Towards a Unified Local-First Computing Environment for Multi-Device Development

Dr. Swati R. Khokale[1], Bhavik D. Pawar[2], Shivam B. More[2], Tasbiya H. Shaikh[2], Ishika S. Sharma[2]

[1]Assistant Professor, Department of Computer Engineering, Guru Gobind Singh College of Engineering and Research Centre, Nashik, India

[2]Student, Department of Computer Engineering, Guru Gobind Singh College of Engineering and Research Centre, Nashik, India

11 November 2026

Abstract

Today's software development increasingly spans mul-tiple personal devices, yet these environments often remain isolated, without cross-device visibility or con-solidated resource awareness. Traditional cloud-centric paradigms move application state and execution to re-mote data centers, adding network latency, requiring persistent internet connectivity, and limiting user own-ership and control. Service discovery and task coordi-nation often rely on central servers, creating a single point of failure and limiting operation during network disconnection.

Existing research addresses local-first computing, de-vice and service discovery, multi-device interaction, dis-tributed resource management, distributed task execu-tion, and metadata and resource discovery. However, these capabilities are generally studied separately and in different deployment contexts. Local-first research often does not extend to system-level resources such as con-tainers, databases, active services, and Git repositories, while distributed resource management and FaaS edge frameworks generally target enterprise environments. Existing solutions therefore leave a gap for a unified, offline-aware, local-first layer that integrates metadata, services, resources, and task execution.

This paper presents DaemonONE, a proposed local-first distributed computing environment that provides a unified logical view across an individual's personal devices. The review establishes an architectural and conceptual foundation for investigating this integrated personal computing environment.

Keywords— ***Local-first computing, multi-device systems, resource discovery, metadata manage-ment, peer-to-peer computing, distributed com-puting, controlled task execution.***

I Introduction

Software developer is now increasingly working with different personal devices such as laptops, desktops, home servers, and laboratory servers to perform tasks like testing, database hosting, compilation, and ser-vice management. Different devices are independent of each other, thus causing the distribution of reposi-tories, databases, containers, and computing resources across different devices. The cloud-based approach has the issue of network latency and requirement for an uninterrupted internet connection, with less control and ownership of the data [2, 21]. The disconnection of the network can cause further issues in the discovery and management of the devices [6, 7].

Local-first computing revolves around the concepts of data ownership, data locality, and limited reliance on central authorities [2, 3]. Other work that has been done in the past includes decentralized device/service discovery [6, 7], multi-device interaction [17, 18], and distributed/edge resource management [14, 15, 23]. Yet again, these aspects have not been explored together but separately, providing an opportunity to bring them all together into one environment.

In this paper we present **DaemonONE**, a distributed computational system which gives a single view about the collection of the user's devices. The system incor-porates decentralized discovery of devices, indexing of device metadata, offline-aware resource management, service monitoring, searching across devices and con-trolled execution of predefined tasks for developers.

The rest of the paper is structured as follows. Sec-tion II provides the literature review, which gives a summary of the selected research papers. Section III de-scribes the research gap, proposed system, methodology, and architecture of DaemonONE. Section IV analyzes the findings and compares the proposed system with existing distributed computing architectures. Section V concludes the paper and highlights future work.

II Literature Survey

Designing computing platforms for personal multi-device software development draws on several areas of parallel and distributed systems. The following sub-sections review six themes: local-first software and data consistency, device and service discovery, multi-device interaction and workflow migration, distributed resource management, distributed task execution, and metadata

indexing. A comparative synthesis closes the survey and identifies the research gaps that motivate DaemonONE.

II.1 Data Overlay Mesh in Distributed Clouds Enabling Collaborative Applications (2025)

M. Simić et al. [1] present a logical overlay mesh that connects distributed cloud resources to enable collabo-rative applications. This technique offers a common ab-straction for accessing distributed computing resources.

II.2 Local-First Software: You Own Your Data, in Spite of the Cloud (2019)

M. Kleppmann et al. [2] describe local-first software, emphasizing data ownership, locality, collaboration, and independence from central cloud services. They ana-lyze design principles for responsive applications that preserve user control over data.

