eXp-AIOS

An Architecture for Interoperable AI, Heterogeneous Computing, and the Quantum Transition

Architecture and Systems Analysis

Brian B.J. Hall

OneKindScience

Abstract

Artificial intelligence is entering an infrastructure transition.

The challenge is no longer simply how to build larger models. Modern AI systems increasingly depend on heterogeneous collections of models, processors, memory systems, accelerators, cloud services, edge devices, and specialized computational resources. These systems must exchange data, coordinate execution, and increasingly determine where a particular computational task should run.

At the same time, conventional computing is encountering increasingly significant constraints associated with memory movement, interconnect bandwidth, power consumption, thermal management, and the complexity of distributed accelerator systems. Quantum computing introduces another computational paradigm, but current quantum processors remain specialized, noisy, and unsuitable as general replacements for classical computing.

This paper proposes eXp-AIOS — the Expandable Artificial Intelligence Operating System — as a systems architecture designed to address these two transitions simultaneously.

The architecture proposes five cooperating components:

- NAISCII — a proposed intermediate communication and instruction layer for AI-to-AI and AI-to-hardware interoperability.

- NASOF — a proposed structured object representation for standardized exchange of heterogeneous AI data.

- MAPLE 1.0 — a proposed development, orchestration, and execution environment.
- Maple-A — a proposed hardware-abstraction and workload-routing architecture for heterogeneous processors.
- OSCQAS — a proposed security architecture incorporating established cryptographic and post-quantum security standards.

Together, these components form an abstraction layer between applications, artificial intelligence models, and heterogeneous computational hardware.

The objective is not to replace existing operating systems, AI frameworks, GPUs, TPUs, or quantum processors. Instead, eXp-AIOS proposes a layer capable of coordinating them.

The architecture therefore represents a research and engineering proposition rather than a claim of completed or benchmarked technology. Its principal hypotheses — including whether a common intermediate representation, structured object model, and hardware-aware routing architecture can reduce integration overhead and improve computational efficiency — require implementation, benchmarking, security review, and independent validation.

The long-term objective is a computing environment in which the same application can move across classical CPUs, GPUs, TPUs, NPUs, edge systems, and quantum processors without requiring the application itself to be fundamentally rewritten for each computational substrate.

---

1. Introduction: The Computing Architecture Is Changing

The computing industry is undergoing a transition from relatively homogeneous computing environments toward increasingly heterogeneous systems.

A contemporary AI workload may involve:

human input → application → AI model → accelerator → memory → network → another AI model → specialized processor → cloud service → edge device

Each layer can be implemented using different technologies, vendors, programming frameworks, data representations, and execution models.

This creates a fundamental systems problem.

The difficulty is no longer merely obtaining computational power. It is increasingly the ability to coordinate computational power efficiently.

Three transitions are occurring simultaneously:

1. AI models are becoming distributed and specialized.
2. Computing hardware is becoming increasingly heterogeneous.
3. Quantum processors are emerging as specialized computational resources alongside classical systems.

eXp-AIOS is proposed as an architectural response to this convergence.

Its central premise is straightforward:

AI systems should be able to communicate through a common computational abstraction, while the underlying infrastructure determines where each operation should execute.

This separates the intelligence of an application from the particular processor executing it.

---

2. The Infrastructure Problem

2.1 The Memory and Data-Movement Challenge

The conventional computing model separates processing from memory. As AI models grow, moving parameters, activations, intermediate representations, and data between memory and computational units can become a significant component of overall system cost.

This problem is commonly described as the memory wall.

The challenge becomes especially important when AI workloads are distributed across multiple accelerators. A computation that appears mathematically simple can become expensive when its intermediate results must repeatedly cross memory, processor, and network boundaries.

The result is an increasingly important distinction:

Computational efficiency is not determined solely by how many operations a processor can perform. It is also determined by how efficiently information can move between the operations.

This creates four related infrastructure pressures:

Infrastructure Pressure	Fundamental Problem	Consequence
Memory movement	Data must move between storage and compute	Latency and energy consumption
Interconnect bandwidth	Accelerators must exchange intermediate results	Synchronization overhead
Power and thermal limits	Computation and data movement consume energy	Infrastructure and cooling constraints
Heterogeneous hardware	Different processors excel at different workloads	Increasing orchestration complexity

These are established systems-engineering problems. eXp-AIOS does not claim to eliminate the underlying physics.

