OS-1: Towards Autonomous Operating Systems with Context-Aware Intelligence

Soham Datta
AI Researcher
University of Pune
Department of Artificial Intelligence and Machine Learning
thesohamdatta@gmail.com

***Abstract*—Modern operating systems remain fundamentally constrained by interaction paradigms established over four decades ago. The Windows, Icons, Menus, Pointer (WIMP) model forces users into manual, app-centric workflows that consume significant cognitive resources and time. Current AI assistants, while increasingly capable, operate as siloed add-ons lacking deep system integration and holistic context awareness. We propose OS-1, a vision for next-generation operating systems that fundamentally reimagines the human-computer relationship through intent-centric computing. OS-1 introduces a five-layer architecture featuring: (1) a unified context engine that fuses multimodal signals into holistic situational awareness, (2) a voice-to-action planner using Belief-Desire-Intention models for autonomous workflow orchestration, (3) a personalized memory system providing continuous learning, (4) an affective computing module for emotional intelligence, and (5) privacy-first design through federated learning and differential privacy. We present the architectural framework, analyze technical feasibility using modern hardware capabilities, identify key research challenges, and discuss implications for human-AI collaboration. This work establishes a blueprint for operating systems that serve as intelligent partners rather than passive tools.**

***Index Terms*—Operating systems, context-aware computing, affective computing, federated learning, human-AI interaction, intent-centric computing, voice interfaces**

I. Introduction

A. Motivation

The foundational paradigm of personal computing—established with the Xerox Alto in 1973 and popularized by the Macintosh in 1984—remains largely unchanged despite over four decades of technological advancement [1]. Users still navigate through nested menus, manage application windows manually, and execute multi-step workflows that could be automated. Research in human-computer interaction consistently demonstrates that users spend substantial time on repetitive, low-value tasks that interrupt higher-level cognitive work [2]. This inefficiency persists despite dramatic improvements in processing power, storage capacity, and connectivity.

Contemporary AI assistants (Siri, Alexa, Google Assistant, Windows Copilot) represent incremental improvements rather than paradigm shifts. They operate reactively, responding to explicit commands without anticipating user needs. They function in isolation from deep system context, accessing limited information about user workflows, emotional states, or long-term patterns. Most critically, they lack the agency to autonomously orchestrate complex, multi-application tasks—the hallmark of true operating system integration [3].

B. Vision: Intent-Centric Computing

We propose OS-1, a conceptual framework for autonomous operating systems that shifts computing from tool-centric to partner-centric interaction. The core innovation is *intent-centric computing at the operating system level*: unlike application-specific automation (Siri Shortcuts, IFTTT) or cloud-dependent assistants, OS-1 enables users to express **high-level goals** in natural language while the system autonomously orchestrates necessary actions across all applications with full system context and privacy guarantees.

Consider an illustrative scenario: A user says, "Prepare for tomorrow's client presentation." A traditional OS requires manual steps: opening calendar, reviewing meeting details, locating relevant documents, checking email threads, preparing slides, setting reminders. An OS-1 system would: identify the meeting, retrieve related documents and communications, summarize key discussion points, prepare a briefing document, set contextual reminders, and proactively suggest preparation tasks—all autonomously, securely, and with explainable reasoning.

C. Key Contributions

This vision paper makes the following contributions:

- **Architectural Framework**: A five-layer architecture for intent-centric, context-aware operating systems integrating unified context engines, autonomous planning, personalized memory, affective adaptation, and privacy-first learning.
- **Feasibility Analysis**: Examination of modern hardware capabilities (neural processing units, secure enclaves, edge AI accelerators) demonstrating technical viability of **on-device** intelligent systems.
- **Research Challenges**: Identification of open problems in context scaling, cross-application orchestration, trust calibration, and privacy-preserving personalization.
- **Comparison Framework**: Systematic analysis positioning OS-1 against existing approaches (Apple Intelligence,

Windows Copilot, Android AI) highlighting gaps and opportunities.

D. Paper Organization

Section II surveys related work in intelligent operating systems, context-aware computing, and affective technologies. Section III defines core design principles. Section IV presents the proposed architecture. Section V analyzes technical feasibility. Section VI identifies research challenges. Section VII discusses broader impact, and Section VIII concludes.