II.3 LoRe: A Programming Model for Verifiably Safe Local-First Software (2023)

J. Haas et al. [3] present LoRe, a programming model that applies formal methods to specify and verify safety properties in local-first software.

II.4 Distributed Computing with the Cloud (2024)

Y. Afek et al. [4] analyze distributed computing models that integrate local communication with cloud storage and computing, focusing on coordination of computing and communication resources.

II.5 Optimization of Resource-Aware Parallel and Distributed Computing: A Review (2025)

P. Czarnul et al. [5] review optimization approaches for resource-aware parallel and distributed computing, including resource modeling, scheduling, workload bal-ancing, and optimization.

II.6 Multicast DNS (2013)

S. Cheshire and M. Krochmal [6] describe Multicast DNS (mDNS), a protocol for resolving device names on a local network without a DNS server. It provides zero-configuration name resolution using multicast com-munication.

II.7 DNS Service Discovery (2013)

S. Cheshire and M. Krochmal [7] describe DNS-based Service Discovery, which uses DNS records to advertise service types and connection details.

II.8 A Distributed Tracing Pipeline for Improving Locality Awareness of Microservices Applications (2024)

C. Colarusso et al. [8] present a distributed tracing pipeline that analyzes communication in microservices applications. They use tracing data to identify service relationships and improve locality awareness during placement.

II.9 Conflict-Free Replicated Data Types: An Overview (2018)

N. Preguiça [9] surveys conflict-free replicated data types (CRDTs), which allow replicas to converge after updates are exchanged under their defined assumptions.

II.10 WireGuard: Next Generation Kernel Network Tunnel (2017)

J. A. Donenfeld [10] presents WireGuard, a protocol for secure network tunnels that uses cryptographic mecha-nisms to protect peer communication.

II.11 A Function-as-a-Service Middleware for Decentralized Collaborative Edge Computing (2026)

C. Gonçalves et al. [11] present Function-as-a-Service middleware for decentralized edge computing, enabling edge nodes to discover one another and collaborate on function execution.

II.12 Behavioural Types for Local-First Software (2023)

R. Kuhn et al. [12] study behavioral types for local-first software, using protocol-based coordination and event logs to describe communication among autonomous nodes.

II.13 Resource-Aware Deployment of Dynamic DNNs over Multi-Tiered Interconnected Systems (2024)

C. Singhal et al. [13] present a scheme for deploying dynamic deep neural networks across mobile, edge, and cloud environments by placing model components ac-cording to resource conditions.

II.14 Scheduling of Distributed Applications on the Computing Continuum: A Survey (2023)

N. Mehran et al. [14] survey scheduling strategies for dis-tributed applications across the computing continuum, considering execution time, energy, and cost.

II.15 Comprehensive Survey on Resource Management for IoT Applications in Edge-Fog-Cloud Environments (2025)

N. E. H. Boubaker et al. [15] survey resource manage-ment for IoT applications in edge-fog-cloud environ-ments, including resource allocation, task scheduling, computation offloading, and service migration.

II.16 Decentralised Global Service Discovery for the Internet of Things (2024)

R. Kurte et al. [16] present a decentralized service-discovery approach for IoT systems using distributed hash tables, cryptographic service identifiers, and digital signatures on metadata.

II.17 Supporting Flexible and Transparent User Interface Distribution Across Mobile Devices (2024)

S. Oh et al. [17] propose a mechanism for distribut-ing interface components across mobile devices through graphics-state serialization and cross-device communi-cation.

II.18 Opportunistic Nudges for Task Migration Between Personal Devices (2024)

N. Joshi et al. [18] examine how visual prompts and device interactions can facilitate cross-device task mi-gration and context transfer.

II.19 Resource-Aware Federated Data Analytics in Edge-Enabled IoT Systems (2024)

H. Khamfroush [19] studies resource-aware federated data analytics in edge-enabled IoT systems. Participat-ing devices compute over local data and send model parameters instead of transferring all data to one loca-tion.

II.20 A Distributed Learned Hash Table (2025)

S. Wang et al. [20] propose a distributed learned hash table that uses learned models for data location and retrieval. The model supports search and range queries in distributed systems.

