Project Unison: Architecting the Future of Human-Centric Multimodal AI

By Darryl Adams

Contents

Executive Summary

We stand at an inflection point in our relationship with technology. For decades, humans have adapted themselves to the constraints and idiosyncrasies of computing platforms. However, advances in artificial intelligence, multimodal interaction, and secure, personalized computing architectures now offer us an unprecedented opportunity to flip this dynamic. Project Unison represents a visionary shift toward technology that adapts to us, understanding our context, intentions, and preferences intuitively, enabling interaction in natural, human-centric ways.

Project Unison envisions a multimodal AI computing platform that fundamentally redefines human-computer interaction, breaking free from traditional paradigms like keyboards, mice, and screens. This next-generation system is designed to interpret diverse user inputs seamlessly, including spoken language, gestures, visuals, text, tactile feedback, and

contextual cues, integrating them fluidly in real-time. Crucially, the system proactively anticipates user needs rather than passively awaiting commands.

At its core, Project Unison prioritizes trust, security, and personalization. Leveraging cutting-edge confidential computing technologies, it securely manages sensitive user data, preferences, and personally identifiable information within protected Trusted Execution Environments (TEEs). By separating private user interactions from public-facing data exchanges, Project Unison maintains rigorous privacy standards without compromising the rich personalization essential for genuinely intuitive interactions.

For hardware manufacturers, Project Unison represents a transformative market opportunity. It provides an architectural blueprint for creating innovative, powerful, and energy-efficient client platforms ideally suited to multimodal computing experiences. The compact form factors and advanced computing capabilities essential for Project Unison open new pathways for OEM differentiation and market leadership.

Ultimately, Project Unison is not simply about refining existing computing tasks, it's about empowering entirely new interactions and experiences, transforming technology into a trusted collaborator that understands, anticipates, and supports us naturally and intuitively.

Vision and Strategic Importance

We are entering an era in which technology evolves beyond being merely a tool or passive appliance. Project Unison embodies a profound transformation in our fundamental relationship with computers, from one in which humans adapt to technological constraints, to one in which technology genuinely understands and adapts to humans. This shift marks a critical moment, as multimodal AI assistants capable of proactively interpreting user context, intent, and emotion move from the realm of science fiction into practical reality.

Historically, technology interactions have predominantly required users to conform to rigid interfaces: typing on keyboards, navigating through menus, clicking icons, and interpreting visual output. While effective, this mode inherently excludes those unable or unwilling to engage through traditional interfaces, particularly individuals with disabilities or those operating in dynamic environments unsuitable for conventional interactions.

Project Unison disrupts this traditional interaction model by leveraging breakthroughs in artificial intelligence, natural language understanding, computer vision, haptics, and contextual analysis. The result is a platform capable of seamlessly interpreting and synthesizing inputs from multiple modalities simultaneously, allowing technology to

comprehend human communication in all its natural complexity, spoken words, gestures, visual expressions, and even subtle contextual signals such as user location, emotional state, and behavioral patterns.

Strategically, this transition to adaptive, multimodal interactions creates a new competitive landscape for manufacturers. It opens substantial opportunities in rapidly expanding markets such as accessible computing, assistive robotics, augmented and virtual reality, automotive interfaces, smart home integration, and personalized healthcare. Manufacturers who embrace this vision early can significantly differentiate themselves by offering user-centric systems that deliver greater autonomy, dignity, and productivity across all demographics and usage contexts.

Further, by embedding privacy, personalization, and trust at the very core of its architecture, Project Unison addresses mounting global concerns around data security and ethical AI deployment. By design, users maintain control of their data within secure environments, enhancing both consumer confidence and regulatory compliance. Manufacturers adopting this approach position themselves as trusted leaders at the intersection of innovation and responsible technology.

In essence, Project Unison offers not only a technological advancement but also a philosophical shift in computing, a move toward systems that are intuitive, adaptive, inclusive, and profoundly human-centered. The strategic importance of this shift cannot be overstated; those who lead in this paradigm will define the next era of computing and user experience.

Architectural Overview

Project Unison is envisioned not as a single product but as a flexible, modular computing architecture. A foundation upon which manufacturers can build innovative multimodal solutions. At its heart, the Unison architecture is guided by several core principles that ensure adaptability, responsiveness, and secure personalization across diverse usage scenarios.

Core Architectural Principles

Multimodal Integration
The architecture seamlessly blends multiple interaction modalities—including voice, vision, text, gestures, and haptics—processing these inputs concurrently in real-time. Rather than treating these modalities as separate data streams, Unison synthesizes them

into a cohesive understanding of user intent and context, delivering proactive and contextually relevant interactions.