II. Background and Related Work

A. Evolution of Operating System Paradigms

Operating systems have evolved through distinct epochs: batch processing (1950s-1960s), time-sharing (1960s-1970s), personal computing with GUIs (1980s-present), and mobile computing (2000s-present) [4]. Each transition corresponded to fundamental shifts in computing scale and user interaction models. We propose that we are entering a fifth epoch: *intelligent, agentic operating systems*.

Early attempts at intelligent systems include Plan 9 [5], which emphasized distributed computing, and BeOS, which prioritized multimedia responsiveness. The Active Database Systems movement [6] explored proactive data management. However, these efforts preceded the AI capabilities, hardware resources, and privacy frameworks necessary for true system intelligence.

B. Context-Aware Computing

Context-aware computing, introduced by Schilit et al. [7] and formalized by Dey and Abowd [8], [9], seeks to leverage situational information to enhance system responsiveness. Dey defines context as "any information that can be used to characterize the situation of an entity" [9]. Early systems focused on location awareness; modern work encompasses identity, activity, time, environmental conditions, and social context [10].

Despite two decades of research, context awareness remains largely confined to specific applications rather than integrated at the OS level. The Context Toolkit [11] provided programming abstractions, but system-wide context fusion with privacy guarantees remains an open challenge.

C. Affective Computing

Affective computing, pioneered by Picard [12], aims to enable systems to recognize, interpret, and respond to human emotions. Recent advances in multimodal emotion recognition—analyzing facial expressions, speech prosody, physiological signals, and text sentiment—have improved accuracy significantly [13]. Contemporary research explores integrating affective capabilities into large language models and conversational AI systems [14].

However, most affective systems operate as standalone applications. Integrating emotional intelligence system-wide—adapting interfaces, notifications, and workflows to user cognitive and emotional states—remains unrealized in mainstream operating systems.

D. Privacy-Preserving Machine Learning

Federated learning (FL), introduced by McMahan et al. [15], enables model training across distributed devices without centralizing raw data. Differential privacy (DP), formalized by Dwork [16], provides mathematical guarantees against information leakage. Combining FL and DP offers principled approaches to **on-device** personalization [17], [18].

Recent work by Google [19] and Meta [20] demonstrates production deployment of FL-DP systems for keyboard prediction and content recommendation. These advances provide foundational techniques for privacy-first operating systems.

E. Current AI-Enhanced Operating Systems

Apple Intelligence integrates **on-device** language models for writing assistance, notification summaries, and Siri improvements [21]. While featuring strong privacy protections through **on-device** processing, it remains largely app-centric without holistic cross-application context integration.

Windows Copilot provides an AI overlay for productivity tasks [22]. However, it operates primarily as a chatbot interface rather than a system-native intelligence, with significant cloud dependency for processing.

Android AI initiatives aim to provide centralized AI services across the platform [23]. Early indications suggest focus on generative AI features (image generation, smart replies) rather than comprehensive context awareness and autonomous workflow orchestration.

AIOS (LLM Agent OS) [24] proposes kernel-level abstractions for LLM agent management, including scheduling, memory management, and access control. While addressing important resource orchestration challenges, AIOS focuses on multi-agent coordination rather than holistic human-computer interaction reimagining.

Gap Analysis: Existing approaches make important progress but lack: (1) unified, multimodal context engines spanning all system activity, (2) proactive, goal-oriented workflow orchestration across applications, (3) emotional intelligence integrated throughout the interaction model, (4) formal privacy guarantees for continuous learning, and (5) architectural vision positioning the OS as autonomous partner rather than reactive tool.

III. OS-1 Design Principles

We propose five core design principles that distinguish OS-1 from existing systems:

A. Principle 1: Intent-Centric Interaction

Current Paradigm: Users specify *how* to accomplish tasks through explicit command sequences (open application, navigate menu, select option, etc.).

OS-1 Paradigm: Users specify *what* they want to accomplish through high-level goals. The system autonomously determines and executes necessary steps.

This shift from procedural to declarative interaction reduces cognitive load and enables users to focus on objectives rather

than mechanics. Natural language serves as the primary interface, with the system maintaining rich internal representations of user intent, context, and constraints.

B. Principle 2: Holistic Multimodal Context

Current Paradigm: Applications maintain isolated state; context is fragmented across app silos.

