs to design fog systems (Section III.B). We conducted multiple iterations to validate and extend our findings, particularly by incorporating practical knowledge and feedback from an expert panel (Section III.C).

A. IDENTIFYING AND SYNTHESIZING FOG USE CASE CHARACTERISTICS AND TASKS

1) CONDUCTING TOP-DOWN AND BOTTOM-UP LITERATURE REVIEWS

We performed a descriptive literature review to identify fog tasks and typical use cases for the implementation of fog systems and to inform our SDP development (Section III.B). We followed extant guidance on conducting literature reviews (e.g., [45], [46]). We focused on conference and journal articles published in IEEE Xplore, ACM Digital Library, EBSCOhost, ScienceDirect, and ProQuest to cover relevant literature in the domains of cloud, fog, and edge computing. To consider both dedicated perspectives on use cases and fog tasks as well as fog system architectures, we combined top-down and bottom-up literature searches. For the top-down search, we aimed to identify related research explicitly examining tasks and therefore applied the broad search string “(fog AND (role or task)) NOT (scheduling* OR prioritization* OR offloading* OR allocation* OR placement* OR distribution*)”. We only selected articles using these terms in their titles due to the generic use of the term “task”. We focused our research on the architecture of fog systems more than on the operations of fog systems and therefore excluded literature related to task scheduling and prioritization, among others.

To complement this search, we performed an additional bottom-up search focusing on fog architectures. We applied the following search string: “fog AND (network OR system OR computing) AND (architecture OR design OR setup)”. This approach permitted to examine the proposed fog systems and derive tasks while simultaneously gathering data on how the fog system is implemented to meet use case requirements, providing the required basis for developing SDPs.

These literature searches resulted in 553 articles that we checked for relevance. We removed four books, 75 duplicates, 51 articles focusing on topics other than fog computing (e.g., literature discussing biological fog), 104 articles applying a generic fog architecture, 168 articles focusing on the processing operations of fog systems, 26 articles providing a survey or taxonomy of fog computing, and 19 articles focusing

on radio access networks (RANs). We excluded works on generic fog architectures because they discuss potential applications but not specifically how fog systems can take on certain tasks under certain (use case) conditions. We also excluded RAN-focused articles because they focus on very specific details of fog computing technologies, such as network transmission bandwidth allocation [47]. The relevancy check resulted in 150 remaining articles that we analyzed in detail, as described below.

2) SYNTHESIZING USE CASE CHARACTERISTICS

We thoroughly analyzed the 150 relevant articles by applying a structured coding approach to iteratively identify, structure, and refine use case characteristics and tasks [48]. We also noted implementation specifics for fog systems to develop our SDP. To identify use case characteristics, we coded varying contextual conditions that guided prior research in designing their systems and choosing certain tasks. For example, Khazael et al. [49] proposed a fog implementation for a smart city monitoring application and described the related use case characteristics, among other factors, as challenging “network dynamics (e.g., the leaving and joining of edge nodes)” (p. 143150) [49]. We coded these use case characteristic as “open system”.

We recognized that use cases typically differ in terms of (1) the involved system entities, their relationships, and dynamics; (2) the entities’ capabilities and functions (e.g., the processing power of edge devices); (3) the contextual conditions (e.g., whether the system is open or mobile); and (4) the intended objectives (e.g., support for ultra-low latency or large-scale systems). We aggregated the identified use case characteristics (n=217) into seven categories (Section IV.A), which helped us distinguish different use cases across industries as well as identify similar ones.

Analyzing the literature also revealed the secondary role of cloud services and their characteristics in fog applications. Prior research defines fog use cases largely independently of the available cloud setup. Clouds are usually characterized as a ubiquitous source of additional processing or long-term storage capacity that can support the fog but is not expected to do so in regular operations. Our categories therefore do not include characteristics related to cloud services but rather those related to edge devices and fog nodes.

3) SYNTHESIZING TASKS

We used a bottom-up approach and started with the fog architecture-focused articles to uncover and synthesize tasks. We coded every activity and task that a fog node was designed to fulfill and noted a title, a description, and its source. For instance, Taherizadeh et al. [50] proposed a fog system that processes data, prioritizes processing tasks, and assigns processing tasks to fog nodes depending on their utilization. Accordingly, we created three new codes for tasks that fog nodes can perform. For the subsequent articles that we coded, we either defined a new task if an activity described by the authors was not already in our list of tasks or assigned an existing task from our codebook to aggregate findings across articles.

We also adapted tasks to aggregate the descriptions of similar activities for fog nodes. For instance, we combined the tasks of “process container” and “process virtual machine” to “process virtualized job” and distinguished that task from “process data”, as the former demands more flexibility from a fog node and imposes higher requirements. To achieve sufficient abstraction and generalization, we aggregated tasks and developed a structure of tasks and subtasks, for instance, summarizing all activities such as data caching, aggregation, and filtering under the subtask “process data”, which belongs to the task “processing”. In total, we coded 227 activities, which we aggregated into ten subtasks and four tasks (Section IV.B.1). As an illustration, Table I shows how coded activities are documented and assigned to a subtask and task. Figure 2 in Section IV.B.1 presents the structure of subtasks and tasks. Although we noted how often an activity or (sub-)task was coded, we did not assess its relevance based on the number of codes. Rather, we evaluated the importance and relevance of each activity or (sub-)task individually to consider the broad spectrum of possibilities and fog system manifestations.

TABLE I
Summary of iterations to validate and extend the research results

| Code Nr. | Excerpt | Activity | Subtask | Task |
|----------|---|---------------------------------------|----------------------|---------------|
| 151 | “the fog node to process the received data and detect relevant information” [51, p.6] | Analyze data to detect patterns | Process data | Processing |
| 172 | “check the fog nodes in the entire network and place application modules based on the available capacity” [52, p. 6065] | Assign jobs according to availability | Administer resources | Orchestration |

While aggregating tasks and subtasks, we noted that certain combinations of tasks occurred regularly. We therefore assessed all similar combinations of tasks iteratively to identify their commonalities and assigned generic roles to describe the key responsibilities of such fog systems across cases and contexts. For example, we identified the role of “smart support”, referring to fog systems that offer processing support for edge devices as well as orchestrate jobs. Overall, we identified six roles that fog systems usually fulfill. We also identified optional tasks that a role can but does not have to perform. For the role “smart support”, for example, we identified several proposed fog systems that also track suspicious activities to identify and stop attacks on the system and added this subtask as optional for the role. Through this mapping of tasks and roles, we ultimately derived a task–role hierarchy for fog computing, which is presented in Section IV.B in detail.

B. DEVELOPING SOFTWARE DESIGN PATTERNS

To develop SDPs for fog systems, we followed seminal articles proposing common pattern structures, comprising the name, context, examples, problem, forces, solution, dynamics, variants, consequences, and known uses of a pattern [38], [53],

[54]. For each pattern, we carefully specified each structural dimension based on extant research, informal discussions with practitioners and researchers, analyses of existing fog systems, and our own experiences, as outlined below.

First, for each SDP, we defined a meaningful name [54]. If suitable, we aligned the name of the derived SDP with the terms used in research proposing (similar) fog architectures (e.g., the SDP “edge controller”, focusing on fog nodes that primarily control edge devices).

Next, we described the context that suggests the SDP’s applicability, in which the problem and its solution recur [54]. For each SDP, we discussed the use case characteristics, challenges, and solutions to which the pattern relates. For example, we identified a single-layer processing assistance fog system (architectural pattern 1, AP1) as a promising SDP for use cases requiring ultralow latency responses. For each SDP, we determined whether it can fulfill our categories of use case characteristics. This mapping helped us ensure that our SDP met the key use case requirements and would support system designers in determining when to apply this SDP.

We added one or more brief use cases as examples for the context to increase the understanding and applicability of SDPs. For example, a single-layer processing assistance fog system (AP1) seems promising for sensor edge devices that monitor conditions (e.g., a fire hazard in a production environment) where fog nodes aggregate and process data from multiple sensors in a short time so that they can react quickly (e.g., to fire threats).

Afterward, we defined a problem and its corresponding forces for the given context [54]. A problem outlines the challenges for fog system design imposed by the use case characteristics and may also comprise preconditions and boundary conditions that the SDP should fulfill. Forces reveal the details of a problem and define the kinds of trade-offs that must be considered in the presence of tensions between use case characteristics, system objectives, and the capabilities of edge devices and fog nodes, among others [54]. For example, in a large system with heterogeneous devices and processing requests with varying priorities, tension arises from the need to immediately process a request and the need to prioritize requests before processing in case of processing capacity shortages. In describing the forces, how they interact, and the resulting trade-offs, we reveal the objectives to be achieved by using the SDP [54]. Analyzing problems and corresponding forces also helped us compare different solutions and their appropriateness for a specific use case.

Next, we focused on describing a solution that included the fog system entities, their (static and dynamic) relationships, and the functions needed to achieve the desired outcome [54]. To define an appropriate solution (i.e., one that fulfills the SDPs’ objectives while considering the problem and forces), we turned to the analyzed literature to conceptualize the task–role hierarchy. For each role, we examined the coded articles, compared their fog system designs and proposed architectures, and considered use case characteristics. While focusing on abstraction through a pattern, we combined the most common similarities and core features among the proposed fog

architectures for a specific role in one SDP. If the specific architectures varied across articles, we defined additional patterns. For example, prior research has proposed single- and multilayer architectures to support the role of processing assistance. We compared and synthesized this information, revealing that a multilayer architecture is effective for use cases with larger system sizes and a greater job variety; therefore, we developed two SDPs.

As the next step, we outlined the resulting dynamics that describe the system state and the interactions of entities after applying a pattern [54]. We carefully considered and explained how edge devices communicate with fog nodes, which tasks the fog node performs, and how the fog node interacts with cloud services, if necessary. This description of the dynamics helps substantiate the solution by justifying how and why the solution works and determines its forces to align with the desired objectives [54]. Since we built our SDPs on justificatory knowledge from extant research, we aimed to summarize the assumptions regarding why the SDP works as a solution.

A comparison of the extant research on fog role implementations and their dynamics revealed variants of the solutions, which we then described. For instance, in a multilayer processing assistance fog system (AP2), fog nodes can either only communicate vertically (with lower and higher layer fog nodes) or can also communicate horizontally (with similar layer fog nodes). Detailing such variants supports system designers in resolving trade-offs between forces and provides guidance on how to implement a solution.

In addition, we elaborated on the consequences of applying a pattern [54]. We reflected on the benefits (e.g., problems solved, enhancements, and achievements) and liabilities (e.g., reduced cost efficiency). In this way, we want to emphasize the value of the pattern and raise system designers' awareness of potential drawbacks that they need to consider.

We finally listed example research articles to show that the SDP is a (potential) solution for a problem and reference the literature used to build the pattern.¹ We thereby provide a foundation for interested readers to explore a specific solution more deeply. By applying this canonical structure to SDPs, we want to improve the understanding and usage of patterns for fog system designers and foster pattern applicability.