Context-Awareness

Central to Unison's effectiveness is the Context Engine, a dedicated component capable of interpreting and managing complex contextual information. This engine continuously assesses user context—including environmental cues, personal preferences, emotional states, and historical behaviors—to inform interactions and personalize experiences.

Privacy-by-Design

Privacy and security considerations are intrinsic to every layer of the architecture. By leveraging Confidential Computing technologies, including Trusted Execution Environments (TEEs), Unison ensures user-sensitive data and processing remain secure and confidential, while still supporting personalized, intelligent interactions.

Low-Latency Interaction

To achieve genuinely intuitive user experiences, the Unison architecture emphasizes minimal latency between user input and system response. It employs localized inference and edge computing strategies to deliver rapid and fluid interaction, critical for applications like accessibility, robotics, and augmented reality.

Modularity and Adaptability

Designed for flexibility, Unison's modular approach allows components to be customized, scaled, and adapted easily to different hardware configurations, application contexts, and user-specific requirements. This modular design encourages innovation among manufacturers, empowering them to differentiate and rapidly adapt to evolving user needs and technological advancements.

High-Level System Architecture

The Unison architecture comprises five primary layers, each responsible for distinct functional roles, yet deeply interconnected:

- ### Perception Layer

  Responsible for capturing and processing user inputs across modalities, including voice recognition, visual processing, gesture interpretation, textual inputs, and haptic signals. This layer performs initial input processing and prepares structured data streams for interpretation.

- ## Context Engine
  Functions as the intelligence core of the system. It continuously builds, refines, and leverages a dynamic model of the user's context and preferences. It uses machine learning and reasoning algorithms to anticipate user needs, adaptively personalizing interactions in real-time.

- ## Interaction Manager
  Manages the dialogue and communication flow with the user. Utilizing data from the Perception Layer and insights from the Context Engine, it coordinates responses and interactions across modalities. It supports smooth interruptions, confirmations, and seamless switching between interaction methods.

- ## Actuation Layer
  Translates interpreted user intentions into concrete actions, controlling connected devices, robotic actuators, smart appliances, digital services, or virtual environments. This layer ensures reliable and timely execution, interacting directly with hardware and external service interfaces.

- ## AI Agent Orchestration Layer
  Sitting above the Context Engine and Interaction Manager, the Agent Orchestration Layer serves as the centralized coordination fabric that manages a dynamic team of specialized AI agents. This layer orchestrates context propagation, decomposes complex user intents into manageable sub-tasks, facilitates inter-agent communication, and dynamically distributes workloads to optimize system responsiveness and scalability.

Key components of the Agent Orchestration Layer include:

- *Protocol Broker*:
  Utilizes the Model Context Protocol (MCP) to standardize secure, structured API and service access for agents. Acting like an OpenAPI manifest for AI, MCP packages metadata, authorization scopes, and credentials into JSON-RPC calls—enabling seamless, trustworthy integration of internal and external tools.

- *Agent Registry & Discovery*:
  Maintains a comprehensive catalog of “Agent Cards,” each representing an agent’s capabilities, supported modalities, and trust level. Using the Agent-to-Agent (A2A) Protocol, agents engage in peer-to-peer discovery, enabling vendor-neutral handshakes and secure task exchanges across a diverse agent ecosystem.