Instead, it proposes addressing the software and architectural layer surrounding those physical constraints.

---

3. The Second Bottleneck: AI Software Fragmentation

The hardware problem is only half of the challenge.

The second problem is interoperability.

Modern AI systems are commonly constructed from specialized models and frameworks. A computer-vision model, language model, forecasting system, optimization engine, robotics controller, and scientific simulation may all use different internal representations and execution environments.

Existing technologies already address portions of this problem.

For example, ONNX addresses model interoperability, while compiler infrastructures such as MLIR address intermediate representations and transformations. Apache Arrow provides a language-independent in-memory data representation designed for efficient data exchange and zero-copy access.

These systems demonstrate an important principle:

Standardized representations can reduce the cost of moving information between otherwise independent computational systems.

The proposed contribution of eXp-AIOS is therefore not simply another data format or another API.

Its objective is broader:

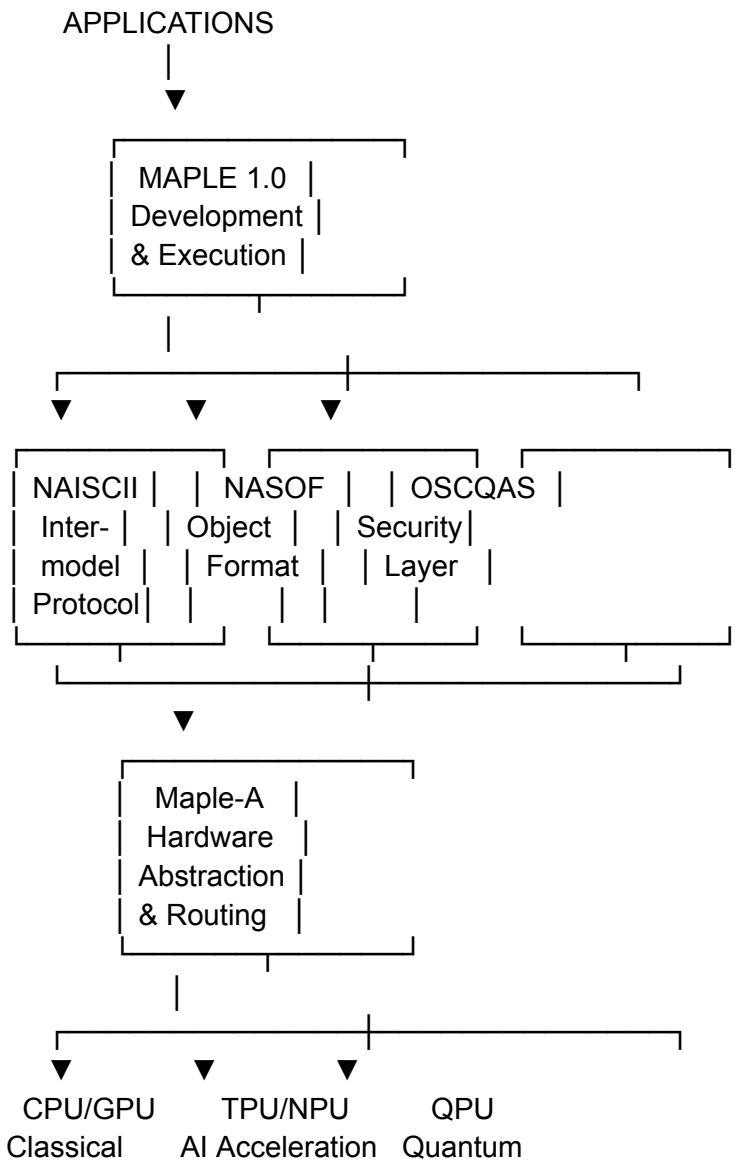
to establish a common architectural layer connecting AI models, data representations, orchestration, hardware selection, security, and future quantum execution.

---

4. The eXp-AIOS Architecture

eXp-AIOS — the Expandable Artificial Intelligence Operating System — is proposed as an orchestration and abstraction architecture positioned between applications and heterogeneous computational infrastructure.

It can be represented conceptually as:



The architecture contains five primary elements.