II.21 End-to-End Slices to Orchestrate Resources and Services in the Cloud-to-Edge Continuum (2023)

F. Tusa and S. Clayman [21] present a method for orchestrating computing and network resources, as well as services, between the cloud and edge using end-to-end slices.

II.22 A Decentralized Approach for Resource Discovery Using Metadata Replication in Edge Networks (2022)

I. Murturi and S. Dustdar [22] propose a decentralized framework for resource discovery using metadata repli-cation in edge networks. This framework allows devices to share and store resource metadata independently of a centralized repository.

II.23 Distributed Resource Scheduling in Edge Computing: Problems, Solutions, and Opportunities (2022)

Y. Sahni et al. [23] review methods for distributed re-source scheduling in edge computing, including compu-tation, storage, networking, device heterogeneity, and resource variability.

II.24 BarMan: A Run-Time Management Framework in the Resource Continuum (2022)

M. Zanella et al. [24] present BarMan, a runtime man-agement framework that assigns tasks across heteroge-neous resources while accounting for changes in resource availability and system performance.

II.25 Resource Provisioning and Allocation in Function-as-a-Service Edge-Clouds (2022)

O. Ascigil et al. [25] examine resource provisioning and allocation for latency-critical FaaS workloads in edge-cloud settings, including centralized and decentralized allocation policies.

III Proposed Methodology

III.1 Research Gap

III.1.1 Cloud-Centric Orchestration and Local-First Offline Resilience

The literature distinguishes cloud-centric distributed computing from local-first systems in their approaches to resource coordination, network connectivity, and of-fline operation. Work on distributed and edge-cloud computing studies resource management, scheduling, service orchestration, and load balancing across dis-tributed infrastructure [5], [14], [15], [21], [23]. These approaches generally coordinate infrastructure whose participating components communicate over a network.

Local-first computing emphasizes local data owner-ship, local computation, and continued operation with-out a persistent connection to a central cloud service [2], [3], [12]. Discovery mechanisms such as mDNS and DNS-SD provide zero-configuration discovery on local networks [6], [7]. Local-first approaches therefore sup-port operation during intermittent connectivity, while cloud and edge research offers methods for resource coordination and management.

DaemonONE aims to combine local-first operation with a distributed network of personal devices. It is designed to retain local metadata and the last-known state of a peer when that peer goes offline, allowing users to view their computing environment without a central cloud service.

III.1.2 Mismatch in Resource Awareness Scope and Abstraction Levels

The reviewed literature addresses resource awareness at different scopes and abstraction levels. Local-first research focuses on application data, state replication, consistency, and coordination among application repli-cas [2], [3], [9], [12]. It provides mechanisms for manag-ing application state during disconnection but generally does not address system-level developer resources such as active database services, Git repositories, container-ized applications, or host-level resource telemetry.

In contrast, research on distributed resource man-agement, edge computing, FaaS, and infrastructure or-chestration focuses on allocation and execution within larger, heterogeneous infrastructures [11], [13], [14], [15], [21], [24], [25]. These studies address computing capac-ity, virtual infrastructure, workloads, and services in edge-cloud or multi-domain environments.

This creates an abstraction-level gap in the reviewed literature. Local-first systems support autonomy and of-fline operation, while distributed resource management systems provide methods for resource awareness and workload coordination. Their integration for personal computers remains underdeveloped. DaemonONE ad-dresses this gap with a logical registry of application resources, development services, and host resource in-formation.

III.1.3 Execution Security Trade-Offs in Distributed Environments

The reviewed literature shows a trade-off between execu-tion flexibility and access control in distributed systems. Edge FaaS and distributed execution platforms provide tools for controlled workload execution [11], [21], [24], [25], but generally focus on functions, services, and managed workloads rather than developers' tasks.

In a personal multi-device setup, remote shell ac-cess offers flexibility but increases security risks. Dae-monONE instead restricts remote execution to approved development tasks. Each target daemon checks the re-quested task against its local whitelist before starting a process. The initial task set is cargo build, cargo test, git status, docker compose up, and docker compose down; interactive shell access is outside the system's scope.