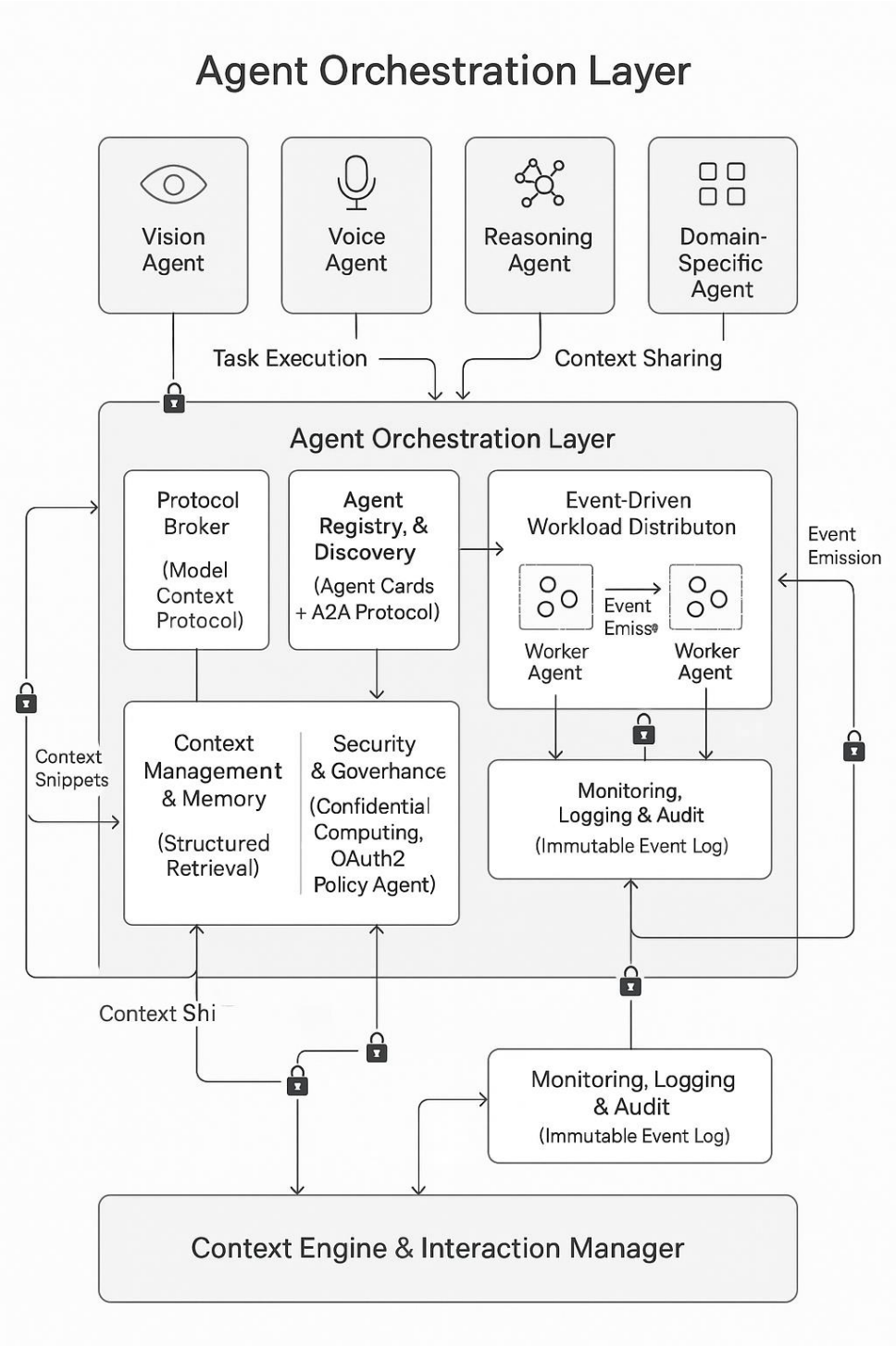
- *Task Decomposition & Hierarchical Scheduler:*
  Implements a hierarchical multi-agent system following a tree-structured orchestration model—comprising global orchestrators for strategic intent breakdown, domain coordinators for tactical delegation, and leaf nodes for task execution. This hierarchy balances specialization and scalability to efficiently manage complex workflows.
- *Event-Driven Workload Distribution:*
  Employs an event-sourced, orchestrator-worker model leveraging technologies like Apache Kafka® to decouple agent interactions. Task events are partitioned and emitted by orchestrators, with worker agent groups consuming workloads in parallel. This design supports automatic load rebalancing and fault recovery through replayable logs.
- *Context Management & Memory:*
  Extends the Context Engine's capabilities using MCP's structured retrieval augmentation. Agents persist and recall only essential context fragments—akin to human memory—avoiding full history reprocessing while ensuring efficient, low-bandwidth state sharing across the agent network.
- *Security & Governance:*
  Enforces privacy-by-design principles using confidential computing through Trusted Execution Environments (TEEs). All inter-agent and tool invocations require scoped OAuth2 tokens and capability-based credentials, guaranteeing least-privilege access with end-to-end encryption. A dedicated "Policy Agent" can be injected within the hierarchy to enforce compliance and policy validation automatically.
- *Monitoring, Logging & Audit:*
  Captures every inter-agent interaction and event in an immutable, event-sourced log, serving as the system's single source of truth. This supports real-time monitoring dashboards, forensic analysis, and regulatory audits with full traceability and replay capabilities.

