Time to Revisit the Internet Layering Principle: AI-Driven Cross-Layer Optimization

JP Vasseur, PhD

Sr Distinguished Engineer NVIDIA jvasseur@nvidia.com

July 2025

Executive Abstract

The foundational principle of layered architecture, a cornerstone of the Internet's success, now creates a performance barrier for high-value, modern applications due to its strict information isolation. Each layer operates with an architectural blind spot, optimizing its behavior using only the information available within its own domain. This paper argues that the time has come to augment this model with AI-driven cross-layer intelligence. By training machine learning models on holistic, multi-layer telemetry—including direct user experience feedback—systems can move beyond optimizing isolated network metrics. The new paradigm is to learn and predictively optimize for the high-level Service-Level Objectives (SLOs) that truly define performance, such as Quality of Experience (QoE) and Job Completion Time (JCT). This approach does not replace the layering principle, which remains essential for many tasks, but complements it through a non-disruptive "soft layering" model. It represents a practical and necessary evolution for engineering the next generation of adaptive, high-performance systems.

1. The Layered Architecture: A Foundation of Modern Networking

The principle of layered Internet protocol architecture is a cornerstone of modern network and system design. It provides a formal methodology for decomposing the complexity of data communication into a set of distinct, hierarchical layers. Each layer is designed to perform a specific set of functions, relying on services provided by the layer below it and, in turn, offering its own services to the layer above. This abstraction is most famously conceptualized in the seven-layer Open Systems Interconnection (OSI) model and is practically implemented in the dominant TCP/IP protocol suite that underpins the global Internet.

The success and longevity of this layered approach are attributable to several key contributions. First, it introduces profound modularity. By encapsulating the functions of a given layer, it hides underlying complexity. An application developer, for instance, can utilize the Hypertext Transfer Protocol (HTTP) at the application layer without needing to understand the intricacies of TCP's congestion control algorithms at the transport layer or the specific modulation schemes used by Wi-Fi at the physical layer. This separation of concerns simplifies development and accelerates innovation.

A similar abstraction is evident in High-Performance Computing (HPC) and large-scale AI data centers, illustrating the power of layering in non-IP environments like the InfiniBand Architecture. A computational scientist using libraries such as MPI (Message Passing Interface) or NCCL for

distributed training can focus entirely on their parallel algorithm logic. They are shielded from the complexities of the InfiniBand fabric, whose lower layers manage critical functions like kernel-bypass communication. The InfiniBand link layer itself provides robust, hardware-level capabilities including credit-based flow control to ensure lossless behavior, as well as error detection and automatic retransmission mechanisms, as defined in its official specification (InfiniBand Trade Association, 2022). These sophisticated features are handled transparently by the fabric, enabling developers to achieve high performance without being experts in the underlying network transport.

Second, layering is the bedrock of standardization and interoperability, a principle critical to the Internet's explosive growth over the past three decades. By defining a common set of rules, it dismantled proprietary, single-vendor network models, fostering a competitive multi-vendor ecosystem that lowered costs and broke vendor lock-in. This open foundation enabled "permissionless innovation," allowing anyone to create new applications and services with the confidence they would run on any standard network. Most importantly, it provided a common “language” especially IP that allowed thousands of different networks to connect, creating a single, seamless global network instead of many isolated digital islands. This interoperability, defined by clear interfaces between layers, was a prerequisite for the growth of decentralized, global-scale networks.

Finally, this architecture facilitates independent evolution. Protocols and technologies at one layer can be upgraded or replaced without necessitating a complete redesign of the entire stack. The transition from 1-Gbps to 10-Gbps to 100s-Gbps Ethernet, the introduction of new Wi-Fi links standard, or the development of new transport protocols (e.g., TCP, UDP, SCTP, MPTCP, and QUIC (Iyengar & Thomson, 2021)) can all occur independently. This capacity for incremental, non-disruptive innovation has been fundamental to the Internet's ability to scale and adapt over several decades.

In summary, the layered architecture was a necessary and highly successful paradigm. It provided the stable, scalable, and interoperable foundation—codified in foundational standards like IP (Postel, 1981a) and TCP (Postel, 1981b)—upon which the Internet evolved and grew. However, while the benefits of this model are substantial, its rigid enforcement of isolation introduces constraints that are increasingly challenged by the performance and dynamism required in contemporary network environments, at least for some applications.

2. Early Attempts at Cross-Layer Communication

Despite the principle of strict layer isolation, the need for richer communication between applications and the network was recognized early in the Internet’s history. It was clear that a purely "best-effort" service model was insufficient for emerging real-time applications. This led to the development of sophisticated architectures designed to allow applications to signal their performance requirements to the network, representing formal, if ultimately unsuccessful, attempts at cross-layer communication.