C. VALIDATING AND EXTENDING FINDINGS

We built the use case characteristic categories, task-role hierarchy, and SDPs on the data and findings from extant research, complemented by informal discussions with practitioners and researchers and reviews of existing fog systems. We conducted five iterations of evaluations to validate and extend our preliminary findings and ensure correctness, comprehensibility, and practical applicability. The iterations comprised a focus group interview, written feedback from experts, one-on-one interviews, quality criteria evaluations, and a final expert review (Table II). We did not weigh the literature and expert

opinions quantitatively but rather focused on iteratively improving and detailing the architectural patterns (also considering that the experts generally confirmed the patterns derived from literature) and providing variants of each pattern in the case of deviating approaches for their implementation.

TABLE II
SUMMARY OF ITERATIONS TO VALIDATE AND EXTEND RESEARCH RESULTS

| Iteration | Objectives | Evaluation | Outcome |
|-----------|---|-----------------------------------|---|
| 1 | <ul style="list-style-type: none"> Review task-role hierarchy Present structure of architectural patterns Onboard experts | Focus group interview | Joint understanding, initial insights from practice |
| 2 | <ul style="list-style-type: none"> Verify task-role hierarchy Incorporate feedback on patterns Incorporate practical knowledge | Written feedback from experts | Generally improved tasks, roles, and patterns |
| 3 | <ul style="list-style-type: none"> Reiterate architectural patterns Strive for theoretical saturation | Semi-structured expert interviews | Practically refined patterns |
| 4 | <ul style="list-style-type: none"> Reassess architectural patterns against SDP quality criteria | Quality criteria related to SDP | Theoretically refined patterns |
| 5 | <ul style="list-style-type: none"> Finalize architectural patterns | Expert reviews | Finalized patterns |

We gathered a panel of eight experts to evaluate our preliminary findings and incorporate practical knowledge (Table III). We chose experts from different industries and companies of different sizes to validate the applicability of our architectural patterns across industries. We included only experts who had sufficient expertise and who either had implemented a fog system or planned to do so soon.

TABLE III
INTERVIEW EXPERT PANEL

| Expert | Industry | Role | Fog Expertise |
|--------|-----------------------------|------------------------------|---------------|
| I | Agricultural machinery | Senior Director Corporate IT | >15 years |
| II | Power tools | Product Manager IT Solutions | >5 years |
| III | Edge IT infrastructure | Head of Connected Devices | >10 years |
| IV | Optoelectronics | Head of Innovation & DT | >15 years |
| V | Hospital IT infrastructure | Chief Technology Officer | >5 years |
| VI | Digital consulting | Expert Associate Partner | >10 years |
| VII | Edge & cloud infrastructure | Chief Technology Officer | >5 years |
| VIII | Research | Research Engineer | >5 years |

¹ We list known uses to provide practitioners and researchers with examples and information for further reading. Although referenced articles often discuss the benefits and shortcomings of their fog system architectures, they

rarely provide information on how to gauge the success of their implementation.

1) FIRST ITERATION: FOCUS GROUP INTERVIEW

For the first iteration of the evaluation, we conducted a focus group interview [44] with the expert panel. We aimed to (1) establish a joint understanding of fog computing and familiarize the experts with our research results; (2) validate our literature findings, as qualitative coding techniques are vulnerable to subjective interpretation and other biases; and (3) incorporate knowledge from experts in the field to extend our findings. We focused the interviews on an in-depth discussion of fog computing in general and related use cases, characteristics, tasks, and roles while only briefly presenting the architectural patterns that would be the focus of follow-up iterations. Focus groups enable participants to engage in thoughtful discussions and generate valuable and extensive data [44].

We conducted the focus group interviews based on an interview guide, such as explaining our task–role hierarchy and each role in detail, followed by facilitating a discussion to validate and gather additional data surrounding the use case characteristics, tasks, and roles. The interviews lasted 1.5 hours. We took field notes, created protocols from the feedback, and considered experts' questions for clarification to increase the comprehensibility of our descriptions.

During the focus group interview, we applied a nonjudgmental form of listening, maintained distance, and strove to maintain an open and nondirective style of conversation to ensure impartiality [44]. We were also sensitive to ethical concerns and (1) balanced anonymity and disclosure; (2) ensured that field notes and other data were kept secure; and (3) respected participants' knowledge and time (i.e., we scheduled meetings to fit the interviewees' schedules and frequently acknowledged their efforts).

Overall, the experts confirmed the categories of use case characteristics and the content and structure of subtasks and tasks relevant for fog computing. Based on the expert questions, we enriched some of the subtask descriptions. For instance, expert IV described the relevance of fog nodes for distributing updates to edge devices and keeping their firmware at the latest version to close security gaps. Accordingly, we incorporated this aspect into the "Manage attacks" subtask by adding that fog nodes could "Implement measures to prevent, identify and stop malicious behavior and external or internal attacks."

For the roles and related architectural patterns, the experts also confirmed the practical relevance of the processing capabilities of fog systems and the differentiation we established based on industry-agnostic sets of use cases. However, they disagreed on the notion of fog nodes as pure security enforcers and the definition of a layer that purely manages the network as a fog layer. Therefore, we adjusted our definition of fog computing to align with its key purpose of supporting edge devices handling demanding tasks and data loads and kept only the roles and architectures that involve subtasks from the "Processing" category. In particular, we degraded two basic roles, namely, security enforcer and network controller (see Appendix A).

2) SECOND ITERATION: EXPERT EVALUATION ('HOMEWORK')

After the focus group interview, we asked the experts to familiarize themselves (as 'homework') with the task–role hierarchy, the use case characterization, and the SDPs to prepare for subsequent interviews and to send us written feedback in advance. We also asked them to compare our findings with their individual use cases and fog systems. For this purpose, we shared a comprehensive description of our refined results and asked for feedback without providing a specific structure, because we wanted the experts to focus on the aspects they were most familiar with.

We gathered the feedback and made further adjustments to our task–role hierarchy and SDPs. For instance, expert II described how the fog systems he supports are similar to processing assistance SDPs but often do not include a permanent relation between edge devices and fog nodes; instead, there can be periodic reconfiguration. We added this as an option for the corresponding SDPs. Based on the questions shared by the experts, we also refined certain descriptions. For example, questions regarding the distinction between edge devices and fog nodes led to a more extensive differentiation of the two in terms of activities, capabilities, and position in the fog system.

3) THIRD ITERATION: EXPERT INTERVIEWS

Next, we conducted eight semistructured interviews with the experts from our panel to enable detailed discussions and gather rich data [44]. We created an interview guide to ensure a structured approach in the interviews [55] that addressed (1) the expert's use case and their fog system, (2) the system's detailed technical implementation, and (3) a review of the most fitting SDPs. First, we asked for a detailed description of the use case so that we could categorize it according to the use case characteristic categories. Subsequently, we discussed the details of the implementation of the proposed fog system in terms of its architecture; the dynamics between the edge, fog, and cloud; the specific tasks that the fog nodes fulfill; and the challenges in the implementation and operation of the fog system. We then jointly went through our SDPs and asked the experts to either select SDPs that were similar to their architecture or propose a new one if they had the impression that their implementation was significantly different. As every expert identified a fitting SDP, they outlined similarities and differences with their architecture and provided valuable feedback on how to improve our patterns and use case characterization.

Based on the interview notes, we iteratively refined our SDPs individually and overall. For instance, based on the feedback from experts II and III, we extended the possible contexts in which the single-layer processing assistance architectural pattern should be applied to "highly cost-efficient use cases" because the pattern can be implemented not only to reduce latency but also to reduce costs. In addition, we added the design option whereby edge devices do not necessarily need a constant connection to their related fog nodes because they can also transfer data batchwise. For architectural patterns related to the role of smart processing support, for instance, expert I reviewed the centralized orchestration pattern and emphasized the importance of the fact that the

orchestrating node should only orchestrate and not take over processing tasks. In addition, this expert described their approach to implementing one large physical fog node with separate virtual machines and split responsibilities for orchestration and processing, which we then added as a variant of the pattern. Overall, the expert interviews supported us in refining all patterns while adding valuable contributions, especially to the variants.

4) FOURTH ITERATION: QUALITY CRITERIA EVALUATION

At this stage of the research, we decided to perform a more formal evaluation of our SDPs by assessing whether the patterns fulfilled the common quality criteria proposed in prior research. In total, we assessed seven criteria for the evaluation of the SDPs, grouped into three categories: pattern design, perception, and utilization [56].

Regarding pattern design, we assessed whether our architectural patterns were complete, consistent, positively framed, and coherent to ensure that they fulfilled all formal criteria imposed on SDPs [53], [57], [58]. Regarding perception, we assessed the relevance, understandability, adoptability, and usefulness of our patterns to ensure that they would provide benefits for actual cases and common use cases while not being described too specifically with certain industry colloquia [58], [59]. Regarding utilization, we assessed the concreteness, effectiveness, usability, and level of abstraction to find a careful balance in our targeted industry-agnostic method while still providing pattern descriptions specific enough to be directly applicable and useful [58], [60]. We also assessed the comprehensiveness of the SDPs to ensure sufficient information for developers and architects to understand potential trade-offs in the selection of certain SDPs.

Based on the assessment performed by the team of authors and an additional independent fog expert researcher, we noted the potential to improve the consistency of our patterns in terms of the terminology used and to increase concreteness while decreasing the level of abstraction for some patterns. Accordingly, we performed another iteration of revising our patterns before the final phase was initiated.

5) FINAL ITERATION: EXPERT REVIEW

As a final evaluation iteration, we decided to send our article and SDPs to the expert panel for final review and ask them once more if they had anything to add or suggestions on how to improve our findings. In both cases, the experts agreed that they were not aware of further improvements.

IV. A USE-CASE-CENTRIC PERSPECTIVE ON FOG SYSTEMS

For the selection of an effective fog architectural pattern, system developers and IT architects need to first assess their use case and the manifestations of its characteristics and then determine the tasks and roles that the future fog system needs to fulfill (Fig. 1). We next provide a use-case-centric perspective on the development of fog systems to guide SDP selection.

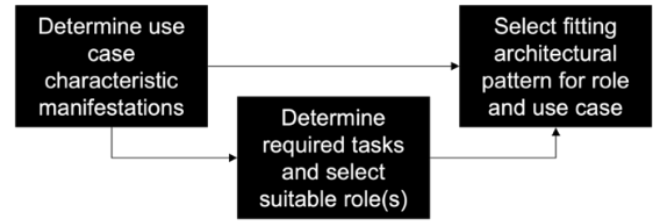


FIGURE 1. Approach to selecting an architectural pattern.

A. FOG COMPUTING USE CASE CHARACTERISTICS

We identified seven categories of characteristics that can be used to classify industry-agnostic fog use cases (Table IV):

Latency criticality describes how urgently edge devices require responses to their job. This can be ultralow, in which the latency must be in milliseconds, or moderate when higher latencies are tolerable, such as hundreds of milliseconds [61], [62].

Job variety describes the range of jobs that are submitted by edge devices. It can be narrow if edge devices submit only one type or a few very similar types of jobs (e.g., the same data processing tasks consistently) or broad if the job types range from simple data processing to containers or VMs that are submitted for extensive processing [63], [64].