III.2 Proposed System

DaemonONE is designed to address resource fragmenta-tion, cloud dependency, and offline fragility in personal multi-device software development. Unlike enterprise cluster managers and cloud orchestrators, it offers a local-first peer-to-peer environment for one software developer's personal devices.

The following subsections describe the DaemonONE architecture, discovery protocols, metadata indexing, offline-aware operation, resource monitoring, and con-trolled task execution.

III.2.1 System Architecture

DaemonONE works on a decentralized, peer-to-peer, and local-first model, meaning that data processing and storage are done locally without depending on the centralized cloud servers. Devices share their resource metadata so that a complete overview of the computing environment is created while keeping local control.

A lightweight daemon named daemononed runs on each device to track its resources. Laptops, desktops, home servers, and lab nodes retain their own operating systems, hardware, storage, and services. DaemonONE tracks resources but does not physically consolidate CPU cores, memory, or storage. The architecture is shown in Figure 1.

III.2.2 Device Discovery and Peer Communication

DaemonONE uses zero-configuration discovery and peer authentication to connect devices on a local network without manual infrastructure configuration.

1. **Zero-configuration link discovery:** At start-up, each DaemonONE instance discovers partici-pating devices and their advertised services on the local network using Multicast DNS (mDNS) [6] and DNS-based Service Discovery (DNS-SD) [7]. Both protocols support device or service discovery without a DNS server or manual configuration [6], [7].

2. **Peer authentication:** Connections are limited to devices approved by the user. Each DaemonONE instance authenticates its authorized peers. The secure transport mechanism remains to be specified; WireGuard provides one relevant encrypted-tunnel approach [10].

3. **Structured payload exchange:** After establish-ing a secure connection, each instance exchanges structured payloads containing node identity, lo-cal resource status, active-service information, and permitted tasks.