OS-1 Paradigm: A unified context engine continuously fuses information from applications, files, calendar, communications, sensors (location, activity, ambient sound), and biometric indicators (if available and consented) into a holistic situational model.

This enables the system to understand not just individual data points but their relationships and implications. For example, detecting that a user is in a loud environment, has an approaching deadline, and recently received urgent messages allows appropriate prioritization and response adaptation.

C. Principle 3: On-Device Privacy

Current Paradigm: AI assistants typically rely on cloud processing, creating privacy vulnerabilities and dependencies.

OS-1 Paradigm: All context processing, personalization, and inference occur **on-device** using modern neural processing units. Optional model improvements use federated learning with differential privacy, ensuring no raw data leaves the device.

This architectural choice prioritizes user sovereignty over data while leveraging advances in edge AI hardware and model compression techniques.

D. Principle 4: Affective Adaptation

Current Paradigm: Interfaces and notifications treat all moments as equivalent, interrupting without regard to user state.

OS-1 Paradigm: The system continuously infers user cognitive and emotional state (focused, stressed, fatigued, creative) from behavioral patterns and contextual signals. Interfaces, notifications, and suggestions adapt accordingly.

For instance, during high-focus periods, non-critical notifications are deferred; during stressed states, calming interface elements and supportive suggestions emerge. This transforms the OS from neutral tool to empathetic partner.

E. Principle 5: Proactive, Trustworthy Automation

Current Paradigm: Systems wait for explicit commands; automation requires complex scripting or third-party tools.

OS-1 Paradigm: The system learns routines and patterns, proactively suggests or executes beneficial automations with appropriate confidence thresholds. All actions are logged, explainable, and easily reversible.

Trust emerges from transparency and control. Users maintain ultimate authority through granular permissions, action previews, and comprehensive audit logs. The system explains its reasoning and learns from user corrections.

IV. Proposed Architecture

OS-1's architecture comprises five interconnected layers that work in concert to enable intent-centric, context-aware computing.

A. Layer 1: Unified Context Engine

The Context Engine serves as the foundational layer, continuously constructing a holistic model of user state, environment, and activity.

1) Data Sources:

- **Application Activity**: Foreground/background apps, active documents, browsing history, content consumption patterns
- **Communications**: Email metadata, message sentiment, meeting schedules, social interactions
- **File System**: Document access patterns, creation/modification timestamps, content relationships
- **Sensors**: Location, motion (walking, stationary), ambient sound level, time of day, weather
- **Behavioral Signals**: Typing speed, mouse movement patterns, pause durations, error rates
- **Biometric Data** (optional, with explicit consent): Heart rate, stress indicators from wearables

2) Fusion Mechanism: Context fusion employs transformer-based architectures [25] with attention mechanisms to weigh signal importance dynamically. Each data stream is embedded into a common semantic space, enabling cross-modal reasoning.

The Context Engine fuses diverse signals into a single Context State Vector C_t. This fusion can be conceptually modeled as:

$$C_t = \mathcal{F}(E_t, A_t, M_t, F_t, S_t)$$

where $\mathcal{F}$ is the multimodal fusion function (e.g., Transformer-based), and E_t, A_t, M_t, F_t, and S_t represent the embedded states of the Environment, Application Activity, Communications, File System, and Sensor inputs, respectively, at time t.

Temporal modeling captures context across multiple timescales:

- **Immediate** (<1 minute): Current activity, active application
- **Short-term** (1-60 minutes): Current task, recent actions
- **Medium-term** (hours-days): Daily routines, ongoing projects
- **Long-term** (weeks-months): Habits, preferences, relationships

Context updates would target sub-50ms latency for real-time responsiveness in a production implementation.

3) Privacy Protections: All context data remains encrypted at rest using AES-256 or stronger. Fusion processing occurs within hardware-isolated secure enclaves (e.g., ARM TrustZone, Intel SGX) where available. Users maintain comprehensive control over what data sources are included and can inspect, edit, or delete any stored context.

B. Layer 2: Personalized Memory System

The Memory System provides long-term retention of user patterns, preferences, and interactions, enabling true personalization without cloud dependency.