TABLE IV

OVERVIEW OF FOG USE CASE CHARACTERISTICS AND THEIR MANIFESTATION

| Category | Description | Continuum | |
|-----------------------------|--|---------------------------------|--------------------------------|
| | | from... | ...to |
| Latency criticality (C1) | The degree of urgency of the response to the jobs of edge devices. | Moderate (<100s ms) | Ultralow (<5 ms) |
| Job variety (C2) | The range of jobs that are submitted by edge devices. | Narrow (1–3 types) | Broad (>3 types) |
| Load dynamics (C3) | The frequency and size of job submissions by edge devices. | Stable (+/-10% of avg.) | Varying (+/-80% of avg.) |
| System borders (C4) | The degree of restrictions regarding which new entities can join and work in the fog system. | Private (one organization) | Public (several organizations) |
| Geographical mobility (C5) | How far edge devices and fog nodes physically move relative to each other. | Limited (within several meters) | Unlimited |
| Edge device capability (C6) | The extent of processing, storage, and networking capabilities of edge devices interacting with fog systems. | Low (device worth several €) | High (device worth 100s €) |
| System size (C7) | The numbers of edge devices and fog nodes that participate in the fog system. | Small (few) | Large (1000s) |

Load dynamics describe how regularly jobs are submitted and how consistent their sizes are. The dynamics can be either stable when edge devices send similar-sized jobs in terms of processing extent on a fixed schedule or varying when jobs are sent on demand and irregularly and have different sizes [50], [63].

System borders describe whether the fog system is private or public and to what extent new entities are restricted

in their interactions with the system. Private in this sense means that all edge devices, fog nodes, the cloud, and accessing users belong to one organization (or a community of organizations), and no external entities can access the system without permission. Public means that any edge device or fog node can join or leave the system flexibly and that anyone can access the system [65]. Between these extremes, the degree of restriction can vary; e.g., new fog nodes can join the system only after an additional registration and authentication process [66].

Geographical mobility describes the magnitude of the distances that edge devices and fog nodes can physically move relative to each other. Mobility can be limited when edge devices and fog nodes stick to their designated locations, with only slight movement of several meters or joint movement, e.g., when they are implemented in a larger system such as a ship. Mobility can also be unlimited if edge and fog nodes move greater distances on a regular basis (e.g., smart vehicles), hence decreasing the geographical proximity of fog nodes and edge devices [67], [68], [69].

Edge device capability describes the processing, storage, and networking capabilities of the edge devices interacting with the fog system [3]. It can either be low if the edge devices are rather limited and cheap or high if the edge devices are very resourceful, often costing 100s of euros.

The **system size** describes the numbers of edge devices and fog nodes that participate in the system [8]. It can either be small if only a few devices and nodes participate or large if thousands of devices and nodes participate.

B. FOG TASK-ROLE HIERARCHY

Fog nodes need to perform varying tasks and can take on different roles depending on the use case characteristic categories that apply. We next describe the fog task-role hierarchy and outline when each role is effective in addressing use case characteristics.

1) TASKS OF A FOG NODE

To allow a structured relation between use cases (i.e., the general context for the fog system to work in) and the roles of a fog system (i.e., the specific purpose a fog system has to fulfill in the context), we derived four key tasks that fog nodes can perform: processing, orchestration, securing, and networking (see Table V).

2) ROLES OF A FOG SYSTEM

We identified six industry-agnostic fog system roles based on commonly occurring sets of tasks that a fog system fulfills in particular use cases (Fig. 2). We briefly summarize the four predominant roles in what follows; fog nodes taking the roles of sole security enforcers and network controllers (Appendix

TABLE V
OVERVIEW OF FOG NODE TASKS AND SUBTASKS.

| Task | Subtask | Description | Examples |
|---------------|--|--|--|
| Processing | Offer software-as-a-service (SaaS) to process data | Process data and store or forward the outcomes (only the data for processing are submitted to the fog node) | Sensors spread across a city sector send traffic data to a fog node for aggregation and provide recommendations to autonomous vehicles for preferred routes [70]. |
| | Offer infrastructure-as-a-service to process virtualized jobs | Run containers and VMs on demand and follow flexibly defined instructions (data and instructions submitted to fog nodes) | Microservices architecture allows fog nodes to address fast-scale IoT applications with heterogeneous devices [71]. |
| Orchestration | Prioritize jobs from different entities | Manage a queue of jobs from different entities, assess criticality and assign jobs according to their positions in the queue | Fog nodes as intelligent controllers establish a multiqueue architecture for large-scale task prioritization [72]. |
| | Administer resources of all entities | Monitor all available processing resources from entities and current and predicted jobs and distribute jobs according to quality-of-service requirements | Central fog node monitors the utilization of all participating fog nodes in real time to assign jobs to underutilized nodes or outsource jobs to the cloud if all fog nodes are highly utilized [50]. |
| Security | Manage internal and external attacks | Implement measures to prevent, identify, and stop malicious behavior and external or internal attacks | Fog nodes shield the edge layer from outside influences to stop, e.g., DDoS attacks [73]. |
| | Manage identities of all entities | Manage all entities in terms of their registration, authentication, emergency control, trust-management, etc. | Central fog node hosts a registry of edge devices, fog nodes and cloud services in the fog system so that every resource needs to authenticate with this node before it can submit, e.g., a processing request [31]. |
| | Preserve data and location privacy | Implement mechanisms preserving privacy across the fog system with role-based access control and encryption | Fog node encrypts data from sensors before sending it through the internet to processing fog or cloud nodes [75]. |
| | Ensure nonrepudiation and immutable traceability | Host infrastructure to track entities and activities reliably and consistently | Fog nodes host a distributed ledger and mining infrastructure to document each transaction in the fog system [76]. |
| Networking | Manage network connections within and adjacent to the fog system | Manage connections among edge, fog and cloud resources dynamically and handle the mobility of nodes and the handover of jobs (the fog node acts as a software-defined networking controller) | Drones as flying fog nodes support edge devices and need to continuously manage their connections and offloading as well as handover decisions [77]. |
| | Distribute traffic within the fog system | Monitor traffic across the entire network and optimize connections, the communication strategy and handovers accordingly | Fog node continuously assesses the network topology and related traffic to optimize routing paths to obtain short distances and low energy consumption [30]. |

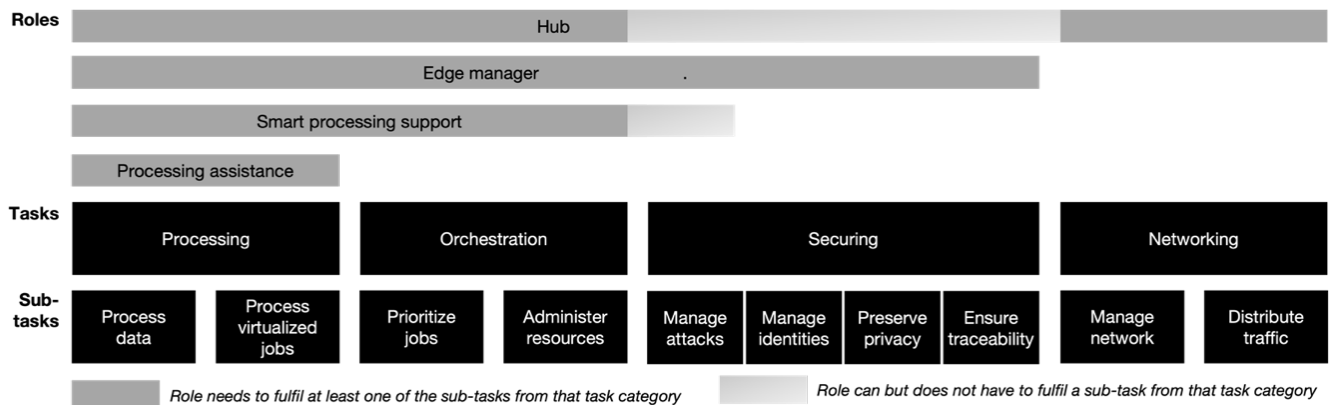


Figure 2. Hierarchy of fog system roles and related (sub)tasks.

A) are theoretically feasible but were not regarded as effective by our interviewees.

Processing assistance. The fog system focuses on the processing of data in a predefined way or on the processing of virtualized jobs (e.g., containers, VMs). Every fog node in the system supports one or several edge devices and can also take over tasks from the cloud or users. For example, a fog system can support a smart city in locally processing sensor data for identifying immediate threats with very low latency (e.g., [70], [71]). Potential implementations of processing assistance extend from the immediate local processing of healthcare data on one layer [71] to multilayer fog architectures in which lower-level nodes perform basic analysis and forward the data to higher-level nodes for complex analysis [70].

The processing assistance role is favorable for use cases that have constant data loads (use case characteristic category C3) and process unchanging jobs (C2) but require ultralow latency (C1) and therefore cannot afford to carry out any prioritization or assignment steps for the data before jobs are processed. Processing assistance allows for a low-cost architecture, as all fog resources can focus on processing without performing overhead tasks such as monitoring resources or prioritizing jobs.

Smart processing support. A fog system offering smart processing support not only processes data or virtualized jobs but also orchestrates jobs and available resources across edge, fog, and cloud entities. Both processing and orchestration can be performed in the same fog node or in separate nodes. Orchestration ranges from prioritizing, queuing, and assigning jobs up to the full monitoring of resources, load predictions and balancing, as well as resource spinup and spindown. Smart processing support can also include assessing the security of submitted jobs before they are assigned to a processing fog node. The smart processing support role is often proposed and implemented in closed but volatile infrastructures such as the (industrial) IoT (e.g., [72]) or small smart grids (e.g., [73]). It can, for instance, manage cyber-physical power systems and can both monitor the power system to notify producers and consumers and orchestrate fog and edge resources [73]. However, smart processing

support can also be applied to larger systems such as smart cities (e.g., [74]).

Implementing the role of smart processing support is effective for use cases requiring volatile loads (C3), varying jobs, and changing priorities to be handled (C2). With a live overview of all resources and jobs, smart processing support can effectively address these flexibility challenges by distributing jobs according to their priority. Furthermore, it allows for better utilization of resources systemwide [50]. The general architecture of a fog system acting as a smart processing support with fog nodes dedicated to orchestration also fits use cases that need to scale quickly. In that case, new fog nodes can be connected with the orchestration node and start operating faster than they can in the role of processing assistance, in which fog nodes are assigned to a fixed number of edge devices and the whole topology may need to be changed if the system is scaled up.

Edge manager. A fog system fulfilling the role of edge manager extends the role of smart support by addressing tasks related to securing the system in addition to orchestration. In this way, such systems can potentially harness the idle capacity of edge devices with extensive capabilities by connecting them with less capable devices. However, the edge manager does not always perform comprehensive securing tasks. Often, it manages identities, prevents attacks, preserves privacy, or ensures traceability of activities close to the edge. Edge managers are applicable across many industries. For instance, a fog system proposed by Núñez-Gómez et al. [65] hosts a distributed ledger architecture for others to exchange processing capacity. Such a fog system could be applied in smart cities, the IoT, and other settings that require the tracking of microtransactions with many stakeholders who post and fulfill processing requests.

The role of edge manager is effective for use cases requiring an open system that must host external participants (C4) and requires low latency for administration (C1). In that case, the edge manager enables a dynamic system that can, for instance, add or remove third-party devices, nodes, and users flexibly while operating. The additional security measures also reduce risks if the fog system is allowed to control edge devices [75], [76].