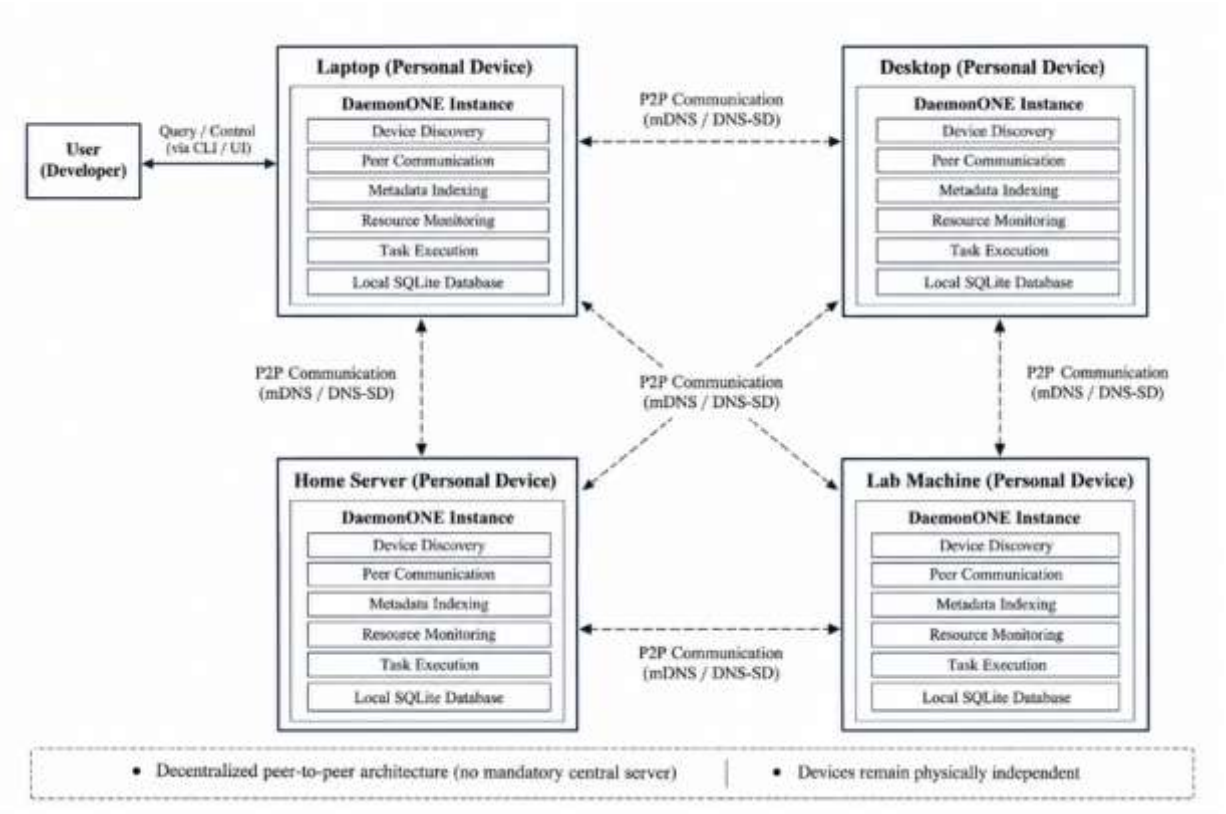


Figure 1: DaemonONE's local-first, peer-to-peer architecture across personal devices and their local daemons.

III.2.3 Metadata Indexing and Unified Resource Registry

Instead of physically moving data, DaemonONE main-tains a five-layer logical resource registry. Each local daemon inspects the file system, active operating-system processes, container interfaces, and project folders on its device, then stores metadata in an embedded lo-cal database. The registry structure is summarized in Table 1.

The five-layer logical resource registry provides a unified view of the developer's heterogeneous devices and identifies the device that physically stores each asset.

III.2.4 Offline-Aware Metadata Management

Operational continuity during network disconnection is a core local-first principle [2]. DaemonONE's offline-aware behavior is designed to support the following:

1. **Last-known state persistence:** When a device is unreachable because of a network or power fail-ure, other active daemons retain its last-known metadata, services, and resource state in their local indexes.
2. **Reconnection synchronization:** When a discon-nected device reconnects, participating daemons exchange updated local index data. The conflict-resolution policy for concurrent changes remains to be specified and evaluated.
3. **Local-first operation:** Users can query the logi-cal registry and view previously stored details for resources associated with disconnected devices, con-sistent with local-first principles [2].

III.2.5 Resource and Service Monitoring

DaemonONE is designed to monitor host hardware resources and development services in near real time. The monitoring involves three activities:

- **System-resource telemetry:** Local daemons monitor CPU and RAM usage, available disk space, network connectivity, and node availability. Re-lated work on locality-aware tracing informs analy-sis of service relationships [8].
- **Service and process monitoring:** Using operating-system process data and container in-terfaces such as the Docker API, DaemonONE is designed to detect and register active services, in-cluding PostgreSQL, MySQL, Redis, containerized services, and local web servers.
- **Logical abstraction:** Hardware metrics and ser-vice information are represented in the consolidated resource registry for developer visibility. Hardware resources are not physically combined or dynami-cally pooled across participating nodes.

Table 1: The five-layer logical resource registry records data, projects, services, host resources, and approved execution tasks for each device.

Layer	Description & tracked entities	System scope & function
1. Data Layer	File names, paths, folder metadata, file sizes, and modification timestamps	Identifies the physical host location of files; not a distributed filesystem
2. Project Layer	Git repositories, active branches, environment files (.env), and build targets	Maps software project locations and active source-code repositories
3. Service Layer	PostgreSQL, MySQL, Redis, Docker containers, local API servers, and web ports	Indexes active backend services and accessible development ports
4. Resource Layer	CPU utilization, RAM usage, available disk capacity, and network latency	Provides hardware telemetry; task-placement policies remain future work
5. Execution Layer	Whitelisted tasks such as cargo build, cargo test, git status, docker compose up, and docker compose down	Authorizes and routes pre-approved development tasks

III.2.6 Cross-Device Search and Controlled Task Execution

Cross-device search. Developers can use a command-line or desktop interface to find project repos-itories, files, environment settings, running containers, or hardware capabilities across connected devices. Re-sults identify the device that contains each resource. The registry supporting these queries is summarized in Table 1.