---

5. NAISCII: The Inter-AI Communication Layer

NAISCII — Nonagon Artificial Intelligence Standardized Code for Information Interchange — is proposed as an intermediate representation for communication among heterogeneous AI systems.

The underlying concept is analogous to an intermediate language in a compiler.

Applications do not need to know the precise architecture of every processor on which their workload may eventually execute.

Likewise, an AI model should not necessarily need to understand the internal implementation of every other AI model with which it interacts.

NAISCII would provide a standardized representation for communicating:

- model inputs;
- intermediate representations;
- feature vectors;
- execution intent;
- metadata;
- task requirements;
- security attributes;
- hardware requirements;
- model-to-model messages; and
- execution results.

The architectural distinction is important.

NAISCII is not proposed as a replacement for ONNX, gRPC, Protocol Buffers, MLIR, or other established technologies.

Rather, its research question is whether a dedicated AI-interoperability layer can provide functionality that currently requires combinations of model formats, APIs, middleware, and application-specific translation.

That question must be tested experimentally.

Core Research Hypothesis

A standardized intermediate representation specifically designed for AI-to-AI interaction may reduce integration complexity and unnecessary translation when multiple heterogeneous models participate in a single computational workflow.

The proposed benefit is therefore a hypothesis, not an established performance result.

---

6. NASOF: A Structured AI Object Model

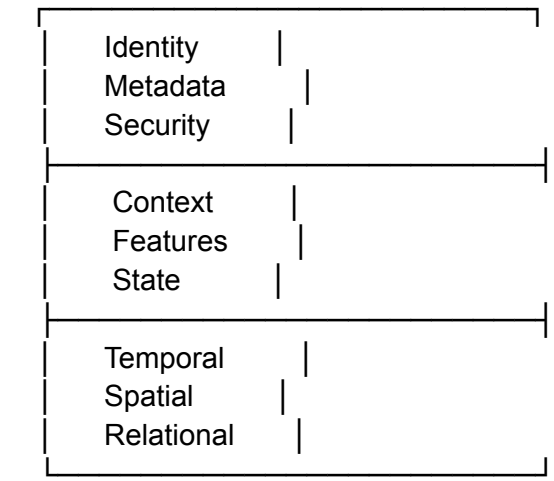
NASOF — Nonagon Artificial Intelligence Standardized Object Format — is proposed as the structured object representation operating alongside NAISCII.

The defining concept is a 9 × 9 × 9 structural representation.

The purpose of the geometry is not to claim that all information is inherently cubic.

Instead, the 9 × 9 × 9 structure is proposed as a standardized computational container capable of representing relationships among different dimensions of an AI object.

A NASOF object could conceptually contain:



The proposed representation could be applied to:

- language;
- sensor data;
- images;
- telemetry;
- time-series information;
- biomedical data;
- industrial systems;
- robotics;
- financial information; and
- AI intermediate states.

However, this is an important area for validation.

Existing formats such as Apache Arrow already provide sophisticated, language-independent memory layouts, including nested structures, zero-copy data exchange, and SIMD-friendly representations.

Therefore, the central research question for NASOF is not whether a standardized structure is useful.

That has already been demonstrated by existing systems.

The question is:

Does the specific NASOF architecture provide an additional advantage for AI execution, heterogeneous accelerator routing, or memory locality that existing representations do not provide?

That advantage must be demonstrated through controlled benchmarks.

---

7. MAPLE 1.0: The AI Execution Environment

MAPLE 1.0 — Multi-modal Adaptive Processing Learning Engine — is proposed as the developer and execution environment connecting applications to the underlying eXp-AIOS architecture.

MAPLE provides the operational context in which the other components interact.

Its responsibilities include:

1. Input ingestion.
2. Model registration.
3. Data normalization.
4. NASOF object creation.
5. NAISCII translation.
6. Hardware capability discovery.
7. Workload decomposition.
8. Execution scheduling.
9. Security policy enforcement.
10. Result reconstruction.

The architectural objective is to separate what the application wants to accomplish from where the computation ultimately executes.

For example, an application might request:

Analyze this data stream, identify anomalies, compare the results with historical patterns, and optimize the resulting response.

MAPLE would determine which computational components are required.

Maple-A would then determine where those components should execute.