Hub. As hubs, fog systems additionally manage advanced networking tasks such as the handover of jobs for mobile edge devices or the dynamic distribution of traffic. The orchestration of the system is spread across several fog nodes of a hub fog system that is usually larger. A hub often fulfills subtasks to secure the fog system. A hub mostly manages identities or ensures traceability. Smart vehicles are one of the key use cases for the hub role, as smart vehicles are essentially mobile and have large data loads and low latency requirements [21]. To implement a hub for smart vehicles, existing infrastructure such as roadside units can be enhanced to function as fog nodes for processing with an overarching, city-wide node that manages the network [21], [65]. This can also be implemented for unmanned aerial vehicles [77]. In addition, smart vehicles can act as fog nodes to support each other or edge devices with limited processing capabilities [78].

A hub is effective for use cases that either include geographically moving edge devices and fog nodes (C5) or create extensive network traffic and potential congestion. As it is close to devices with location awareness, a hub can monitor and predict the devices' positions as a foundation for an effective handover of processing jobs. Moreover, local fog nodes can collaboratively ensure up-to-date routing tables for mobile devices so that these can be accessed at any time. In the case of high risks of harm from network congestion, a hub can quickly react to sharply increasing network traffic by changing routes and data allocation.

C. DETERMINING THE CHARACTERISTICS OF USE CASES AND SELECTING EFFECTIVE TASKS AND ROLES


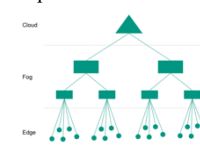
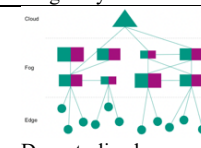
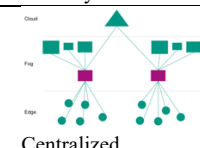
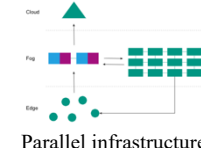

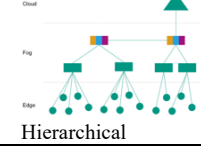
Practitioners and researchers should first characterize their use case by positioning its conditions and requirements between the two poles of each category-specific continuum (Table IV). Consider the following example: A fog use case involves numerous simple sensors tracking basic parameters in a building and requires fog nodes that aggregate the sensor data for real-time decision making. Such a use case requires ultralow latency (e.g., to instantly react to immediate threats), and the load of sensor data and processing steps will be rather narrow and stable [79]. Accordingly, such a use case is on the right side of the continuum for the category "latency criticality" (C1) and on the left side for the categories "job variety" (C2) and "load dynamics" (C3) in their respective continuums. As this is a closed local system with one host responsible for the infrastructure and sensors as well as fog nodes that do not move, the use case is on the left side of the continuums for the categories "system borders" (C4) and "geographical mobility" (C5). As this use case only considers a moderate number of simple sensors in one building, the "edge device capabilities" (C6) are low, and the "system size" (C7) is rather small. To illustrate and further guide the determination of use case characteristics, we derived four commonly occurring archetypes of fog use cases: efficient, flexible, open, and mobile systems (Appendix B).

As a second step, practitioners and researchers need to determine the tasks and role a fog system needs to take on depending on the use case and the manifestations. Specifically, if the characteristics of a use case are on the right side of the continuums for categories C1-C5, this influences the recommended choice of the fog tasks to be implemented. For instance, a smart manufacturing use case with production edge devices (such as logistics robots and machine tools) and fog nodes that automatically manage the manufacturing environment faces a broad range of different jobs (C2) and varying load dynamics (C3) [80]. Accordingly, the fog nodes also need to perform orchestration tasks related to their processing resources and the prioritization of edge device requests in times of peak loads to avoid long waiting times for critical requests (smart processing support role). Public system borders (C4) require the fog system to perform security-related tasks such as the management of identities or the preservation of privacy (edge manager role). Edge devices and fog nodes that move great distances (C5) mean that the fog system must manage the network around the edge devices, e.g., by tracking their positions and ensuring up-to-date routing tables for all edge devices and fog nodes (hub role).

V. ARCHITECTURAL PATTERNS OF FOG SYSTEMS

We developed seven architectural patterns to guide the implementation of fog systems (Table VI). These patterns represent possibilities for an effective architecture to implement a particular role and meet use case conditions (Table VII). Apart from the hub role, we identified different pattern variants based on the use case characteristics,

TABLE VI
SCHEMES OF THE VARIOUS ARCHITECTURAL PATTERNS

| Roles | Variations of architectural patterns* | |
|--------------------------|--|---|
| Processing assistance |  |  |
| Smart processing support |  |  |
| Edge manager |  |  |
| Hub |  | |

* Edge devices are represented as circles, fog nodes are represented as rectangles (green = processing, purple = orchestration, blue = securing, and orange = networking), and cloud nodes are represented as triangles.

TABLE VII
RELATION BETWEEN FOG TASKS, ROLES, DESIGN PATTERN VARIANTS, AND USE CASE CHARACTERISTICS.

| Critical Tasks | Processing | | Orchestration | | Security | | Networking |
|---------------------------------|----------------------|-------------|----------------------------|-------------|-----------------------------|----------------------|--------------|
| Favorable Roles* | Processing Assistant | | Smart Processing Assistant | | Edge Manager | | Hub |
| SDP | Single-layer | Multi-layer | Decentra- lized | Centralized | Parallel infrastructures | Temporary cluster | Hierarchical |
| C1: Ultralow la- tency | X | X | (X) | | | (X) | |
| C2: Broad variety | | (X) | | X | X | (X) | (X) |
| C3: Varying dy- namics | | | X | X | X | (X) | (X) |
| C5: Public system | | | | | X | X | (X) |
| C5: Unlimited mo- bility | | | | | | | X |
| C6: High device ca- pability | | | | | X | | |
| C7: Large system | | X | | X | (X) | | (X) |

* While some roles are in general broadly applicable, based on our literature review and expert interviews, we found specific “favorable roles” to be most efficient in certain use cases, as they only fulfill the minimum required tasks; this avoids unnecessary work and an increased overhead on the fog layer, which is more expensive than the cloud.

X – Role is effective for use cases with characteristics on the right side of the respective use case characteristic continuum.

(X) – Role can be applied in use cases with characteristics on the right side of the respective use case characteristic continuum.

particularly those related to edge device capability (C6) and system size (C7). For instance, the architectural patterns for the processing assistance role differ in their general configuration, either consisting of a single layer of fog nodes or several layers. If the use case includes a large system size (C7), a multilayer variant is more suitable. The supplementary material provides detailed descriptions of each architectural pattern. Next, we summarize each pattern.

A. ARCHITECTURAL PATTERNS FOR PROCESSING ASSISTANCE

1) SINGLE-LAYER PROCESSING ASSISTANCE

For simple and static use cases requiring a very low-latency (C1) or cost-efficient system, the single-layer processing assistance architectural pattern considers one layer of fog nodes that are directly connected to one or several edge devices and only focus on the processing of jobs (e.g., [26], [81], [82], [83]). With a direct relation between a fog node and one or several edge devices, there is a clear path of communication enabling the direct processing of jobs without additional overhead (C1) [26]. With static loads (C3) and regular tasks (C2), the number of fog nodes and their capabilities can be precisely sized depending on the number of edge devices connected and their data loads [82].

2) MULTILAYER PROCESSING ASSISTANCE

For static use cases with a broader variety of jobs (C2) and a larger system (C7), the multilayer processing assistance architectural pattern includes several layers of fog nodes with increasing capabilities on the higher layer(s) (e.g., [79], [84], [85], [86]). Multiple layers enable ultralow latency responses (C1) with direct processing on the lower layers while handling tasks that demand more processing on the higher layers in a cost-efficient architecture [85]. The lower layer should follow predefined rules regarding when to forward processing requests to higher layers, such as those based on their utilization or on the extent of a request [84]. This architecture

also accommodates occasional failures of single fog nodes because nodes from other layers can take over [79].

B. ARCHITECTURAL PATTERNS FOR SMART PROCESSING SUPPORT

1) DECENTRALIZED ORCHESTRATION

Use cases that have an increased variety of loads over time (C3) but rather similar jobs to fulfill (C2) should perform decentralized orchestration, where every node in the fog system is both a processor and an orchestrator because it allows for a better distribution of loads (e.g., [87], [88], [89], [90]). When all fog nodes are aware of their own utilization and the utilization of neighboring fog nodes, they can locally decide whether to process a job themselves or forward it to other fog nodes [87]. However, due to the complexity arising from connecting neighboring fog nodes and the administration of a decentralized system, this architecture is limited to small to medium systems (C7). The system benefits from avoiding queues while allowing for ultralow latency responses (C1) without significant orchestration overhead [88]. In addition, the system can flexibly manage the forwarding of more extensive requests to fog nodes with higher capabilities [89].

2) CENTRALIZED ORCHESTRATION

For use cases with varying loads over time (C3) and a broad range of jobs to fulfill (C2), centralized orchestration ensures a homogeneous distribution of jobs across the entire system and avoids queues for critical jobs (e.g., [50], [91], [92], [93]). For this purpose, one or several dedicated fog nodes act as orchestrator(s) and accept all incoming jobs for distribution to connected processing fog nodes [50]. The orchestration fog nodes usually only perform orchestration and no further processing of jobs. While this causes higher latency due to the intermediate step of accepting processing requests and routing them to available fog nodes, this system achieves high utilization and cost-effectiveness [91]. However, a single orchestrator node imposes resilience risks if it fails, so multiple nodes for orchestration should be considered in

implementation, [92]. Due to its topography, the centralized orchestration architecture can be easily scaled by adding further orchestration nodes and is therefore suitable for large systems (C7).

C. ARCHITECTURAL PATTERNS FOR THE EDGE MANAGER

1) PARALLEL INFRASTRUCTURES

For a use case with a public system of edge devices and fog nodes that flexibly join and leave (C4) and that has broad (C2) and varying (C3) loads, establishing parallel infrastructures for securing the system and for processing jobs reduces the risk of security-related incidents (e.g., [65], [66], [94]). Dedicated security fog nodes administer the entire system and supervise communication between processing nodes and edge devices. New devices and nodes then need to first register with the security nodes to access to the system [65]. As an additional security measure, the security nodes can check processing requests and shared data loads from edge devices before they are sent to the processing nodes [66]. Similarly, security nodes can also check all communications from processing nodes to edge devices. Based on the division into nodes for securing and nodes for processing, this architecture can also be scaled effectively and is therefore suitable for large systems (C7).

2) TEMPORARY CLUSTER

Use cases with a public system (C4) that host a range of edge devices with different capabilities (C6) should deploy temporarily adjacent fog nodes and edge devices that then form provisional fog clusters [13], [95], [96]. This means that all participating nodes and devices in a particular cluster can collaborate to distribute loads, jointly process jobs, and leverage their potentially underutilized capacity to address both broad (C2) and varying (C3) loads. Due to the flexible connection of unknown devices and nodes, such clusters must follow predefined security protocols and strictly adhere to particular security measures. Participants in the cluster profit from either low latency support to fulfill their demanding jobs (C1) or additional monetization of underutilized resources [95]. To establish such temporary fog clusters, the fog nodes and edge devices need to be equipped with compatible software [13], and the edge devices need to have a medium to high level of capabilities (C6). In addition, mechanisms for the compensation of fog nodes flexibly supporting edge devices must be prealigned and established, e.g., with smart contracts [95].