How the Agent Orchestration Layer Fits into Project Unison

By integrating this Agent Orchestration Layer, Project Unison gains a robust, standards-based coordination fabric enabling plug-and-play extensibility. New agents, spanning modalities like vision, voice, reasoning, or domain-specific expertise, are seamlessly onboarded through MCP and A2A protocols, discovered via the registry, and orchestrated hierarchically to handle increasingly sophisticated tasks. This design ensures Unison

remains modular, scalable, and secure, positioning it to fully leverage the next generation of agentic AI while redefining users' relationship with computing technology.

Agent Orchestration Diagram



Multimodal Interaction Paradigm

Project Unison's core strength lies in its ability to seamlessly integrate and synchronize multiple interaction modalities, enabling truly natural and adaptive user experiences. In this section, we explore the technical foundations and design considerations for supporting voice, vision, gesture, text, haptics, and contextual signals in a unified framework.

Supported Modalities and Technologies

Input Modalities

Voice Input

- Automatic Speech Recognition (ASR) models optimized for on-device and edge deployment.
- Wake-word detection engines for low-power continuous listening.

Vision Input

- Real-time object detection and scene understanding using lightweight convolutional and transformer-based architectures.
- Face and emotion recognition with privacy-preserving anonymization options.
- Real-time sign language interpretation leveraging vision models.

Gesture Input

- Hand and body tracking via sensor fusion (camera, IMU, radar).
- Custom gesture vocabulary definition and dynamic learning for personalized commands.
- Real-time sign language interpretation leveraging gesture sensors.

Text Input

- Natural Language Understanding (NLU) pipelines for intent classification and entity recognition.
- Support for textual command parsing and semantic analysis.

Contextual Signals

- Environmental context signals (location, ambient noise level, lighting conditions).
- User context signals (historical interactions, user preferences, physiological metrics).

Braille Input

- Support for refreshable Braille displays via USB and Bluetooth for direct tactile input.
- Real-time embossed Braille recognition using camera-based OCR sensors for legacy materials.

Output Modalities

Voice Output

- Neural Text-to-Speech (TTS) engines supporting prosody, emotion, and multilingual synthesis.

Visual Output

- Augmented reality and graphical overlays for context-rich visual feedback.
- Screen-based visual displays adaptable for varying accessibility needs.

Text Output

- Rich text generation for screen, braille displays, and console logs.
- Dynamic text formatting optimized for clarity and readability.

Haptic Output

- Tactile feedback frameworks for wearable devices and integrated actuators.
- Haptic language libraries for semantic feedback and notifications.

Braille Output

- Generation of dynamic Braille output compatible with refreshable Braille displays and embossers.

Real-Time Synchronization and Fallback Strategies

- ### Sensor Fusion and Data Prioritization
  Techniques for merging asynchronous data streams into a coherent context representation, employing time-series alignment and confidence-based weighting.
- ### Concurrency Management
  Approaches to parallel processing of modality-specific pipelines while avoiding resource contention and ensuring predictable latency bounds.
- ### Fallback Pathways
  Design patterns for gracefully handling unavailable or degraded modalities, such as switching from gesture recognition to voice or text commands without user-visible disruption.

Interaction Scenarios and Use Cases

- ### Accessible Navigation
  Combining vision-based obstacle detection, voice prompts, and haptic alerts for real-time guidance in unfamiliar environments.
- ### Hands-Free Workflows
  Enabling professionals to interact via voice and gesture while maintaining focus on physical tasks (e.g., surgeons, mechanics).
- ### Rich Screenless Conversational Experiences
  Delivering contextually rich dialogues through voice, haptics, and audio-only feedback for blind and visually impaired users, enabling seamless information retrieval and system control without any visual interface.
- ### Immersive VR/AR Interfaces
  Integrating speech, gaze tracking, and contextual understanding for fully immersive, intuitive virtual environments.

Perception Layer Deep Dive

The Perception Layer serves as the front line of Project Unison's multimodal architecture, responsible for capturing, preprocessing, and translating raw sensor data into structured inputs for downstream components. This deep dive examines the design and implementation strategies for voice and vision pipelines.

Voice Pipeline

Audio Capture & Preprocessing

- High-fidelity microphone arrays with beamforming for noise suppression and directional focus.
- Preprocessing steps: echo cancellation, normalization, and dynamic range compression.

Automatic Speech Recognition (ASR)

- On-device ASR models: optimized RNN-T or transformer architectures for low-latency inference.
- Adaptive language modeling: domain-specific vocabulary injection and continuous model fine-tuning.

Wake-Word Detection

- Lightweight neural networks trained for specific keyword triggers.
- Energy-efficient execution in microcontroller or NPU for always-on listening.