This creates a layered execution model:

Application intent → AI workflow → intermediate representation → hardware selection → execution

---

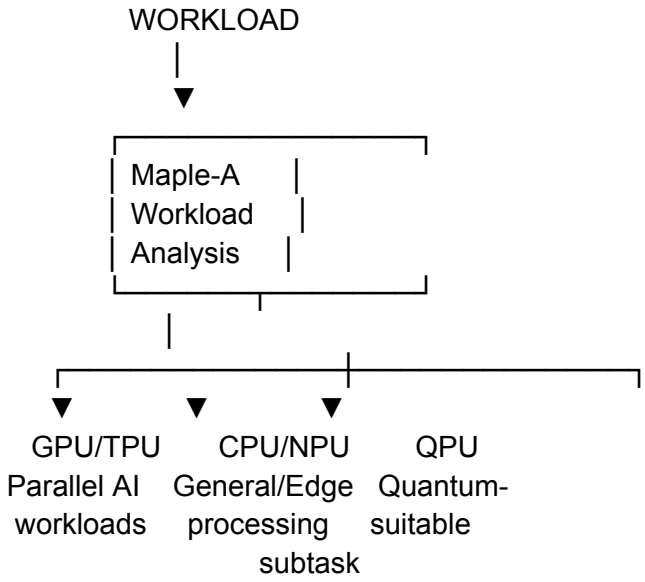
8. Maple-A: Hardware Abstraction and Intelligent Routing

Maple-A is proposed as the hardware abstraction and workload-routing layer of eXp-AIOS.

The initial architecture should be understood as a software/firmware architecture, not as a claim that a completed custom processor currently exists.

Its purpose is to determine which computational resource is best suited to a given task.

A simplified decision model is:



A mature implementation could consider:

- latency;
- throughput;
- memory availability;
- power consumption;

- accelerator utilization;
- data locality;
- network cost;
- security requirements;
- quantum availability;
- queue depth; and
- expected execution time.

This transforms hardware selection from a static configuration decision into a runtime systems problem.

The longer-term Maple-A vision could include specialized silicon.

However, such silicon should be treated as a separate engineering program involving:

- architecture definition;
- instruction-set design;
- RTL development;
- simulation;
- FPGA prototyping;
- fabrication;
- packaging;
- thermal engineering;
- driver development; and
- independent benchmarking.

The immediate research opportunity is therefore to implement Maple-A as a software-defined hardware abstraction layer before committing to custom silicon.

---

9. OSCQAS: Security for Distributed AI

OSCQAS — Omni-Secure Cypher Quantum Authentication System — is proposed as the security architecture operating across the eXp-AIOS communication layer.

Its purpose is to establish:

- identity;
- authentication;
- authorization;
- message integrity;
- encryption;
- workload isolation;
- auditability; and
- security-policy enforcement.

The architecture should not claim that OSCQAS by itself creates HIPAA or GDPR compliance.

Compliance is an organizational property involving technology, governance, policies, contracts, auditing, personnel, and data-handling procedures.

Similarly, the phrase post-quantum cryptography should refer to established cryptographic standards rather than implying that quantum hardware is required.

NIST finalized FIPS 203, FIPS 204, and FIPS 205 for post-quantum cryptography, covering ML-KEM, ML-DSA, and SLH-DSA.

Accordingly, the proposed OSCQAS architecture should be designed to incorporate vetted cryptographic standards, while retaining an extensible security-policy layer for future algorithms.

This gives OSCQAS a much stronger architectural position:

OSCQAS is not a new cryptographic universe. It is the security-control architecture through which established and future cryptographic mechanisms can be applied consistently across AI communications and heterogeneous execution environments.

---

10. The Integrated eXp-AIOS Execution Model

The five components become meaningful when considered as one system.

A complete execution cycle can be represented as:

Step 1 — Understand

MAPLE receives human, machine, sensor, application, or model input.

Step 2 — Structure

The input is represented using NASOF-compatible structures.

Step 3 — Communicate

NAISCII establishes the machine-readable representation of the requested operation and associated model or execution metadata.

Step 4 — Secure

OSCQAS applies identity, authorization, integrity, encryption, and policy controls.

Step 5 — Route

Maple-A evaluates the workload and available computational resources.