Controlled task execution. DaemonONE is de-signed to launch approved build or test processes on another personal device, such as running a resource-intensive build on a desktop while working from a laptop. The initial task set is cargo build, cargo test, git status, docker compose up, and docker compose down.

- **Pre-execution validation:** The target daemon checks each requested task against its local autho-rization whitelist before starting a child process.
- **Output streaming:** The target daemon captures standard output, standard error, and the process exit code, then returns them to the requesting device.
- **Scope boundary:** Arbitrary remote shell execu-tion, including interactive SSH or shell sessions, is outside the system's scope.

IV Findings

The literature review and the DaemonONE design high-light how distributed computing systems address re-source discovery, state consistency, execution safety, and offline resilience. The comparison in Table 2 summarizes these findings across the approaches surveyed. Related work also provides precedents in overlay resource access [1], locality-aware tracing [8], resource-aware deploy-
ment [13], and federated analytics [19].

Summary of Findings and System Design Im-plications. The reviewed domains offer components relevant to DaemonONE, including zero-configuration device and service discovery [6], [7], local-first comput-ing [2], [3], [9], decentralized resource discovery [16], [22], distributed resource management [5], [14], [15], [23], and controlled distributed execution [11], [24], [25]. These features are generally studied in other applications or deployment contexts. DaemonONE combines a local-first peer-to-peer approach, a five-layer logical resource registry for data, projects, services, system resources, and controlled execution, and an authorization model for remote development tasks. It integrates capabilities from the reviewed literature and is not a substitute for the individual systems surveyed.

V Conclusion

Personal computing environments increasingly include laptops, desktop computers, and home or laboratory servers. Projects, repositories, development services, and system resources may still be managed separately on each device. This creates an opportunity to combine local-first computing, decentralized device and service discovery, metadata management, inter-device commu-nication, and distributed resource management in one personal computing environment. This paper proposed **DaemonONE**, a local-first distributed computing plat-form that provides a unified logical view of personal devices through lightweight background daemons, de-centralized device discovery, offline-aware metadata han-dling, and a five-layer logical resource registry covering data, projects, services, resources, and execution.

The proposed architecture provides a basis for coor-dinating development resources across personal devices, but it has not yet been empirically evaluated for perfor-mance, usability, or efficiency. Future work will imple-ment DaemonONE across diverse devices and measure resource-discovery and metadata-synchronization over-head, task-execution latency, monitoring accuracy, and usability. Further work may address stronger execution security policies, conflict resolution during disconnection and reconnection synchronization, discovery of relation-ships between development services, and resource-aware task scheduling.

Table 2: Comparison of centralized cloud, edge FaaS, local-first, peer-to-peer discovery, and DaemonONE approaches across network topology, cloud dependence, offline operation, resource scope, execution, and deployment context.

Dimension	Centralized Cloud and Cloud Slicing [4], [21]	Edge FaaS and Distributed Middleware [11], [25]	Local-First Software [2], [3], [9]	P2P Discovery & Resource Lookup [16], [20], [22]	Proposed DaemonONE
Network topology	Cloud/edge infrastructure with centralized or multi-domain coordination [4], [21]	Distributed edge/cloud execution [11], [25]	Local replicas with peer synchronization [2], [3], [9]	Decentralized P2P overlays and clusters [16], [22]	Decentralized personal-device P2P
Cloud dependency	Cloud/edge infrastructure is part of the deployment model [4], [21]	Edge-cloud infrastructure [11], [25]	Emphasizes local operation and reduced dependence on centralized services [2]	Decentralized discovery and lookup [16], [22]	No mandatory central cloud server
Offline operation	Depends on participating infrastructure and network connectivity [4], [21]	Depends on participating edge/cloud resources and connectivity [11], [25]	Strong emphasis on local operation during disconnection [2], [3], [9]	Provides decentralized lookup when the peer overlay is reachable; offline behavior is not specified [16], [20], [22]	Retains last-known metadata and device status
Resource scope	Compute, network, infrastructure, and service resources [4], [21]	Functions, workloads, and edge-cloud resources [11], [25]	Application data and replicated state [2], [3], [9]	Services, resources, metadata, and distributed lookup [16], [20], [22]	Five-layer logical registry covering data, projects, services, resources, and execution
Execution model	Distributed work-load/resource orchestration [4], [21]	Function- and service-oriented execution [11], [25]	Primarily application-level local execution and synchronization [2], [3]	Primarily discovery and resource lookup [16], [20], [22]	Controlled execution of predefined development tasks
Deployment context	Cloud-to-edge and distributed infrastructure [4], [21]	Edge-cloud and distributed service environments [11], [25]	Local-first collaborative applications [2], [3], [9]	IoT, edge, and decentralized resource environments [16], [20], [22]	Personal multi-device developer environment