1) Memory Types: Following cognitive science models [26], OS-1 maintains multiple memory stores:

- **Episodic Memory**: Specific events and experiences (e.g., "prepared presentation on March 15")
- **Semantic Memory**: Facts and knowledge (e.g., "prefers dark mode interfaces")
- **Procedural Memory**: Skills and routines (e.g., "typically reviews email before meetings")
- **Affective Memory**: Emotional associations (e.g., "stressed when deadline proximity increases")

2) Storage and Retrieval: Memories are stored in vector databases enabling semantic similarity search [27]. When context changes or user issues queries, relevant memories are retrieved to inform system responses. Storage uses hierarchical compression: recent memories retain full detail; older memories are progressively compressed while preserving semantic content.

3) Federated Learning Integration: Local memory patterns can contribute to optional global model improvements via federated learning [17]. Differential privacy mechanisms [16] ensure that even gradient updates cannot leak individual user information. Participation is opt-in, and users can verify what patterns are shared via privacy dashboards.

C. Layer 3: Voice-to-Action Planner

The Voice-to-Action Planner translates natural language goals into executable workflows across multiple applications.

1) Pipeline Stages:

1) **Speech Recognition**: On-device automatic speech recognition (ASR) converts voice input to text. Modern neural ASR models (e.g., RNN-Transducers [28]) achieve near-human accuracy with $<$200ms latency on current mobile SoCs.
2) **Intent Extraction**: Natural language understanding (NLU) identifies user goals, entities, and constraints. Fine-tuned language models (e.g., distilled BERT variants [29]) operating **on-device** extract structured intent representations.
3) **Context Integration**: The extracted intent is enriched with relevant context from the Context Engine and Memory System. Ambiguous references ("that document" $\rightarrow$ specific file based on recent activity) are resolved.
4) **Goal Decomposition**: Complex goals are decomposed into subgoals using Belief-Desire-Intention (BDI) planning [30]. The BDI framework represents system beliefs (current state), desires (goals), and intentions (committed plans).
5) **Action Planning**: For each subgoal, the planner identifies necessary application actions. Template libraries encode common workflows; neuro-symbolic reasoning [31] handles novel scenarios by combining learned patterns with logical constraints.
6) **Execution and Monitoring**: Actions are executed via accessibility APIs and application automation interfaces. The system monitors execution, detects failures, and adapts plans dynamically. Users receive real-time feedback and can intervene at any point.

2) Learning from Interaction: When users modify, reject, or correct system actions, these interactions become training signals for improving future planning. Online learning techniques [32] enable continuous adaptation without requiring extensive offline retraining.

D. Layer 4: Affective Computing Module

The Affective Module enables emotional intelligence throughout the system.

1) Emotional State Inference: Emotion recognition integrates multiple modalities [13]:

- **Behavioral Patterns**: Typing rhythm, mouse dynamics, pause patterns correlate with cognitive load and stress [33]
- **Contextual Signals**: Deadline proximity, task difficulty, interruption frequency
- **Communication Sentiment**: Natural language processing of messages, emails
- **Physiological Data** (opt-in): Heart rate variability, galvanic skin response from wearables

A continuous emotion model would estimate valence (positive-negative), arousal (calm-excited), and discrete states (focused, stressed, creative, fatigued) updated every 30 seconds in a production system.

2) Adaptive Responses: The system adapts multiple dimensions based on inferred state:

- **Notification Management**: Critical notifications only during high-stress; deferred non-urgent items during focus
- **Interface Elements**: Calming colors and animations during stress; energizing elements during low arousal
- **Proactive Support**: Suggesting breaks during extended focus, offering organization help when overwhelmed
- **Communication Assistance**: Draft tone adjustment based on sender relationship and user state

3) Wellbeing Support: Beyond productivity optimization, affective intelligence supports digital wellbeing. The system can detect patterns indicating burnout risk, suggest schedule adjustments, and encourage healthy boundaries. All interventions are suggestion-based, preserving user autonomy.

E. Layer 5: Privacy and Security Framework

Privacy is not a feature but a foundational architectural constraint permeating all layers.

1) Data Minimization: The system collects only necessary data, with clear purpose specification. Temporary context (e.g., current ambient sound level) is processed and discarded rather than stored. Long-term storage is user-controlled with automatic expiration policies.