Step 6 — Execute

The workload is distributed among CPUs, GPUs, TPUs, NPUs, edge devices, cloud resources, or quantum processors where appropriate.

Step 7 — Recombine

Results are returned through the same architectural layers.

Step 8 — Learn

Execution metrics become available to the orchestration layer for future scheduling decisions.

The result is an architectural feedback loop:

Input → Representation → Communication → Security → Routing → Execution → Result → Optimization

That loop is the heart of eXp-AIOS.

---

11. The Quantum Transition

Quantum computing should not be presented as the destination of eXp-AIOS.

It is better understood as one additional computational substrate.

Current quantum systems remain constrained by noise, limited coherence, gate errors, hardware availability, and the difficulty of transferring classical information into and out of quantum systems.

Consequently, the near-term objective is not to replace classical AI with quantum AI.

It is to determine whether a particular portion of a larger computational workload is appropriate for quantum execution.

This leads to the central quantum proposition of eXp-AIOS:

Quantum computing becomes a callable computational resource rather than a separate application architecture.

---

12. The Classical-to-Quantum Handoff

The proposed eXp-AIOS quantum pipeline consists of five stages.

Phase 1 — Data Ingestion and Encoding

MAPLE receives the original workload and converts relevant information into the internal representation required by the execution pipeline.

Phase 2 — Workload Decomposition

Maple-A examines the workload and separates:

- classical computation;
- accelerator computation;
- optimization problems;
- sampling problems;
- simulation problems; and
- potentially quantum-suitable subroutines.

The default remains classical execution.

Only appropriate subproblems are considered for QPU execution.

Phase 3 — Quantum Circuit Translation

Where appropriate, classical information is converted into quantum circuit representations.

One possible technique is amplitude encoding, in which N numerical values can theoretically be represented using approximately $\log_2 N$ qubits.

However, this does not mean that arbitrary classical data can automatically be loaded into a quantum computer at logarithmic cost.

State preparation itself can be computationally expensive.

Therefore, the architecture treats encoding efficiency as an engineering problem rather than assuming a guaranteed quantum advantage.

Phase 4 — Hybrid Execution

Classical and quantum processors operate as cooperating computational resources.

For example:

Classical preprocessing → QPU subroutine → classical optimization → QPU iteration → classical synthesis

The architecture must also support immediate classical fallback when the quantum execution environment does not provide sufficient performance, fidelity, or availability.

Phase 5 — Result Synthesis

Quantum measurements are converted into classical results and returned through the eXp-AIOS representation layer.

The application therefore does not need to be rewritten around the physical characteristics of the QPU.

---

13. Why This Architecture Matters

The significance of eXp-AIOS is therefore not any single component.

It is the relationship among the components.

Existing technologies solve portions of the problem:

Problem	Existing Technology	eXp-AIOS Research Question
Model interoperability	ONNX, MLIR	Can NAISCII provide useful AI-state interoperability beyond existing model exchange?
Data exchange	Apache Arrow and related formats	Can NASOF provide an additional execution advantage for heterogeneous AI objects?
Workflow orchestration	Ray, Kubernetes, Kubeflow and others	Can MAPLE provide unified AI/hardware/quantum orchestration?
Hardware acceleration	GPUs, TPUs, NPUs	Can Maple-A optimize heterogeneous resource selection dynamically?
Quantum execution	Quantum runtimes and cloud platforms	Can eXp-AIOS provide a hardware-neutral quantum handoff layer?

Security	TLS, PKI, NIST PQC standards	Can OSCQAS unify security policy across AI execution boundaries?

This distinction is critical.

eXp-AIOS does not need to replace the existing ecosystem to be valuable.

It needs to demonstrate that a higher-level architecture can coordinate that ecosystem more effectively.

---

14. Sector Applications

The architecture is intentionally domain-independent.

Its potential applications arise anywhere multiple AI systems, data sources, and computational resources must cooperate.

14.1 Healthcare and Life Sciences

Potential applications include:

- medical imaging;
- genomic analysis;
- drug discovery;
- clinical decision-support systems;
- biomedical simulation; and
- distributed research.

The architectural benefit would be interoperability among specialized models and computing resources while maintaining strong security and governance controls.

Any real-world healthcare implementation would require independent validation, clinical governance, privacy controls, and regulatory review.