Noise Robustness & Acoustic Adaptation

- Multi-band noise gating and beamforming assisted by directional arrays.
- Acoustic environment adaptation using on-the-fly calibration and user-specific profiles.

Neural Text-to-Speech (TTS)

- Tacotron-LSTM or FastSpeech backends with neural vocoders for natural prosody.
- Emotion and style control via learned embedding vectors for expressive output.

Vision Pipeline

Sensor Types & Fusion

- RGB cameras, depth sensors (RGB-D), and infrared arrays for robust capture in varied conditions.
- Sensor fusion frameworks to synchronize frame alignment and resolve occlusions.

Object Detection & Classification

- Lightweight CNNs (e.g., MobileNetV3) or transformer-based models (e.g., DETR-lite) for real-time inference.

- Quantization-aware training for efficient edge deployment.

Scene Understanding & Semantic Segmentation

- Encoder–decoder architectures (e.g., DeepLabv3+) for pixel-level understanding.
- Hierarchical segmentation pipelines to partition semantic regions and objects.

Face & Emotion Recognition

- Privacy-preserving face embedding generation with on-device feature extraction.
- Emotion classification via lightweight MLPs on top of embeddings, with optional anonymization.

Sign Language Interpretation

- Combined hand keypoint extraction (e.g., MediaPipe) and spatio-temporal sequence modeling (e.g., LSTM/Transformer).
- Continuous recognition with context-aware smoothing to handle co-articulation.

Context Engine Deep Dive

The Context Engine is the cognitive core of Project Unison, responsible for building and maintaining a dynamic model of user context, preferences, and environmental state. It provides the foundation for personalized, proactive interactions by analyzing input data, inferring intent, and delivering contextual insights to downstream components and agents.

Architectural Overview

Role and Responsibilities

- Aggregates and normalizes data from Perception Layer inputs, agent orchestration events, and historical interaction logs.
- Maintains a multi-layered context model encompassing short-term session data, mid-term task state, and long-term user profile and preferences.

Core Components

- Context Aggregator: Ingests raw and preprocessed data streams, applies feature extraction, and forwards structured context events.

- Context Model Store: A combination of time-series database, vector store, and graph database to represent temporal, semantic, and relational aspects of context.
- Inference & Reasoning Engine: Executes predictive models and rule-based reasoning to infer user intent, next-best-action suggestions, and anomaly detection.

Context Representation

Temporal Layers

- Short-Term Context: Session-bound state (e.g., current task variables, modality status, immediate environment).
- Mid-Term Context: Task workflows and multi-step interactions (e.g., ongoing navigation, open document editing).
- Long-Term Context: Persistent user profile, preferences, interaction history, and learned behaviors.

Data Structures

- Knowledge Graph: Nodes represent entities (users, devices, tasks, objects), edges encode relationships and probabilities.
- Vector Embeddings: Context snapshots encoded as high-dimensional vectors for similarity search and retrieval.
- Time-Series Events: Annotated event logs capturing chronological context changes and user actions.

Inference and Prediction

- Machine Learning Models
    - Intent classification and next-action prediction using transformer-based architectures trained on composite context features.
    - Anomaly and outlier detection for unusual user behaviors or environmental conditions.

Rule-Based Reasoning

- Customizable policy rules for domain-specific logic (e.g., accessibility constraints, security policies).
- Hybrid reasoning pipelines combining ML outputs and symbolic rules for deterministic scenarios.

Context Memory Management

Fragmentation & Pruning

- Store only essential context fragments, with configurable retention policies for short-, mid-, and long-term data.
- Garbage collection and summarization routines to prune stale context entries.

Efficient Retrieval

- Indexing by time, semantic tags, and embedding similarity to enable fast context lookups.
- Context subscription API for agents to register interest in specific context changes or events.

Privacy and Security Integration

Confidential Computing

- Execution of sensitive context processing tasks within Trusted Execution Environments (TEEs).
- Encryption-at-rest and in-transit for context data stores and communication channels.

Access Control

- Scoped OAuth2 tokens and capability-based credentials enforce least-privilege context queries.
- Audit logs capturing context access and inference decisions for compliance and traceability.

Performance and Scalability

Distributed Architecture

- Sharded context stores and replicated inference services for high availability.
- Load balancing and horizontal scaling strategies to support thousands of concurrent users.

Caching & Pre-Fetching

- In-memory caching of hot context shards and precomputed inference results.

- Predictive pre-fetching based on user patterns to minimize latency.