References

[1] M. Simić, J. Dedeić, M. Stojkov, and I. Prokić, "Data overlay mesh in distributed clouds allowing collabo-rative applications," *IEEE Access*, vol. 13, pp. 6180–6203, 2025, https://doi.org/10.1109/ACCESS.2024. 3525336.

[2] M. Kleppmann, A. Wiggins, P. van Hardenberg, and M. McGranaghan, "Local-first software: You own your data, in spite of the cloud," in *Proc. ACM SIG-PLAN Int. Symp. New Ideas, New Paradigms, and Reflections on Programming and Software (Onward! '19)*, 2019, pp. 154–178, https://doi.org/10.1145/ 3359591.3359737.

[3] J. Haas, R. Mogk, E. Yanakieva, A. Bieniusa, and M. Mezini, "LoRe: A programming model for veri-fiably safe local-first software," in *Proc. 37th Euro-pean Conf. Object-Oriented Programming (ECOOP 2023)*, LIPIcs, vol. 263, 2023, pp. 12:1–12:15, https: //doi.org/10.4230/LIPIcs.ECOOP.2023.12.

[4] Y. Afek, G. Giladi, and B. Patt-Shamir, "Distributed computing with the cloud," *Distributed Computing*, vol. 37, pp. 1–18, 2024, https://doi.org/10.1007/ s00446-024-00460-w.

[5] P. Czarnul et al., "Optimization of resource-aware paral-lel and distributed computing: A review," *The Journal of Supercomputing*, vol. 81, no. 7, Art. no. 848, 2025, https://doi.org/10.1007/s11227-025-07295-7.

[6] S. Cheshire and M. Krochmal, "Multicast DNS," RFC 6762, Internet Engineering Task Force, Feb. 2013, https://doi.org/10.17487/RFC6762.

[7] S. Cheshire and M. Krochmal, "DNS-based service dis-covery," RFC 6763, Internet Engineering Task Force, Feb. 2013, https://doi.org/10.17487/RFC6763.

[8] C. Colarusso, A. De Caro, I. Falco, L. Goglia, and E. Zimeo, "A distributed tracing pipeline for improving locality awareness of microservices applications," *Soft-ware: Practice and Experience*, vol. 54, no. 6, pp. 1118–1140, 2024, https://doi.org/10.1002/spe.3317.

[9] N. Preguiça, "Conflict-free replicated data types: An overview," arXiv:1806.10254, 2018.

[10] J. A. Donenfeld, "WireGuard: Next generation ker-nel network tunnel," in *Proc. Network and Dis-tributed System Security Symp. (NDSS)*, San Diego, CA, USA, 2017, pp. 1–12, https://doi.org/10.14722/ ndss.2017.23160.

[11] C. Gonçalves, J. Simão, and L. Veiga, "A function-as-a-service middleware for decentralized collaborative edge computing," *Future Generation Computer Systems*, vol. 175, Art. no. 108069, 2026, https://doi.org/10.1016/ j.future.2025.108069.

[12] R. Kuhn, H. C. Melgratti, and E. Tuosto, "Behavioural types for local-first software," in *Proc. 37th Euro-pean Conf. Object-Oriented Programming (ECOOP 2023)*, LIPIcs, vol. 263, 2023, pp. 15:1–15:28, https: //doi.org/10.4230/LIPIcs.ECOOP.2023.15.