14.2 Financial Systems

Potential applications include:

- fraud detection;
- portfolio optimization;
- risk analysis;
- market simulation;
- anomaly detection; and
- financial forecasting.

The central opportunity is the coordination of multiple specialized models operating on high-volume data streams.

Quantum computing could eventually become relevant to selected optimization and sampling problems, but no general quantum advantage should be assumed without workload-specific benchmarking.

14.3 Manufacturing and Robotics

Manufacturing environments increasingly combine:

- machine vision;
- robotics;
- predictive maintenance;

- digital twins;
- industrial IoT;
- scheduling;
- quality control; and
- supply-chain optimization.

eXp-AIOS could provide a common execution architecture across these systems while allowing individual AI components to remain specialized.

14.4 Autonomous Systems

Autonomous systems require continuous interaction among perception, prediction, planning, navigation, and control.

The proposed architecture could allow these components to operate as cooperating AI services rather than isolated models.

Importantly, this would complement rather than replace established transportation and communications standards.

14.5 Aerospace and Space Systems

Aerospace environments provide an extreme example of heterogeneous computing.

Spacecraft may combine:

- constrained edge processors;
- specialized accelerators;
- autonomous navigation systems;
- sensor fusion;

- communications systems;
- ground-based AI; and
- cloud infrastructure.

A hardware-independent orchestration layer could potentially allow computational tasks to move among these resources according to availability and mission requirements.

Flight-critical implementations would, of course, require extensive certification and verification.

---

15. The Research and Validation Program

The credibility of eXp-AIOS ultimately depends on experimentation.

The next stage should therefore not be another theoretical expansion of the architecture.

It should be a benchmarking program.

Experiment 1 — NAISCII Interoperability

Construct a prototype connecting at least three heterogeneous AI frameworks.

Measure:

- integration effort;
- serialization overhead;
- latency;
- throughput;
- memory consumption;
- developer complexity; and
- error rates.

Compare the results against established approaches.

Experiment 2 — NASOF Performance

Implement NASOF alongside existing data representations.

Benchmark:

- encoding;
- decoding;
- memory footprint;
- cache behavior;
- transfer time;
- zero-copy potential;
- accelerator utilization; and
- total end-to-end execution time.

The critical comparison is not whether NASOF works.

It is whether NASOF works better for a defined class of AI workloads.

Experiment 3 — Maple-A Scheduling

Implement Maple-A as software.

Provide it with heterogeneous CPU, GPU, and accelerator resources.

Measure whether dynamic routing improves:

- utilization;
- latency;

- throughput;
- power efficiency;
- cost per inference; and
- workload completion time.

Experiment 4 — Quantum Handoff

Build a small hybrid classical/quantum workflow.

Compare:

classical-only execution

against

classical + quantum execution

across carefully selected problems.

Measure the entire pipeline rather than only the QPU execution time.

This distinction is essential because communication and state-preparation costs may overwhelm the quantum computation itself.

Experiment 5 — Security

Evaluate OSCQAS against established security mechanisms.

Measure:

- authentication latency;
- encryption overhead;
- throughput;

- key-management complexity;
- failure recovery;
- attack resistance; and
- interoperability with established cryptographic standards.

Security claims should be subjected to independent review.

---

16. Development Roadmap

The architecture should mature in stages.

Stage I — Software Proof of Concept

Implement:

- NAISCII prototype;
- NASOF prototype;
- MAPLE orchestration environment;
- Maple-A software scheduler; and
- OSCQAS security framework.

No custom silicon is required.

Stage II — Distributed Prototype

Demonstrate communication among:

- multiple AI frameworks;
- multiple GPUs;
- CPUs;

- edge devices; and
- cloud resources.

Stage III — Quantum Integration

Connect the orchestration architecture to available quantum computing platforms.

Demonstrate workload partitioning and classical fallback.

Stage IV — Hardware Acceleration

Identify the specific operations where dedicated hardware produces measurable advantages.

Develop FPGA prototypes before pursuing custom silicon.

Stage V — Maple-A Silicon

Only after the software architecture and workload requirements have been validated should a dedicated Maple-A processor become a fabrication objective.

This sequence reduces technical and financial risk by validating the architecture before committing to semiconductor development.