API & Integration Interfaces

Context Query API

- Synchronous and asynchronous endpoints for retrieving context snapshots, embedding vectors, and graph queries.

Event Subscription & Webhooks

- Publish–subscribe interface for agents to consume context change events in real-time.
- Webhook support for external systems to receive context updates and inference notifications.

Interaction Manager Deep Dive

The Interaction Manager orchestrates dialogue flow, handles multimodal user interactions, and coordinates responses across devices and output channels. This deep dive covers the Interaction Manager's architecture, state management, and dialogue strategies.

Functional Responsibilities

- Dialogue Coordination: Manages turn-taking, modality switching, and interruption handling.
- Response Generation: Selects and formats responses based on context, user preferences, and modality availability.
- Session Management: Tracks session state, user goals, and ongoing tasks across interactions.

Architectural Components

Dialogue State Tracker

- Maintains the current conversation state, including user intents, slots, and dialogue history.
- Supports commit and rollback for transactional dialogue operations.

Policy Manager

- Implements dialogue policies for selecting system actions using a combination of ML-driven and rule-based approaches.
- Engages reinforcement learning components to optimize interaction strategies over time.

Response Planner

- Constructs response plans, determining which modalities to utilize (voice, text, haptic, visual).
- Integrates context from the Context Engine to personalize responses.

Modality Dispatcher

- Routes formatted responses to the appropriate output pipelines.
- Manages fallback logic if a modality is unavailable or degraded.

Interrupt & Recovery Handler

- Detects user interruptions and gracefully transitions the dialogue state.
- Provides repair strategies, confirmations, and clarifications to maintain conversational coherence.

Dialogue Management Strategies

Hierarchical Dialogue Management

- Top-level manager for high-level intent orchestration, with sub-managers for domain-specific dialogues.
- Allows isolation of complex sub-dialogues (e.g., booking, navigation) in separate modules.

Mixed-Initiative Interaction

- Balances user-led and system-led dialogue, enabling the system to ask clarifying questions or suggest actions proactively.

Adaptive Turn-Taking

- Dynamically adjusts turn-taking thresholds based on modality confidence levels and user behavior patterns.

State and Context Integration

Context Synchronization

- Continuously syncs dialogue state with the Context Engine to access the latest user context and history.
- Subscribes to relevant context change events via webhooks or push notifications.

Persistent Session Storage

- rsists session data across device restarts and user logins, supporting seamless multi-device handover.

Scalability and Performance

Microservices Architecture

- Each component (State Tracker, Policy Manager, Response Planner, etc.) runs as a scalable service with independent deployment.
- Supports auto-scaling and health checks for resilience.

Low-Latency Pathways

- Critical dialogue paths optimized with in-memory caches and direct context lookups.
- Hot paths executed within local edge nodes to minimize round-trip time.

Monitoring, Analytics, and Continuous Improvement

Interaction Logging

- Comprehensive logging of dialogue turns, policy decisions, and user feedback.
- Data pipeline for offline analytics and model retraining.

A/B Testing and Experimentation

- ramework for testing alternative dialogue strategies and measuring user satisfaction metrics.

User Feedback Loop

- Mechanisms for collecting explicit and implicit user feedback on interaction quality.
- Integrates feedback into model retraining and policy updates.

Actuation Layer Deep Dive

The Actuation Layer is responsible for translating interpreted user intents and system decisions into tangible actions in the physical or digital world. This layer interfaces with hardware devices, robotic systems, IoT endpoints, and external services to effectuate tasks reliably and securely.

Functional Responsibilities

- Command Translation: Maps high-level intent representations into low-level device commands or API calls.
- Device Driver Abstraction: Provides a unified interface to disparate hardware controllers and communication protocols.
- Safety and Compliance: Ensures all commands comply with safety rules, regulatory standards, and user-defined policies.
- Feedback and Confirmation: Captures device state feedback and confirms action completion back to the Interaction Manager.

Architectural Components

Command Router

- Routes command messages to appropriate device controllers or service endpoints.
- Supports prioritization and scheduling of concurrent commands.

Driver & Protocol Adapters

- Abstracts communication with devices via protocols such as MQTT, WebSocket, REST, gRPC, Modbus, CAN bus, BLE, and proprietary SDKs.
- Enables hot-swappable driver modules for extensibility.

Safety & Policy Engine

- Validates commands against safety constraints and policy rules (e.g., speed limits, force thresholds).
- Integrates a Policy Agent that can veto or adjust commands based on compliance requirements.