D. ARCHITECTURAL PATTERNS FOR THE HUB

A hub architecture is suitable for use cases with broad (C2) and varying (C3) loads in a public system (C4) that hosts edge devices (and fog nodes) that move significantly (C5). A hub splits the fog system into two layers: (1) a networking layer and (2) a processing layer. This splitting of layers allows the system to be administered effectively (e.g., [21], [68], [97], [98]), and both layers can be independently optimized based on evolving requirements and the system context [68].

The higher-layer networking fog nodes are responsible for tracking and predicting movements while monitoring available resources to coordinate handovers. This layer also takes over security-related jobs (e.g., authentication of edge devices that would like to join the system) to ensure that only trusted edge devices send jobs and that only trusted fog nodes process these jobs [31]. The networking fog nodes are interconnected for continuous exchange [97]. The networking nodes thereby ensure the efficient handover of jobs and reachability of mobile edge devices.

In the lower layer, fog nodes perform processing and, if needed, orchestration and security tasks [21]. The edge devices are connected to their closest processing fog nodes. The processing fog nodes are not connected to each other to maintain clear routes of communication [97]. While the networking nodes ensure the efficient handover of jobs and constant reachability of edge devices, the processing fog nodes allow for low-latency support for processing, orchestration, and security tasks [21].

VI. DISCUSSION

A. PRINCIPLE FINDINGS

The aim of this study was to develop SDPs to harness the clear opportunities of fog computing as an emerging technology while addressing the lack of best practices on designing fog systems that meet particular use case requirements. We revealed two driving forces for the design of fog systems. First, we provide seven categories and their specific manifestation continuums that can be applied to characterize a use case. Second, we define ten tasks that fog systems can fulfill and aggregate their commonly occurring permutations into six roles. We develop and richly describe seven architectural patterns for processing-related fog roles. We based our research results on a literature review with 105 selected articles as well as a workshop and several rounds of interviews with fog experts from industry and academia.

In general, the interviewed experts confirmed the challenge of designing and implementing fog systems that may need to operate with edge devices for several decades. They also stressed that high costs may arise if later system changes affect the physical edge device setup because devices are built and optimized in a cost-effective manner, requiring only minimum capabilities and allowing a long lifetime.

Reflecting our synthesis of proposed architectures for key fog roles, we were able to identify varying SDPs to address the unique use case manifestations in an effective way. Surprisingly, for the most extensive fog role, the hub, neither the literature nor the experts proposed an architecture different from a standardized split of fog nodes in processing and networking layers. All the sources we found that address this role define a hierarchical architectural pattern to harness the decentralization and geographical distribution of fog systems.

Notably, the experts disagreed with the literature regarding its suggestion to operate fog nodes purely for the management of a network or for taking over single security-related tasks, such as authenticating devices. The experts underlined the

core feature of fog computing as an extension of the edge to provide additional processing capabilities, for instance, to enable low-latency data analysis close to the edge.

Neither the literature nor the experts considered the characteristics of available cloud solutions when determining their fog use cases or specifying their architectural patterns. Instead, cloud services were regarded as globally available backup solutions for temporarily failing fog systems. Additionally, their extensive computation and storage capabilities could be harnessed if needed. This less prominent role of cloud services stems from the predominant focus on the edge–cloud continuum, which identifies measures to effectively and efficiently connect edge devices with cloud services while considering the limitations of (costly) edge devices (e.g., short battery lifetime, no internet access).

In general, our proposed architectures are rather focused on new implementations of fog systems. However, we also found that architectural patterns for less complex roles (such as processing assistance or smart processing support) can be extended with further nodes and layers to transform them into architectural patterns for more complex roles (such as edge manager or hub) with minor reorganization required. The other way would be rather challenging, as major parts of the architecture need to be cut. If a company needs to address multiple use cases with different characteristics that are to be combined in a single architecture, the hub or at least the edge manager architectural patterns can provide sufficient flexibility while still providing the benefits of an overarching architecture.

B. IMPLICATIONS FOR RESEARCH

This study provides three key contributions to related research. The available research on fog architectures provides an extensive collection of sophisticated theoretical and practically tested fog systems and their underlying architectures. However, researchers often follow a technology-centric approach to improve certain aspects of a fog architecture (e.g., [12], [21]) or an industry-specific approach to focus on a certain context (e.g., [16], [26]) when defining their architectures. To support a design approach that is both use-case-centric and industry-agnostic, we not only identify seven categories of use case characteristics but also determine potential manifestation continuums. We explain how these characteristics shape the selection of tasks and roles and impact architectural designs. We particularly clarify how latency criticality, job variety, load dynamics, system borders, and geographic mobility require the fog system to perform varying tasks and how edge device capability and system size impact architectural design decisions. Our proposed characteristics can therefore be used in research to categorize a fog use case, laying the foundation for decisions regarding the activities that fog nodes will perform in that use case as well as the selection of an effective fog architecture. Furthermore, our findings support bridging knowledge transfer in the implementation of fog systems across different industries, as these often share similar contexts and require similar fog system architectures.

Existing research on fog tasks takes a detailed technical perspective on specific tasks (e.g., [17], [32]) or defines high-level categories of tasks (e.g., [3], [33]) but does not discuss effective sets of tasks that a fog system can, cannot, or should perform depending on the use case. We also note that the description of fog system tasks differs greatly by industry, leading to overall low conceptual maturity [8]. We developed a task–role hierarchy for fog systems to address these research problems. This hierarchy comprises four key tasks and ten subtasks that fog nodes can perform, including processing, orchestration, securing, and networking. We synthesize and combine these tasks to reveal six roles that fog systems usually fulfill in different use cases independently of the industry. For each role, we explain under which use case conditions it is effective and develop and analyze a suitable architectural pattern. In this way, we support researchers in describing the application of the novel fog computing paradigm to address open research challenges or enhance business cases. We also seek to mitigate the conceptual ambiguity in extant research by providing descriptions of generic tasks and roles that fog systems can fulfill.

Finally, we found that many articles on fog architectures provide comprehensive suggestions for conceptualizing and designing (e.g., [40], [41], [42]) but rarely compare several architectural options. Extant research also neglects the broader context of a use case and does not consider the applicability of different architectural patterns for the use case. We developed seven architectural patterns, constituting a rich contribution to the literature. Our supplementary material provides comprehensive descriptions of each architectural pattern, including its context, examples, problems, forces, solutions, dynamics, variants, consequences, and known uses. We link use case characteristics and fog tasks and roles with our patterns and explain why an architectural pattern is effective in fulfilling the corresponding demands, enabling us to discuss the patterns' benefits and boundary conditions. By iteratively discussing and refining our literature-based results with fog computing experts, we provide empirical validation of our findings. Researchers and practitioners can leverage this study's collection of patterns as a foundation to build new architectures by selecting the most effective pattern for their use case.

C. IMPLICATIONS FOR PRACTICE

We support fog system stakeholders along the implementation of a new system with fog use case characteristics, tasks, roles, and architectural patterns. First, fog system architects can leverage our use case characteristics to assess their use cases in a structured way and reach agreement with business owners on the most critical requirements and characteristics. The use case characteristics thereby establish a common language for business and technical experts. Our use case archetypes (Appendix B) further support the categorization of a use case and can hint at existing fog implementations for similar use cases.

Second, system architects and designers can receive guidance on effective types of architectural patterns for their

case based on our linkage of use case characteristics and patterns. Drawing on a generic description of fog roles and tasks, they can also more efficiently discuss effective sets of combined tasks or options to extend or reduce the number of activities that the fog system will take on.

Third, developers can apply the detailed descriptions of architectural patterns together with their variants when implementing the system. By applying the canonical pattern structure proposed in prior research, our SDPs include a discussion of the consequences of patterns, which helps developers make better decisions for using SDPs and ultimately avoid common mistakes. Furthermore, they can more easily identify and refer to similar projects based on the examples and reference implementations provided in each pattern and use case archetype, even from other industries, to draw on a greater base of knowledge. Following the selection of an architectural pattern based on the initial use case characteristics and the tasks and roles, our research results also support consistent terminology across all entities included in the system design and implementation. Practitioners thereby avoid creating ambiguity during the course of implementing a new fog system. As a result, business owners, system architects, and developers can achieve greater fog system effectiveness by choosing an initial fog architectural pattern that fits their use case while fostering cross-industry knowledge evolution.

D. LIMITATIONS AND FUTURE WORK

Our study is subject to limitations, paving the way for future research. We built our research on 105 articles and interviews with eight experts to ensure a broad view of fog use cases and architectures, but we cannot guarantee the comprehensiveness of our work. Given fast technology lifecycles, further systems may have been developed recently that were not covered by our analysis. For instance, focusing on the term “fog” in our literature review ensured that the selected articles fit our scope. However, we also observed prevalent conceptual ambiguity related to fog computing. Literature and practice use varying terms to name architectures and technologies, such as far and near edge, mobile, or mist computing. We therefore might have missed relevant architectures. In addition, the analyzed literature focuses on the design and implementation of new fog systems in a greenfield environment, assuming that these systems can be mostly implemented independently from an existing architecture. Companies that plan to improve edge applications by introducing fog computing will instead need to incorporate a fog system into their existing infrastructure. Future research is thus needed to investigate how the architectural patterns we defined can be effectively applied in existing IT landscapes.

Our research is limited by the fact that the interviewed experts were mostly from a personal network of practitioners in central Europe (i.e., a convenience sample). Although they all have varying backgrounds and work for a diverse set of companies in different industries, further research may also focus on incorporating a more global perspective on fog computing. The literature coding was conducted mostly by

one author, posing the risk of subjective researcher bias. We addressed this limitation by conducting discussions and iterating our results among the team of authors, supported by the subsequent validation and refinement of the results through expert interviews.

VII. CONCLUSION

Fog computing can provide numerous benefits beyond the reduction of latency in edge device use cases. However, for an effective system comprising cloud, fog, and edge entities, the fog system architecture and related tasks must fit the use case characteristics. Otherwise, a high administrative overhead or a nonfunctioning system may erode the benefits of the fog system and necessitate expensive hardware updates. In this paper, we provide extensive recommendations on fog system architectures, which we hope to further explore in future studies.

APPENDIX A

Security manager. To fulfill the role of a security manager, a fog system takes over one or several advanced security-enhancing tasks to support edge devices. For instance, a fog system can constantly monitor and filter traffic to detect suspicious IP addresses and manage (DDoS) attacks [29] for many connected IoT devices. Alternatively, fog systems can manage the identities and authentication of edge devices [36] or support the preservation of privacy by encrypting data close to the edge. In this way, fog nodes take over energy-consuming security tasks for energy-restricted edge devices and pool resources to achieve higher cost efficiency. Fog systems can take over such advanced security tasks in use cases with high security requirements (e.g., healthcare patient data management) or a high risk of significant damage if attacks are recognized too late or are not mitigated (e.g., in IoT production settings), while additional data processing and orchestration capabilities are not required at the edge.

Network controller. The role of network controller requires a fog system to manage the mobility of edge devices that change networks over time or manage highly volatile network loads. The former scenario can include tracking the location of devices, adapting routing tables, and instructing other network devices to perform appropriate updates. Potential use cases for fog systems as pure network controllers include smart vehicle systems that require location-aware fog nodes close to vehicles to balance network traffic and avoid congestion [73]. In addition, local fog nodes can track and predict the movements of edge devices and prepare for the handover of processing support in the cloud if devices are close to changing networks [83]. For the role of network controller, fitting use cases require real-time coordination of moving devices and nodes without the need for further processing or resource orchestration capabilities close to the edge.