2) On-Device Processing: All context fusion, personalization, emotion inference, and planning occur **on-device**. Network connections are required only for optional features (web search, cloud service integration) with explicit user consent per interaction.

3) Differential Privacy: When users opt into model improvements, differential privacy guarantees [16] provide formal bounds on information leakage. Local gradient updates have calibrated noise added before aggregation, ensuring that even with complete server compromise, individual user data cannot be reconstructed.

Recent production deployments demonstrate feasibility: Google's Gboard achieves strong differential privacy guarantees (ρ <1.0 zero-Concentrated Differential Privacy) for federated learning [20], stronger than the ρ =2.63 used by the 2020 US Census for data release.

4) Security Hardening:

- **Secure Enclaves**: Context processing in hardware-isolated environments (ARM TrustZone, Apple Secure Enclave)
- **Encrypted Storage**: All context and memory data encrypted with user-controlled keys
- **Sandboxing**: Application access to context mediated through permission system with fine-grained control
- **Audit Logs**: Comprehensive, immutable logs of all system actions enable forensic analysis and trust building
- **Adversarial Robustness**: Input validation, anomaly detection for context poisoning attempts (detailed in Section VI-E)

V. Technical Feasibility Analysis

A. Hardware Capabilities

Modern computing hardware increasingly supports **on-device** AI:

Neural Processing Units (NPUs): Apple M3/M4 chips feature Neural Engines delivering 18-38 TOPS; Qualcomm Snapdragon 8 Gen 3 includes Hexagon NPU with 45 TOPS; Intel Meteor Lake integrates dedicated AI accelerators. These enable continuous **on-device** inference for context fusion and emotion recognition.

Memory and Storage: Modern flagship smartphones typically provide 8-12GB RAM and 256-512GB storage, sufficient for context caching and local model hosting. NVMe SSDs with 7GB/s throughput enable rapid model loading.

Battery Efficiency: Modern NPUs achieve high energy efficiency, enabling always-on AI inference with minimal battery impact. Production federated learning systems have demonstrated <5% battery overhead for continuous operation [19].

Secure Hardware: ARM TrustZone provides hardware-isolated trusted execution environments across mobile and embedded devices. Intel SGX and Apple Secure Enclave offer similar capabilities on PC and Mac platforms.

B. Software Stack Feasibility

Model Size and Latency: Distilled language models (e.g., DistilBERT [29], MobileBERT [37]) achieve 97% of BERT-base performance at 40% size, fitting in <100MB. Quantization to INT8 or INT4 further reduces size and accelerates inference. **On-device** inference achieves <50ms latency for intent extraction on current hardware.

Context Fusion: Transformer models with 6-12 layers, 512-1024 hidden dimensions can process multimodal context in <50ms on NPUs. Efficient attention mechanisms (e.g., Linformer [34], Performer [35]) reduce complexity from $O(n^2)$ to $O(n)$.

Memory Requirements: Vector databases (e.g., FAISS [27]) enable billion-scale similarity search with <1GB memory footprint. Hierarchical quantization (Product Quantization) balances retrieval accuracy and resource usage.

Federated Learning Infrastructure: TensorFlow Federated and PyTorch Mobile provide production-ready frameworks. Secure aggregation protocols enable privacy-preserving model updates with acceptable communication overhead (<10MB per round) [36].

C. OS Integration Strategies

Microkernel Extension: OS-1 components could operate as privileged services in microkernel architectures, accessing system resources while maintaining isolation.

User-Mode Supervisor: Alternatively, a user-mode supervisor with accessibility API access and inter-process communication could orchestrate applications without kernel modifications.

Developer APIs: Third-party applications would opt into context sharing via standardized APIs, enabling richer automation while preserving sandboxing and security.

Backward Compatibility: Legacy applications function normally; OS-1 features activate progressively as applications adopt context APIs.