[13] C. Singhal, Y. Wu, F. Malandrino, M. Levorato, and C. F. Chiasserini, "Resource-aware deployment of dynamic DNNs over multi-tiered interconnected systems," in *Proc. IEEE INFOCOM 2024*, Vancouver, BC, Canada, 2024, https://doi.org/10.1109/INFOCOM52122.2024. 10621218.

[14] N. Mehran, D. Kimovski, H. Hellwagner, D. Roman, A. Soylu, and R. Prodan, "Scheduling of distributed applications on the computing continuum: A sur-vey," in *Proc. IEEE/ACM 16th Int. Conf. Utility and Cloud Computing (UCC 2023)*, 2023, pp. 1–6, https://doi.org/10.1145/3603166.3632540.

[15] N. El Houda Boubaker, K. Zarour, N. Guermouche, and D. Benmerzoug, "A comprehensive survey on resource management for IoT applications in edge-fog-cloud environments," *IEEE Access*, vol. 13, pp. 111892–111925, 2025, https://doi.org/10.1109/ ACCESS.2025.3583584.

[16] R. Kurte, Z. Salcic, and K. I.-K. Wang, "Decen-tralised global service discovery for the Internet of Things," *Sensors*, vol. 24, no. 7, Art. no. 2196, 2024, https://doi.org/10.3390/s24072196.

[17] S. Oh, A. Kim, S. Lee, K. Lee, D. R. Jeong, S. Y. Ko, and I. Shin, "Supporting flexible and transpar-ent user interface distribution across mobile devices," *IEEE Transactions on Mobile Computing*, vol. 23, no. 8, pp. 8398–8417, 2024, https://doi.org/10.1109/TMC. 2024.3349561.

[18] N. Joshi et al., "Opportunistic nudges for task migration between personal devices," in *Extended Abstracts of the 2024 CHI Conference on Human Factors in Computing Systems (CHI EA '24)*, Honolulu, HI, USA, 2024, Art. no. 256, pp. 1–8, https://doi.org/10.1145/3613905. 3651037.

[19] H. Khamfroush, "Resource-aware federated data ana-lytics in edge-enabled IoT systems," *Proceedings of the AAAI Symposium Series*, vol. 3, no. 1, p. 305, 2024, https://doi.org/10.1609/aaaiss.v3i1.31219.

[20] S. Wang, Y. Liu, X. Zhang, L. Hu, and C. Qian, "A distributed learned hash table," in *Proc. IEEE 33rd Int. Conf. on Network Protocols (ICNP)*, Seoul, South Korea, 2025, pp. 1–11, https://doi.org/10.1109/ ICNP65844.2025.11192384.

[21] F. Tusa and S. Clayman, "End-to-end slices to or-chestrate resources and services in the cloud-to-edge continuum," *Future Generation Computer Systems*, vol. 141, pp. 473–488, 2023, https://doi.org/10.1016/j. future.2022.11.026.

[22] I. Murturi and S. Dustdar, "A decentralized approach for resource discovery using metadata replication in edge networks," *IEEE Transactions on Services Com-puting*, vol. 15, no. 5, pp. 2526–2537, 2022, https: //doi.org/10.1109/TSC.2021.3082305.

[23] Y. Sahni, J. Cao, L. Yang, and S. Wang, "Distributed resource scheduling in edge computing: Problems, so-lutions, and opportunities," *Computer Networks*, vol. 219, Art. no. 109430, 2022, https://doi.org/10.1016/ j.comnet.2022.109430.

[24] M. Zanella, F. Sciamanna, and W. Fornaciari, "Bar-Man: A run-time management framework in the re-source continuum," *Sustainable Computing: Infor-matics and Systems*, vol. 35, Art. no. 100663, 2022, https://doi.org/10.1016/j.suscom.2022.100663.

[25] O. Ascigil, A. G. Tasiopoulos, T. K. Phan, V. Sourlas, I. Psaras, and G. Pavlou, "Resource provisioning and allocation in function-as-a-service edge-clouds," *IEEE Transactions on Services Computing*, vol. 15, no. 4, pp. 2410–2424, 2022, https://doi.org/10.1109/TSC. 2021.3052139.