APPENDIX B

Practitioners and researchers should characterize their use cases by positioning the conditions and requirements between

the two poles of each category-specific continuum (Table IV). To illustrate and guide the determination of use case characteristics, we derived four commonly occurring archetypes of fog use cases: efficient, flexible, open, and mobile systems:

Efficient fog systems focus on keeping the latency as low as possible and consist of only low-capability edge devices [15], [99]. Edge devices in efficient systems are often simple sensors that usually submit only small data packages on a regular basis. Fog nodes consistently perform the same jobs and do not move geographically. These systems are often used for monitoring and reacting to threats (e.g., fire hazards) that require continuous tracking and immediate reactions to changes [79].

Flexible fog systems focus on handling different data loads originating from heterogeneous devices and their activities [52]. Such systems consist mostly of low-capability edge devices. Very low latency is still critical, but for times of peak loads, short waiting times for less critical jobs are acceptable [91]. Edge devices and fog nodes stay geographically close to each other and do not move within small to medium-sized systems. Flexible systems are often implemented, for instance, to represent and steer production lines and their logistics infrastructure, and they need to manage occasional spikes in processing requests [72].

Open fog systems focus on hosting environments that allow edge devices and fog nodes from different organizations to collaborate in a public system. Entities that join the system usually do not need specific permission but only need to adhere to the systems' preset rules [13]. Accordingly, the system handles less critical jobs and can therefore tolerate moderate latencies. Due to the openness of the system as well as its dynamic nature, both the heterogeneity of device capabilities and the variety of jobs and loads are high [100]. For instance, smart grids that allow devices to flexibly join and leave the system require advanced security measures close to the edge [73].

Mobile fog systems focus on the efficient management of geographically moving edge devices and fog nodes [68]. As they are usually in a public environment, such systems reach large sizes and incorporate a broad variety of heterogeneous devices with different jobs to submit [101]. Depending on the specific job, latency requirements can be restricted to ultralow or up to moderate reaction times. Such systems require specific architectures due to increased networking, routing, and handover challenges, for instance, in smart cities [102].

REFERENCES

- [1] S. N. Srirama, "A decade of research in fog computing: Relevance, challenges, and future directions," *Softw. Pract. Exp.*, vol. 54, no. 1, pp. 3–23, Jan. 2024, doi: 10.1002/spe.3243.
- [2] A. Maia *et al.*, "A survey on integrated computing, caching, and communication in the cloud-to-edge continuum," *Comput. Commun.*, vol. 219, pp. 128–152, Apr. 2024, doi: 10.1016/j.comcom.2024.03.005.
- [3] A. Yousefpour *et al.*, "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *J. Syst. Archit.*, vol. 98, pp. 289–330, Jan. 2019, doi: 10.1016/j.sysarc.2019.02.009.
- [4] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing - MCC '12*, M. Gerla and D. Huang, Eds., New York, NY, USA: ACM Press, 2012, p. 13, doi: 10.1145/2342509.2342513.
- [5] O. Nazih, N. Benamar, H. Lamaazi, and H. Chaoui, "Toward Secure and Trustworthy Vehicular Fog Computing: A Survey," *IEEE Access*, vol. 12, pp. 35154–35171, 2024, doi: 10.1109/ACCESS.2024.3371488.
- [6] A. Hazra, P. Rana, M. Adhikari, and T. Amgoth, "Fog computing for next-generation Internet of Things: Fundamental, state-of-the-art and research challenges," *Comput. Sci. Rev.*, vol. 48, p. 100549, May 2023, doi: 10.1016/j.cosrev.2023.100549.
- [7] S. Laato, M. Mäntymäki, A. K. M. N. Islam, S. Hyrynsalmi, and T. Birkstedt, "Trends and Trajectories in the Software Industry: implications for the future of work," *Inf. Syst. Front.*, Apr. 2022, doi: 10.1007/s10796-022-10267-4.
- [8] P. Habibi, M. Farhoudi, S. Kazemian, S. Khorsandi, and A. Leon-Garcia, "Fog Computing: A Comprehensive Architectural Survey," *IEEE Access*, vol. 8, pp. 69105–69133, Jan. 2020, doi: 10.1109/ACCESS.2020.2983253.
- [9] R. Das and M. M. Inuwa, "A review on fog computing: Issues, characteristics, challenges, and potential applications," *Telemat. Inform. Rep.*, vol. 10, p. 100049, Jun. 2023, doi: 10.1016/j.teler.2023.100049.
- [10] A. Mahta, W. Tärneberg, C. Klein, J. Tordsson, M. Kihl, and E. Elmroth, "How Beneficial Are Intermediate Layer Data Centers in Mobile Edge Networks?," presented at the IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS*W), Augsburg, Germany, 2016.
- [11] M. Blume, S. Lins, and A. Sunyaev, "Uncovering Effective Roles and Tasks for Fog Systems," in *Service-Oriented and Cloud Computing*, vol. 14183, G. A. Papadopoulos, F. Rademacher, and J. Soldani, Eds., in Lecture Notes in Computer Science, vol. 14183, Cham: Springer Nature Switzerland, 2023, pp. 119–135, doi: 10.1007/978-3-031-46235-1_8.
- [12] M. Veeramani and S. Sankaranarayanan, "Publish/subscribe broker based architecture for fog computing," in *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, 2017, pp. 1024–1026, doi: 10.1109/ICECDS.2017.8389592.
- [13] H. Du, S. Leng, F. Wu, X. Chen, and S. Mao, "A New Vehicular Fog Computing Architecture for Cooperative Sensing of Autonomous Driving," *IEEE Access*, vol. 8, pp. 10997–11006, Jan. 2020, doi: 10.1109/ACCESS.2020.2964029.
- [14] I. Ahammad, "Fog Computing Complete Review: Concepts, Trends, Architectures, Technologies, Simulators, Security Issues, Applications, and Open Research Fields," *SN Comput. Sci.*, vol. 4, no. 6, p. 765, Oct. 2023, doi: 10.1007/s42979-023-02235-9.
- [15] P. Maiti, H. K. Apat, A. Kumar, B. Sahoo, and A. K. Turuk, "Deployment of Multi-tier Fog Computing System for IoT Services in Smart City," in *2019 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, 2019, pp. 1–6, doi: 10.1109/ANTS47819.2019.9117921.
- [16] R. K. Barik, H. Dubey, and K. Mankodiya, "SOA-FOG: Secure service-oriented edge computing architecture for smart health big data analytics," in *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2017, pp. 477–481, doi: 10.1109/GlobalSIP.2017.8308688.
- [17] R. Mahmud, K. Ramamohanarao, and R. Buyya, "Application Management in Fog Computing Environments: A Taxonomy, Review and Future Directions," *ACM Comput. Surv.*, vol. 53, no. 4, Jan. 2020, doi: 10.1145/3403955.
- [18] S. Chatterjee, S. Sarker, M. J. Lee, X. Xiao, and A. Elbanna, "A possible conceptualization of the information systems (IS) artifact: A general systems theory perspective," *Inf. Syst. J.*, vol. 31, no. 4, pp. 550–578, Jan. 2021, doi: 10.1111/isj.12320.
- [19] M. Iorga, L. Feldman, R. Barton, M. J. Martin, N. Goren, and C. Mahmoudi, "Fog computing conceptual model," National Institute of Standards and Technology, Gaithersburg, MD, Jan. 2018, doi: 10.6028/NIST.SP.500-325.
- [20] L. M. Vaquero and L. Roderio-Merino, "Finding your Way in the Fog," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 27–32, Jan. 2014, doi: 10.1145/2677046.2677052.
- [21] B. Cao, Z. Sun, J. Zhang, and Y. Gu, "Resource Allocation in 5G IoV