D. Performance Targets

Based on current hardware capabilities and optimization techniques, a production OS-1 implementation could target:

- **Context Update Latency**: <50ms for real-time responsiveness
- **Voice-to-Action Latency**: <500ms for simple actions, <2s for complex multi-step workflows
- **Memory Retrieval**: <100ms for semantic search across 1M+ memory entries
- **Battery Impact**: <5% additional power consumption for continuous operation
- **Storage Footprint**: 500MB-2GB for models, context, and memory
- **Privacy Guarantee**: ρ <1.0 zCDP for federated learning with differential privacy

TABLE I
COMPARISON OF AI-ENHANCED OS APPROACHES

Capability	OS-1 (Proposed)	Apple Intel.	Win Copilot	Android AI
Unified Context	Yes	Partial	No	Limited
Proactive Automation	Yes	No	Limited	No
Affective Adaptation	Yes	No	No	No
On-Device Primary	Yes	Yes	No	Partial
Formal Privacy (DP)	Yes	No	No	Unknown
Cross-App Orchestration	Yes	Limited	No	No
Emotional Intelligence	Yes	No	No	No

E. Comparison with Existing Systems

Table I compares OS-1's proposed capabilities with current AI-enhanced operating systems.[1]

VI. Research Challenges and Open Problems

While technically feasible, OS-1 faces significant research challenges:

A. Scaling Context and Memory

Challenge: As context history grows to months or years, efficient retrieval becomes critical. Semantic embeddings help but don't solve forgetting (what to discard) or hierarchical memory organization (different abstraction levels).

Open Questions: How should memory consolidation balance specificity versus generalization? What forgetting mechanisms preserve useful patterns while discarding noise? How can memory organization adapt to evolving user patterns?

B. Reliable Cross-Application Orchestration

Challenge: Application sandboxing improves security but complicates automation. Accessibility APIs enable some actions but lack semantic understanding of application state. Fragile automation breaks with UI updates.

Open Questions: Can we develop robust, semantic application control APIs balancing security and automation? How should systems gracefully handle workflow failures? What verification mechanisms ensure action correctness before execution?

C. Intent Ambiguity and Multi-Goal Reasoning

Challenge: Natural language goals are often ambiguous or underspecified. Users may have implicit constraints or preferences difficult to articulate. Complex goals may involve trade-offs requiring value judgments.

Open Questions: How should systems detect and resolve ambiguity through minimal, well-targeted clarifying questions? When should systems make autonomous decisions versus seeking guidance? How can preference learning occur from indirect feedback?

[1]Assessments based on publicly available documentation and feature analysis as of October 2024.

D. Explainability and Trust Calibration

Challenge: Users must understand system reasoning to develop appropriate trust. Over-trust leads to over-reliance; under-trust prevents adoption. Explanations must be truthful yet comprehensible. This process requires accurately calibrating the system's confidence in its own intent recognition and planning accuracy.

Open Questions: What explanation granularities balance detail and comprehensibility? How should confidence be communicated to support calibrated trust? How can systems learn optimal autonomy levels per user?

E. Privacy Attacks and Adversarial Robustness

Challenge: Despite **on-device** processing, attacks remain possible: context poisoning (injecting false data), inference attacks (deducing information from behavior), and federated learning attacks (model poisoning, gradient analysis).

Open Questions: How can systems detect and mitigate context poisoning attempts? What additional privacy guarantees beyond differential privacy are needed? How should anomaly detection balance sensitivity and false positives? What formal verification methods can provide security assurances for autonomous systems?

F. Evaluation Methodologies

Challenge: Traditional OS benchmarks (throughput, latency) are inadequate for assessing the quality of autonomous behavior, intent recognition accuracy, and trust calibration in intelligent, agentic systems.

Open Questions: How should we define and measure "trustworthiness" and "utility" in agentic OS environments? What reproducible metrics can quantify the reduction in cognitive load? How can ethical compliance (e.g., bias mitigation) be benchmarked across diverse user contexts?

VII. Broader Impact

VIII. Conclusion

References

[1] B. A. Myers, "A brief history of human-computer interaction technology," *ACM Interact.*, vol. 5, no. 2, pp. 44–54, 1998.

[2] E. Horvitz, J. Breese, and D. Heckerman, "The principles of the windows 98 assistant," in *Proc. UAI*, 1999.

[3] S. Amershi et al., "Guidelines for human-AI interaction," in *Proc. CHI*, 2019.

[4] A. S. Tanenbaum and H. Bos, *Modern Operating Systems*, 4th ed. Pearson, 2014.

[5] R. Pike et al., "Plan 9 from Bell Labs," *Bell Labs Tech. J.*, vol. 2, no. 1, pp. 5–44, 1995.