---

17. The Central Proposition

The computing industry has historically progressed through abstraction.

Applications became independent of individual processors.

Operating systems became independent of individual applications.

Virtualization separated software environments from physical machines.

Cloud computing separated infrastructure from physical data centers.

The next abstraction may be the computational substrate itself.

An application should increasingly be able to ask:

What computation needs to occur?

rather than:

Which processor must perform it?

That is the architectural proposition behind eXp-AIOS.

The system does not assume that GPUs will disappear.

It does not assume that CPUs will disappear.

It does not assume that TPUs or NPUs will disappear.

It does not assume that quantum computers will replace classical computers.

Instead, it proposes an architecture in which all of these resources can become participants in a common computational environment.

---

18. Conclusion

Artificial intelligence is moving from isolated models toward interconnected computational systems.

At the same time, computing infrastructure is moving from relatively homogeneous processors toward heterogeneous collections of CPUs, GPUs, TPUs, NPUs, edge processors, specialized accelerators, and eventually increasingly capable quantum processors.

The resulting challenge is architectural.

How can these systems communicate?

How can data move between them efficiently?

How can workloads be divided according to the capabilities of each processor?

How can security remain consistent across those boundaries?

And how can today's classical applications gain access to tomorrow's computational technologies without requiring complete redevelopment?

eXp-AIOS proposes one answer.

Its architecture combines:

NAISCII for inter-AI communication,

NASOF for structured AI objects,

MAPLE 1.0 for orchestration,

Maple-A for hardware abstraction and workload routing,

and OSCQAS for security and cryptographic policy.

Together, these components form a proposed architecture for an expandable AI computing environment.

The immediate objective is not to claim that the architecture has already solved the memory wall, eliminated AI integration overhead, or produced quantum advantage.

The immediate objective is to build it.

The most important next step is therefore empirical validation.

If NAISCII can demonstrate measurable interoperability advantages, if NASOF can demonstrate measurable execution benefits, if Maple-A can improve heterogeneous hardware utilization, and if the quantum handoff can demonstrate useful hybrid execution without prohibitive communication overhead, then the architecture can progress from a systems hypothesis into an engineered platform.

That progression is the essential purpose of the eXp-AIOS program:

Architectural Summary

Component	Proposed Role	Primary Research Question
eXp-AIOS	Overall AI systems architecture	Can heterogeneous AI and computing resources operate through a common abstraction layer?
NAISCII	Inter-AI communication/intermediate representation	Can model-to-model interoperability be improved beyond existing approaches?
NASOF	Structured AI object representation	Can a standardized object architecture improve execution or data locality?
MAPLE 1.0	Development and orchestration environment	Can AI workflows be unified across heterogeneous resources?
Maple-A	Hardware abstraction and workload routing	Can computation be dynamically assigned to the most appropriate processor?
OSCQAS	Security and cryptographic policy layer	Can security controls be applied consistently across distributed AI execution?
Quantum Bridge	Classical-to-quantum workload interface	Can quantum processors become optional

computational resources
within existing AI workflows?

Status of the Proposal

The eXp-AIOS architecture described in this paper should be understood as a proposed research and engineering framework.

The existence of the underlying infrastructure problems — including data movement, heterogeneous accelerator management, AI interoperability, and the constraints of contemporary quantum computing — provides the motivation for the architecture.

The performance advantages of the proposed eXp-AIOS components remain testable hypotheses.

No performance, energy, latency, security, regulatory, or quantum-advantage claim should be considered demonstrated until supported by reproducible implementation and independent benchmarking.

This distinction is not a limitation of the proposal.

It is the foundation for turning the architecture into a credible engineering program.

Selected Technical References

National Institute of Standards and Technology (NIST). Post-Quantum Cryptography project and finalized FIPS 203, FIPS 204, and FIPS 205 standards. NIST states that organizations should begin migration toward quantum-resistant cryptography.

Apache Arrow. Language-independent columnar memory format and data-interchange architecture supporting efficient memory access, vectorization, serialization, and zero-copy data movement.

Additional references should be incorporated during the prototype and validation phase, particularly covering ONNX, MLIR, distributed AI orchestration, accelerator scheduling, quantum-classical hybrid computing, quantum error correction, and AI systems architecture.