- Architecture Based on SDN and Fog-Cloud Computing,” *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3832–3840, Jan. 2021, doi: 10.1109/TITS.2020.3048844.
- [22] M. Blume, S. Lins, and A. Sunvaev, “Understanding Interdependencies among Fog System Characteristics,” in *2022 IEEE 24th Conference on Business Informatics (CBI)*, Amsterdam, Netherlands: IEEE, Jun. 2022, pp. 126–135. doi: 10.1109/CBI54897.2022.00021.
- [23] H. Qusa and J. Tarazi, “Collaborative Fog Computing Architecture for Privacy-Preserving Data Aggregation,” in *2021 IEEE World AI IoT Congress (AlloT)*, 2021, pp. 0086–0091. doi: 10.1109/AI-IoT52608.2021.9454198.
- [24] M. Younas and I. Awan, “Cloud, IoT and Data Science,” *Inf. Syst. Front.*, vol. 26, no. 4, pp. 1219–1222, Aug. 2024, doi: 10.1007/s10796-024-10521-x.
- [25] C. Avasalcari, I. Murturi, and S. Dustdar, “Edge and Fog: A Survey, Use Cases, and Future Challenges,” in *Fog Computing*, 1st ed., A. Zomaya, A. Abbas, and S. Khan, Eds., Wiley, 2020, pp. 43–65. doi: 10.1002/9781119551713.ch2.
- [26] P. Fraga-Lamas et al., “Design and Empirical Validation of a Bluetooth 5 Fog Computing Based Industrial CPS Architecture for Intelligent Industry 4.0 Shipyard Workshops,” *IEEE Access*, vol. 8, pp. 45496–45511, Jan. 2020, doi: 10.1109/ACCESS.2020.2978291.
- [27] Z. Constantinescu and M. Vladoiu, “Towards Vehicular Fog Computing: an Architecture for Connected Vehicles and Vehicular Clouds,” in *2020 19th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, 2020, pp. 1–6. doi: 10.1109/RoEduNet51892.2020.9324868.
- [28] R. Mahmud, R. Kotagiri, and R. Buyya, “Fog Computing: A Taxonomy, Survey and Future Directions,” in *Internet of Everything*, B. Di Martino, K.-C. Li, L. T. Yang, and A. Esposito, Eds., in *Internet of Things*, Singapore: Springer Singapore, 2018, pp. 103–130. doi: 10.1007/978-981-10-5861-5_5.
- [29] W. Razouk, D. Sgandurra, and K. Sakurai, “A New Security Middleware Architecture Based on Fog Computing and Cloud to Support IoT Constrained Devices,” in *Proceedings of the 1st International Conference on Internet of Things and Machine Learning*, in IML ’17, New York, NY, USA: Association for Computing Machinery, 2017. doi: 10.1145/3109761.3158413.
- [30] K. Ma, A. Bagula, C. Nyirenda, and O. Ajayi, “An IoT-Based Fog Computing Model,” *Sens. 14248220*, vol. 19, no. 12, p. 2783, Jan. 2019, doi: 10.3390/s19122783.
- [31] W. Kim and S. Chung, “User-Participatory Fog Computing Architecture and Its Management Schemes for Improving Feasibility,” *IEEE Access*, vol. 6, pp. 20262–20278, Jan. 2018, doi: 10.1109/ACCESS.2018.2815629.
- [32] A. Ahmed et al., “Fog Computing Applications: Taxonomy and Requirements,” Jan. 2019. [Online]. Available: <http://arxiv.org/pdf/1907.11621v1>
- [33] J. Singh, P. Singh, and S. S. Gill, “Fog computing: A taxonomy, systematic review, current trends and research challenges,” *J. Parallel Distrib. Comput.*, vol. 157, pp. 56–85, Jan. 2021, doi: 10.1016/j.jpdc.2021.06.005.
- [34] R. K. Naha et al., “Fog Computing: Survey of Trends, Architectures, Requirements, and Research Directions,” *IEEE Access*, vol. 6, pp. 47980–48009, Jan. 2018, doi: 10.1109/ACCESS.2018.2866491.
- [35] S. B. Nath, H. Gupta, S. Chakraborty, and S. K. Ghosh, “A Survey of Fog Computing and Communication: Current Researches and Future Directions,” *arXiv*, Jan. 2018. doi: 10.48550/arXiv.1804.04365.
- [36] C. Mouradian, D. Naboulsi, S. Yangu, R. H. Gliho, M. J. Morrow, and P. A. Polakos, “A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges,” *IEEE Commun. Surv. Tutor.*, vol. 20, no. 1, pp. 416–464, Jan. 2018, doi: 10.1109/COMST.2017.2771153.
- [37] E. Gamma, R. Helm, R. Johnson, and J. M. Vlissides, *Entwurfsmuster: Elemente wiederverwendbarer objektorientierter Software*, 6. Aufl. in *Programmer's choice*. München: Addison-Wesley, 2011.
- [38] F. Buschmann, *Pattern-oriented software architecture: a system of patterns*. Chichester: J. Wiley, 1996.
- [39] A. Seitz, F. Thiele, and B. Bruegge, “Fogxy: An Architectural Pattern for Fog Computing,” in *Proceedings of the 23rd European Conference on Pattern Languages of Programs*, in EuroPLoP ’18, New York, NY, USA: Association for Computing Machinery, 2018. doi: 10.1145/3282308.3282342.
- [40] M. H. Syed, E. B. Fernandez, and M. Ilyas, “A Pattern for Fog Computing,” in *Proceedings of the 10th Travelling Conference on Pattern Languages of Programs*, in VikingPLoP ’16, New York, NY, USA: Association for Computing Machinery, 2016. doi: 10.1145/3022636.3022649.
- [41] Y. Liu, J. E. Fieldsend, and G. Min, “A Framework of Fog Computing: Architecture, Challenges, and Optimization,” *IEEE Access*, vol. 5, pp. 25445–25454, Jan. 2017, doi: 10.1109/ACCESS.2017.2766923.
- [42] I. T. Popović and A. Ž. Rakić, “The Fog-Based Framework for Design of Real-Time Control Systems in Internet of Things Environment,” in *2018 International Symposium on Industrial Electronics (INDEL)*, 2018, pp. 1–6. doi: 10.1109/INDEL.2018.8637639.
- [43] G. Paré, M.-C. Trudel, M. Jaana, and S. Kitsiou, “Synthesizing information systems knowledge: A typology of literature reviews,” *Inf. Manage.*, vol. 52, no. 2, pp. 183–199, Mar. 2015, doi: 10.1016/j.im.2014.08.008.
- [44] M. D. Myers, *Qualitative research in business & management*, Repr. Los Angeles: Sage, 2011.
- [45] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, “Systematic literature reviews in software engineering – A systematic literature review,” *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, Jan. 2009, doi: 10.1016/j.infsof.2008.09.009.
- [46] J. Vom Brocke, A. Simons, K. Riemer, B. Niehaves, R. Plattfaut, and A. Cleven, “Standing on the Shoulders of Giants: Challenges and Recommendations of Literature Search in Information Systems Research,” *Commun. Assoc. Inf. Syst.*, vol. 37, 2015, doi: 10.17705/1CAIS.03709.
- [47] D. Roy, S. Dutta, A. Datta, and G. Das, “A Cost Effective Architecture and Throughput Efficient Dynamic Bandwidth Allocation Protocol for Fog Computing Over EPON,” *IEEE Trans. Green Commun. Netw.*, vol. 4, no. 4, pp. 998–1009, Jan. 2020, doi: 10.1109/TGCN.2020.2994409.
- [48] M. C. Lacity, S. A. Khan, and L. P. Willcocks, “A review of the IT outsourcing literature: Insights for practice,” *J. Strateg. Inf. Syst.*, vol. 18, no. 3, pp. 130–146, Jan. 2009, doi: 10.1016/j.jsis.2009.06.002.
- [49] B. Khazael, H. T. Malazi, and S. Clarke, “Complex Event Processing in Smart City Monitoring Applications,” *IEEE Access*, vol. 9, pp. 143150–143165, Jan. 2021, doi: 10.1109/ACCESS.2021.3119975.
- [50] S. Taherizadeh, V. Stankovski, and M. Grobelnik, “A Capillary Computing Architecture for Dynamic Internet of Things: Orchestration of Microservices from Edge Devices to Fog and Cloud Providers,” *Sensors*, vol. 18, no. 9, p. 2938, Jan. 2018, doi: 10.3390/s18092938.
- [51] G. Ortiz, M. Zouai, O. Kazar, A. Garcia-de-Prado, and J. Boubeta-Puig, “Atmosphere: Context and situational-aware collaborative IoT architecture for edge-fog-cloud computing,” *Comput. Stand. Interfaces*, vol. 79, p. 103550, Jan. 2022, doi: 10.1016/j.csi.2021.103550.
- [52] Syed Rizwan Hassan, I. Ahmad, J. Nebhen, Ateeq Ur Rehman, M. Shafiq, and J.-G. Choi, “Design of Latency-Aware IoT Modules in Heterogeneous Fog-Cloud Computing Networks,” *Comput. Mater. Contin.*, vol. 70, no. 3, pp. 6057–6072, Jan. 2022, doi: 10.32604/cmc.2022.020428.
- [53] J. O. Borchers, “A pattern approach to interaction design,” in *Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques*, New York City New York USA: ACM, Aug. 2000, pp. 369–378. doi: 10.1145/347642.347795.
- [54] B. Appleton, “Patterns and software: essential concepts and terminology,” *Object Mag. Online*, vol. 3, no. 5, pp. 20–25, 1997.
- [55] R. K. Yin, *Case study research: design and methods*, 4. ed., [Nashchdr.]. in *Applied social research methods series*, no. 5. Los Angeles, Calif.: Sage, 20.
- [56] N. Kannengieser, S. Lins, C. Sander, K. Winter, H. Frey, and A. Sunyaev, “Challenges and Common Solutions in Smart Contract Development,” *IEEE Trans. Softw. Eng.*, vol. 48, no. 11, pp. 4291–4318, Nov. 2022, doi: 10.1109/TSE.2021.3116808.
- [57] S. Petter, D. Khazanchi, and J. D. Murphy, “A design science based evaluation framework for patterns,” *ACM SIGMIS Database DATA-BASE Adv. Inf. Syst.*, vol. 41, no. 3, pp. 9–26, Aug. 2010, doi: 10.1145/1851175.1851177.
- [58] C. Rolland, J. Stima, N. Prekas, P. Loucopoulos, A. Persson, and G.