State Monitor & Sensor Feedback:

- Aggregates telemetry and sensor readings from actuators to report status and detect anomalies.
- Provides real-time feedback loops to adapt ongoing actions (e.g., braking, motion planning adjustments).

Transactional Execution & Rollback:

- Supports atomic command sequences with the ability to rollback or compensate in case of failures.

Hardware Integration

Robotics Interface

- Standardized APIs for robot kinematics and motion control (e.g., ROS2 nodes, OPC UA).
- Support for real-time control loops and safety-rated motion profiles.

IoT & Smart Devices

- Lightweight edge gateways for bridging between local devices and cloud services.
- Secure onboarding and provisioning workflows for new devices using mutual TLS and certificate-based authentication.

Assistive Devices

- Integration frameworks for specialized peripherals (e.g., refreshable braille displays, haptic wearables).
- Real-time actuation patterns optimized for accessibility feedback (e.g., vibration patterns for notifications).

Performance, Reliability, and Scalability

Real-Time Constraints

- Prioritized command queues with hard and soft real-time guarantees.
- Latency budgets defined per device class to meet responsiveness requirements.

Fault Tolerance

- Heartbeat monitoring and failover mechanisms for critical actuators.

- Circuit breakers and retry strategies for transient communication errors.

Scalability

- Modular microservices for driver adapters and command routing.
- Horizontal scaling of the Command Router and safety engine to support large fleets of devices.

Security and Governance

Secure Channel Enforcement

- End-to-end encryption for all command and telemetry channels using TLS or DTLS.
- Device identity management with hardware-backed keys (e.g., TPM, Secure Element).
- ### Audit and Traceability
- Immutable logs of all actuation commands and device responses.
- Integration with system-wide monitoring and logging for comprehensive audit trails.

Cross-Sectional Considerations

This section highlights overarching themes and design guidelines that traverse all architectural layers of Project Unison, ensuring consistency in trust, privacy, security, hardware requirements, and seamless agent orchestration integration.

Trust and Security

Confidential Computing & TEEs

- All sensitive processing (context inference, policy evaluation, orchestration decisions) executes within Trusted Execution Environments.
- TEEs isolate private user data and critical orchestration logic from the public-facing components.

Policy Agent & Governance

- A dedicated Policy Agent, injected into the Agent Orchestration Layer, enforces enterprise and regulatory rules at each decision point.

- Fine-grained capability-based credentials (OAuth2 scopes) ensure least-privilege access across agents and services.

Secure Communication

- End-to-end encryption for all inter-layer and inter-agent channels using TLS/DTLS.
- The Protocol Broker in the orchestration layer mediates API access with signed JSON-RPC tokens (MCP manifests).

Privacy-by-Design

Data Minimization & Context Pruning

- Orchestration events and context fragments shared between agents are minimal, with retention policies enforced by the Context Engine's memory management.
- Structured retrieval ensures only necessary context is fetched for each task, reducing exposure.

Anonymization & Access Controls

- User-identifiable information is anonymized in logs; only pseudonymous metadata propagates through agent workflows.
- Scoped tokens managed by the Protocol Broker restrict context queries and tool invocations based on user consent.

Hardware Reference Guidelines

Compute Fabric

- Support for heterogeneous on-device NPUs, GPUs, and CPUs to accelerate ASR, vision, and inference models.
- Minimum recommended specifications: quad-core CPU, 4 TOPS NPU, 2 TFLOPS GPU for real-time performance.

Secure Element & TPM

- Hardware security modules or TPM chips facilitate key storage for TLS, OAuth2 credentials, and TEE attestation.
- Firmware support for secure boot and hardware-backed identity.

Connectivity & I/O

- Low-latency interfaces (PCIe, UCSI, SPI) for camera arrays, microphone arrays, haptic actuators, and Braille displays.
- High-throughput network interfaces (Wi-Fi 6E, Ethernet) for edge-cloud orchestration and real-time agent coordination.

Agent Orchestration Integration

Protocol Broker (MCP)

- Centralized manifest-based broker defines service endpoints, scopes, and credentials, enabling secure API access across agents and hardware modules.

Agent Registry & Discovery

- Hardware modules and service endpoints register as "Agent Cards," advertising capabilities (e.g., camera, microphone, actuator), enabling dynamic discovery and orchestration.

Hierarchical Scheduler & Workload Distribution

- Topology-aware scheduling maps agents to hardware resources, optimizing for latency (e.g., local NPU for vision tasks, edge server for heavy inference).

Context Memory & Retrieval

- Orchestrator components leverage structured retrieval to pull context embeddings and graph fragments, minimizing data transfer and ensuring privacy.