[6] J. Widom and S. Ceri, *Active Database Systems: Triggers and Rules For Advanced Database Processing*. Morgan Kaufmann, 1996.

[7] B. Schilit, N. Adams, and R. Want, "Context-aware mobile computing," in *Proc. Mobile Computing Systems and Applications*, 1994.

[8] G. D. Abowd et al., "Context-awareness in wearable and ubiquitous computing," in *Proc. Int. Symp. Wearable Computers*, 1999.

[9] A. K. Dey and G. D. Abowd, "Understanding and using context," *Pers. Ubiquitous Comput.*, vol. 5, no. 1, pp. 4–7, 2001.

[10] C. Perera et al., "Context-aware computing for the internet of things: A survey," *IEEE Commun. Surv. Tutor.*, vol. 16, no. 1, pp. 414–454, 2014.

[11] A. K. Dey, D. Salber, and G. D. Abowd, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications," *Hum.-Comput. Interact. (HCI)*, vol. 16, no. 2, pp. 97–166, 2001.

[12] R. W. Picard, *Affective Computing*. MIT Press, 1997.
[13] S. Poria et al., "A review of affective computing: From unimodal analysis to multimodal fusion," *ACM Trans. Interact. Intell. Syst.*, vol. 7, no. 3, pp. 1–36, 2017.
[14] Z. Liu et al., "Affective computing with large language models: A review," *arXiv preprint arXiv:2403.01878*, 2024.
[15] H. B. McMahan et al., "Communication-efficient learning of deep networks from decentralized data," in *Proc. AISTATS*, 2017.
[16] C. Dwork, "Differential privacy," in *Proc. ICALP*, 2006.
[17] P. Kairouz et al., "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, no. 1–2, pp. 1–210, 2021.
[18] K. Wei et al., "Balancing privacy and utility in federated learning with differential privacy: A survey," *IEEE Trans. Dependable Secure Comput.*, 2024.
[19] M. Girgis et al., "Shuffled models for federated learning in mobile devices," in *Proc. MLSys*, 2021.
[20] P. Kairouz et B. McMahan, "Practical differentially private federated learning for Gboard," *Google AI Blog*, 2021.
[21] Apple Inc., "Introducing Apple Intelligence," Press Release, 2024.
[22] Microsoft Corporation, "Windows Copilot: Your everyday AI assistant," Product Documentation, 2024.
[23] Google LLC, "Android AI Initiatives and Generative Features," Developer Documentation, 2024.
[24] H. Mei et al., "AIOS: A Large Language Model Agent Operating System with Kernel Abstractions," *arXiv preprint arXiv:2404.09033*, 2024.
[25] A. Vaswani et al., "Attention is all you need," in *Proc. NeurIPS*, 2017.
[26] E. Tulving, "Episodic and semantic memory," in *Organization of Memory*. Academic Press, 1972, pp. 381–403.
[27] T. Johnson et al., "Billion-scale similarity search with GPUs," *IEEE Trans. Big Data*, vol. 7, no. 3, pp. 535–547, 2019.
[28] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. ICASSP*, 2013.
[29] V. Sanh et al., "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.
[30] A. S. Rao and M. P. Georgeff, "BDI agents: From theory to practice," in *Proc. Int. Conf. Multi-Agent Systems*, 1995.
[31] A. d'Avila Garcez et al., "Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning," *J. Appl. Logics*, vol. 6, no. 4, pp. 611–631, 2019.
[32] S. C. H. Hoi et al., "Online learning: A comprehensive survey," *Neurocomputing*, vol. 459, pp. 249–289, 2021.
[33] J. Hernandez et al., "BioPhone: Physiology monitoring from peripheral smartphone interactions," in *Proc. CHI*, 2014.
[34] S. Wang et al., "Linformer: Self-attention with linear complexity," *arXiv preprint arXiv:2006.04768*, 2020.
[35] K. Choromanski et al., "Rethinking attention with performe," in *Proc. NeurIPS*, 2020.
[36] K. Bonawitz et al., "Practical secure aggregation for federated learning on millions of user devices," in *Proc. ACM CCS*, 2017.
[37] Z. Sun et al., "MobileBERT: a compact task-agnostic BERT for resource-limited devices," in *Proc. AAAI*, 2020.