- Grosz, "Evaluating a Pattern Approach as an Aid for the Development of Organisational Knowledge: An Empirical Study," in *Active Flow and Combustion Control 2018*, vol. 141, R. King, Ed., in Notes on Numerical Fluid Mechanics and Multidisciplinary Design, vol. 141, Cham: Springer International Publishing, 2000, pp. 176–191. doi: 10.1007/3-540-45140-4_13.
- [59] D. Lea, "Christopher Alexander: an introduction for object-oriented designers," *ACM SIGSOFT Softw. Eng. Notes*, vol. 19, no. 1, pp. 39–46, Jan. 1994, doi: 10.1145/181610.181617.
- [60] S. Niebuhr, K. Kohler, and C. Graf, "Engaging Patterns: Challenges and Means Shown by an Example," in *Engineering Interactive Systems*, vol. 4940, J. Gulliksen, M. B. Harning, P. Palanque, G. C. Van Der Veer, and J. Wesson, Eds., in Lecture Notes in Computer Science, vol. 4940, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 586–600. doi: 10.1007/978-3-540-92698-6_35.
- [61] P. Maiti, K. S. Sahoo, B. Sahoo, and A. K. Turuk, "Smart Gateway Based Multi-Tier Service-Oriented Fog Computing Architecture for Achieving Ultra-Low Latency," in *2018 International Conference on Information Technology (ICIT)*, Bhubaneswar, India: IEEE, Dec. 2018, pp. 278–283. doi: 10.1109/ICIT.2018.00063.
- [62] R. Andreoli *et al.*, "A Multi-Domain Survey on Time-Criticality in Cloud Computing," *IEEE Trans. Serv. Comput.*, vol. 18, no. 2, pp. 1152–1170, Mar. 2025, doi: 10.1109/TSC.2025.3539197.
- [63] E. A. Alshuaibi, A. M. Hamdi, and F. K. Hussain, "Volunteer Computing for fog scalability: A systematic literature review," *Internet Things*, vol. 25, p. 101072, Apr. 2024, doi: 10.1016/j.iot.2024.101072.
- [64] H. A. Alharbi, T. E. H. Elgorashi, and J. M. H. Elmurghani, "Energy Efficient Virtual Machines Placement Over Cloud-Fog Network Architecture," *IEEE Access*, vol. 8, pp. 94697–94718, 2020, doi: 10.1109/ACCESS.2020.2995393.
- [65] C. Núñez-Gómez, B. Caminero, and C. Carrión, "HIDRA: A Distributed Blockchain-Based Architecture for Fog/Edge Computing Environments," *IEEE Access*, vol. 9, pp. 75231–75251, Jan. 2021, doi: 10.1109/ACCESS.2021.3082197.
- [66] D. D. Sánchez-Gallegos *et al.*, "On the Continuous Processing of Health Data in Edge-Fog-Cloud Computing by Using Micro/Nanoservice Composition," *IEEE Access*, vol. 8, pp. 120255–120281, Jan. 2020, doi: 10.1109/ACCESS.2020.3006037.
- [67] A. A. Khan, M. Abolhasan, and W. Ni, "5G next generation VANETs using SDN and fog computing framework," in *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, 2018, pp. 1–6. doi: 10.1109/CCNC.2018.8319192.
- [68] A. Souza and S. Tohme, "Multi-level SDN with vehicles as fog computing infrastructures: A new integrated architecture for 5G-VANETs," in *2018 21st Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, Paris: IEEE, Feb. 2018, pp. 1–8. doi: 10.1109/ICIN.2018.8401604.
- [69] K. Ostrowski, K. Małeck, P. Dziurzański, and A. K. Singh, "Mobility-aware fog computing in dynamic networks with mobile nodes: A survey," *J. New. Comput. Appl.*, vol. 219, p. 103724, Oct. 2023, doi: 10.1016/j.jnca.2023.103724.
- [70] B. Tang, Z. Chen, G. Heffernan, T. Wei, H. He, and Q. Yang, "A Hierarchical Distributed Fog Computing Architecture for Big Data Analysis in Smart Cities," in *Proceedings of the ASE BigData & Social Informatics 2015*, in ASE BD&SI '15, New York, NY, USA: Association for Computing Machinery, 2015. doi: 10.1145/2818869.2818898.
- [71] R. Pitchai, K. V. Guru, J. N. Gandhi, C. R. Komala, J. R. D. Kumar, and S. Boopathi, "Fog Computing-Integrated ML-Based Framework and Solutions for Intelligent Systems: Digital Healthcare Applications," in *Advances in Systems Analysis, Software Engineering, and High Performance Computing*, S. Sharma, A. Prakash, and V. Sugumaran, Eds., IGI Global, 2024, pp. 196–224. doi: 10.4018/979-8-3693-0968-1.ch008.
- [72] I. Ungurean and N. C. Gaitan, "Software Architecture of a Fog Computing Node for Industrial Internet of Things," *Sensors*, vol. 21, no. 11, p. 3715, Jan. 2021, doi: 10.3390/s21113715.
- [73] H. Wang, Q. Wang, Y. Li, G. Chen, and Y. Tang, "Application of Fog Architecture Based on Multi-agent Mechanism in CPPS," in *2018 2nd IEEE Conference on Energy Internet and Energy System Integration (EI2)*, 2018, pp. 1–6. doi: 10.1109/EI2.2018.8582467.
- [74] M. Aldossary, "Multi-Layer Fog-Cloud Architecture for Optimizing the Placement of IoT Applications in Smart Cities," *Comput. Mater. Contin.*, vol. 75, no. 1, pp. 633–649, 2023, doi: 10.32604/cmc.2023.035414.
- [75] H. Zhang, B. Qin, T. Tu, Z. Guo, F. Gao, and Q. Wen, "An Adaptive Encryption-as-a-Service Architecture Based on Fog Computing for Real-Time Substation Communications," *IEEE Trans. Ind. Inform.*, vol. 16, no. 1, pp. 658–668, Jan. 2020, doi: 10.1109/TII.2019.2948113.
- [76] A. H. Mayer, V. F. Rodrigues, C. A. de Costa, R. d. R. Righi, A. Roehrs, and R. S. Antunes, "FogChain: A Fog Computing Architecture Integrating Blockchain and Internet of Things for Personal Health Records," *IEEE Access*, vol. 9, pp. 122723–122737, Jan. 2021, doi: 10.1109/ACCESS.2021.3109822.
- [77] A. Gupta and S. K. Gupta, "Unmanned aerial vehicles in collaboration with fog computing network for improving quality of service," *Int. J. Commun. Syst.*, vol. 37, no. 9, p. e5759, Jun. 2024, doi: 10.1002/dac.5759.
- [78] T. Mekki, I. Jabri, A. Rachedi, and M. Ben Jemaa, "Towards Multi-Access Edge Based Vehicular Fog Computing Architecture," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6. doi: 10.1109/GLOCOM.2018.8647850.
- [79] A. Seitz, J. O. Johanssen, B. Bruegge, V. Lofness, V. Hartkopf, and M. Sturm, "A Fog Architecture for Decentralized Decision Making in Smart Buildings," in *Proceedings of the 2nd International Workshop on Science of Smart City Operations and Platforms Engineering*, in SCOPE '17, New York, NY, USA: Association for Computing Machinery, 2017, pp. 34–39. doi: 10.1145/3063386.3063768.
- [80] Q. Qi and F. Tao, "A Smart Manufacturing Service System Based on Edge Computing, Fog Computing, and Cloud Computing," *IEEE Access*, vol. 7, pp. 86769–86777, 2019, doi: 10.1109/ACCESS.2019.2923610.
- [81] H. Deng, Z. Guo, R. Lin, and H. Zou, "Fog Computing Architecture-Based Data Reduction Scheme for WSN," in *2019 1st International Conference on Industrial Artificial Intelligence (IAI)*, 2019, pp. 1–6. doi: 10.1109/ICIAI.2019.8850817.
- [82] M. H. Yaghmaee Moghaddam and A. Leon-Garcia, "A fog-based internet of energy architecture for transactive energy management systems," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1055–1069, Jan. 2018, doi: 10.1109/JIOT.2018.2805899.
- [83] H. Alzahrani, T. Sheltami, A. Barnawi, M. Imam, and A. Yaser, "A Lightweight Intrusion Detection System Using Convolutional Neural Network and Long Short-Term Memory in Fog Computing," *Comput. Mater. Contin.*, vol. 80, no. 3, pp. 4703–4728, 2024, doi: 10.32604/cmc.2024.054203.
- [84] V. Rampérez, J. Soriano, D. Lizcano, and Raquel Lacuesta, "A Multidomain Standards-Based Fog Computing Architecture for Smart Cities," *Wirel. Commun. Mob. Comput. Online*, vol. 2018, Jan. 2018, doi: 10.1155/2018/4019858.
- [85] A. Souza, L. Izidio, A. Rocha, N. Cacho, and T. Batista, "Sapparchi: An Architecture for Smart City Applications from Edge, Fog and Cloud Computing," in *2019 IEEE International Smart Cities Conference (ISC2)*, 2019, pp. 262–267. doi: 10.1109/ISC246665.2019.9071686.
- [86] H. Martinez, F. J. Rodriguez-Lozano, F. León-García, J. M. Palomares, and J. Olivares, "Distributed Fog computing system for weapon detection and face recognition," *J. Netw. Comput. Appl.*, vol. 232, p. 104026, Dec. 2024, doi: 10.1016/j.jnca.2024.104026.
- [87] S. S. Xu, C. Chen, and T. Chang, "Design of oneM2M-Based Fog Computing Architecture," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 9464–9474, Jan. 2019, doi: 10.1109/JIOT.2019.2929118.
- [88] W. Lee, K. Nam, H. Roh, and S. Kim, "A gateway based fog computing architecture for wireless sensors and actuator networks," in *2016 18th International Conference on Advanced Communication Technology (ICACT)*, 2016, pp. 210–213. doi: 10.1109/ICACT.2016.7423332.
- [89] J. Santos, J. v. d. Hooft, M. T. Vega, T. Wauters, B. Volckaert, and F. D. Turck, "SRFog: A flexible architecture for Virtual Reality content delivery through Fog Computing and Segment Routing," in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2021, pp. 1038–1043.
- [90] A. Almas, W. Iqbal, A. Altaf, K. Saleem, S. Mussiraliyeva, and M.

- W. Iqbal, "Context-Based Adaptive Fog Computing Trust Solution for Time-Critical Smart Healthcare Systems," *IEEE Internet Things J.*, vol. 10, no. 12, pp. 10575–10586, Jun. 2023, doi: 10.1109/IJOT.2023.3242126.
- [91] N. Kaur and A. Mittal, "Fog Computing Serverless Architecture for Real Time Unpredictable Traffic," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1022, no. 1, p. 012026, Jan. 2021, doi: 10.1088/1757-899X/1022/1/012026.
- [92] X. Wei and L. Wu, "A New Proposed Sensor Cloud Architecture Based on Fog Computing for Internet of Things," 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)*, 2019, pp. 615–620. doi: 10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00120.
- [93] S. Iftikhar *et al.*, "HunterPlus: AI based energy-efficient task scheduling for cloud–fog computing environments," *Internet Things*, vol. 21, p. 100667, Apr. 2023, doi: 10.1016/j.iot.2022.100667.
- [94] Z. Zhou, Y. Tian, J. Xiong, C. Peng, J. Li, and N. Yang, "Blockchain and signcrypton enabled asynchronous federated learning framework in fog computing," *Digit. Commun. Netw.*, vol. 11, no. 2, pp. 442–454, Apr. 2025, doi: 10.1016/j.dcan.2024.03.004.
- [95] X. Huang, D. Ye, R. Yu, and L. Shu, "Securing parked vehicle assisted fog computing with blockchain and optimal smart contract design," *IEEECAA J. Autom. Sin.*, vol. 7, no. 2, pp. 426–441, Mar. 2020, doi: 10.1109/JAS.2020.1003039.
- [96] Y. Hou, Z. Wei, R. Zhang, X. Cheng, and L. Yang, "Hierarchical Task Offloading for Vehicular Fog Computing Based on Multi-Agent Deep Reinforcement Learning," *IEEE Trans. Wirel. Commun.*, vol. 23, no. 4, pp. 3074–3085, Apr. 2024, doi: 10.1109/TWC.2023.3305321.
- [97] Uikyun Na and L. Eun-Kyu, "Fog BEMS: An Agent-Based Hierarchical Fog Layer Architecture for Improving Scalability in a Building Energy Management System," *Sustainability*, vol. 12, no. 7, p. 2831, Jan. 2020, doi: 10.3390/su12072831.
- [98] A. U. R. Butt *et al.*, "Proactive and data-centric Internet of Things-based fog computing architecture for effective policing in smart cities," *Comput. Electr. Eng.*, vol. 123, p. 110030, Apr. 2025, doi: 10.1016/j.compeleceng.2024.110030.
- [99] I. Zyrianoff *et al.*, "Architecting and Deploying IoT Smart Applications: A Performance-Oriented Approach," *Sensors*, vol. 20, no. 1, p. 84, Jan. 2020, doi: 10.3390/s20010084.
- [100] F. La Vega, J. Soriano, M. Jimenez, D. Lizcano, and Raquel Lacuesta, "A Peer-to-Peer Architecture for Distributed Data Monetization in Fog Computing Scenarios," *Wirel. Commun. Mob. Comput. Online*, vol. 2018, p. 15, Jan. 2018, doi: 10.1155/2018/5758741.
- [101] H. Le, N. Achir, and K. Boussetta, "Fog computing architecture with heterogeneous Internet of Things technologies," in *2019 10th International Conference on Networks of the Future (NoF)*, 2019, pp. 130–133. doi: 10.1109/NoF47743.2019.9014960.
- [102] E. Baccarelli, P. G. V. Naranjo, M. Scarpiniti, M. Shojafar, and J. H. Abawajy, "Fog of Everything: Energy-Efficient Networked Computing Architectures, Research Challenges, and a Case Study," *IEEE Access*, vol. 5, pp. 9882–9910, Jan. 2017, doi: 10.1109/ACCESS.2017.2702013.



MAXIMILIAN BLUME is a PhD student at the Karlsruhe Institute of Technology (KIT). In parallel to working on his PhD, he gathered work experience at McKinsey Digital, Tozero, and Orcan Energy. His research addresses challenges on the IT infrastructure between edge devices and the cloud with a focus on fog computing. His research has been published in proceedings of conferences such as the IEEE Conference on Business Informatics or the European Conference on

Service-Oriented and Cloud Computing.



SEBASTIAN LINS is a full professor for information systems at the University of Kassel. Before joining the University of Kassel, he was postdoctoral researcher at the Institute of Applied Informatics and Formal Description Methods, Karlsruhe Institute of Technology (KIT). He works on research challenges concerned with the design, use, and assessment of secure and trustworthy information systems. His work has been published in international journals such as *Information Systems Research*, *Journal of the Association for Information Systems*, *Electronic Markets*, *ACM SIGMIS Database*, and *ACM Computing Surveys*.



ALI SUNYAEV is a full professor for computer science at the Technical University of Munich (TUM). Before joining the TUM, he was a professor at the Karlsruhe Institute of Technology, the University of Kassel, and the University of Cologne. His research work accounts for the multifaceted use contexts of digital technologies with research on human behavior affecting IT applications and vice versa. His research appeared in journals, including *Information Systems Research*, *Journal of Management Information Systems*, *Journal of Information Technology*, *Journal of the Association for Information Systems*, *IEEE Transactions on Cloud Computing*, *Communications of the ACM*, and others.