Compliance and Standards

Model Context Protocol (MCP)

- Standardizes secure, structured interactions using OpenAPI-like manifests in JSON-RPC calls for both internal and third-party agents.

Agent-to-Agent (A2A) Protocol

- Facilitates vendor-neutral discovery and secure, peer-to-peer task handoffs, promoting interoperability in heterogeneous ecosystems.

Regulatory Adherence

- GDPR, HIPAA, and other relevant frameworks guide data handling policies.

- Audit logs from the Monitoring, Logging & Audit component support compliance reporting.

Performance Monitoring & Continuous Governance

Event Logging & Telemetry

- Immutable, event-sourced logs capture orchestration events, context updates, and actuation outcomes for real-time dashboards.

Health Checks & Metrics

- eartbeat signals from agents and hardware modules ensure liveness; metrics (latency, throughput, error rates) feed into auto-scaling policies.

Feedback Loops

- Insights from user interactions and system telemetry inform model retraining, policy updates, and orchestration optimizations, sustaining a cycle of continuous improvement.

Reference Hardware Platforms

This section provides guideline specifications and reference designs for hardware implementations of Project Unison. These recommendations balance performance, energy efficiency, security, and extensibility, enabling OEMs to build client platforms optimized for multimodal AI workloads and secure, personalized interactions.

Compute Fabric

CPU

- Multi-core (minimum quad-core) ARM or x86_64 processors supporting out-of-order execution and SIMD extensions (e.g., AVX2/NEON).
- Base frequency ≥ 2.0 GHz, boost up to ≥ 3.5 GHz for latency-sensitive tasks.

GPU

- Integrated or discrete GPUs capable of at least 2 TFLOPS FP16 compute for accelerated vision and graphics workloads.
- Support for common AI frameworks (CUDA, OpenCL, Vulkan AI).

NPU / AI Accelerator

- Dedicated neural processing unit delivering ≥ 10 TOPS for on-device inference.
- Programmable for CNN, transformer, and sequence models; support for INT8/FP16 quantized execution.

Memory & Storage

System Memory

- Minimum 16 GB LPDDR5 or DDR4 RAM; recommend 32 GB for complex context modeling and parallel pipelines.

Non-Volatile Storage

- NVMe SSD or eMMC with capacity ≥ 512 GB for model caches, context stores, and local data.
- Optional SD/UFS expansion slot for additional offline storage.

Trusted Execution & Security Elements

Trusted Platform Module (TPM) / SE

- TPM 2.0 or Secure Element for secure key storage, device identity, and cryptographic operations.
- Integration with secure boot and firmware attestation.

TEE Support

- Hardware-level virtualization features (e.g., Intel SGX, ARM TrustZone) enabling isolation of confidential workloads.

Sensor & I/O Interfaces

Camera Interfaces

- MIPI CSI-2 lanes (minimum 2 lanes) supporting up to 1080p60, plus support for RGB-D or ToF sensors.

Microphone / Audio

- I^2S / PDM microphone array support with integrated ADCs; optional beamforming DSP.

Haptic & Actuator Ports

- PWM / I^2C / SPI interfaces for tactile actuators, Braille display controllers, and vibration motors.

Braille Display Connection

- USB HID and Bluetooth LE GATT support for refreshable Braille displays.

General-Purpose I/O

- GPIO headers for custom sensor integration and peripheral expansions.

Connectivity & Networking

Wireless

- Wi-Fi 6E (802.11ax) for high-throughput local connectivity.
- Bluetooth 5.2 for low-latency device pairing and audio transmission.

Wired

- Gigabit Ethernet (or 2.5 GbE) for robust edge-cloud synchronization and orchestration messaging.

Cellular (Optional)

- 5G Cat 20 or LTE Advanced Pro modules with fallback; integrated SIM support.

Power & Thermal Design

Power Input

- 12–20 V DC input with power budgeting to support peak compute (≤ 65 W).
- Support for UPS or battery backup in mobile scenarios.

Thermal Management

- Passive heatsinks or active cooling (fan) options; temperature sensors for thermal throttling.

Form Factor & Mechanical Considerations

Compact Modular Design

- Mini-PC (e.g., 4 × 4-inch SBC) or SFF chassis with mounting brackets for displays or VESA mounts.

- Stackable modules for sensor, compute, and power boards.

Environmental Ratings

- Operating temperature range: 0 °C to 50 °C; optional industrial-grade variants (-20 °C to 70 °C).
- Ingress Protection (IP) ratings if deployed in harsh environments.

//End of Document