The most prominent example in the IP world was the Integrated Services (IntServ) architecture, which used the Resource Reservation Protocol (RSVP) as its signaling mechanism. The model allowed an application (Layer 7) to request a specific Quality of Service (QoS)—such as guaranteed bandwidth and bounded delay—for a particular data flow. RSVP would then carry this request through the network (Layer 3), and each router along the path would attempt to reserve

the necessary resources. This provided a direct communication channel from the application to the network infrastructure.

A more scalable, albeit less granular, approach was Differentiated Services (DiffServ). Here, traffic is classified and marked at the edge of a network with a specific code point (DSCP) in the IP header. Core routers then apply different "per-hop behaviors" based on these markings, giving preferential treatment to priority traffic. While the signaling is less explicit than RSVP, DiffServ still represents a form of cross-layer communication, where an application's requirements are translated into a network-layer marking that influences forwarding behavior.

These concepts were not limited to the IP world. Connection-oriented technologies like Asynchronous Transfer Mode (ATM) had powerful, built-in QoS capabilities at their core, allowing applications to request service classes with specific performance guarantees.

However, these early architectures failed to gain widespread adoption across the public Internet, primarily due to issues of scalability and complexity. The per-flow state required by IntServ/RSVP was deemed untenable for Internet-scale routers. DiffServ, while more scalable, proved difficult to manage consistently across different administrative domains, limiting its utility to private enterprise networks. Ultimately, the simpler, stateless, best-effort model prevailed. Thus, while the need for cross-layer signaling was well understood, the fundamental challenge of implementing it in a scalable and deployable manner remained unsolved.

3. The Isolation Principle: Emerging Limitations in High-Demand Systems

The failure of early QoS architectures to be widely deployed meant that the strict encapsulation of the layered model remained the de facto standard. Each layer operates with an **architectural blind spot**, optimizing its behavior using only the information available within its own domain. This leads to locally optimal decisions that are often globally suboptimal for the end-to-end service. This information boundary prevents a holistic view of the system, leading to significant inefficiencies, particularly in environments with demanding performance requirements or constrained resources. Several well-understood examples illustrate this structural problem.

Wireless Networks: A classic case is the performance of TCP over wireless links (e.g., Wi-Fi, 5G). TCP's congestion control algorithms were designed for wired networks, where packet loss is almost exclusively a signal of network congestion. In a wireless environment, however, loss is frequently caused by factors at the physical (PHY) and MAC layers, such as radio interference or signal fading. Because the transport layer is unaware of the root cause, it reacts incorrectly, throttling throughput unnecessarily (this issue was historically addressed through specialized workarounds like performance-enhancing proxies that split the TCP connection, or robust link-layer retransmission schemes designed to hide wireless errors from the transport layer).

Video and Real-Time Media: The quality of experience (QoE) for real-time applications is highly sensitive to network conditions. Networks often provide static QoS guarantees, but the application's tolerance to impairments is dynamic. A high-motion 4K video stream is far more sensitive to jitter than a static presentation. A lack of direct, real-time communication from the network layers prevents proactive, content-aware resource allocation by the application's adaptive bitrate (ABR) codecs.

Semantic-Aware Routing in IoT: In low-power and lossy networks (LLNs), traditional routing based on simple network-layer metrics like hop count is insufficient. Semantic-aware routing, which considers the content and criticality of the data, is often required. While the IETF's RPL (Winter et al., 2012) is a notable departure from strict layering, the broader field has seen extensive research into cross-layer solutions for wireless sensor networks to address these challenges (Lin et al., 2017). These solutions allow routing decisions to incorporate application-layer metadata and physical-layer metrics, such as a node's remaining energy.

Delay Tolerant Networks (DTN): In Delay Tolerant Networks (DTNs), which operate in environments with intermittent connectivity such as space or vehicular systems, the bundle layer (above the transport layer) handles store-and-forward mechanisms. Forwarding decisions at this layer can be improved by incorporating physical layer information about future link availability or contact durations. For example, research has shown that a joint routing and scheduling policy that uses cross-layer information can optimize message delivery based on predicted contact opportunities (Krifa et al., 2008).

Data Center Networking for ML/HPC: Modern data centers designed for large-scale AI, sometimes referred to as "AI Factories," (NVIDIA, 2024) represent a pinnacle of system complexity where the limitations of strict layering are particularly acute. Training a large language model (LLM) often involves thousands of GPUs operating in concert using sophisticated techniques like 3D parallelism (tensor, pipeline, and data parallelism). This process involves a cascade of layered, yet interdependent, systems. A high-level job scheduler initiates the process, assigning workloads to specific GPUs. However, its decisions are deeply impacted by underlying physical topologies: GPUs may be tightly coupled within a chassis via high-speed NVLink, or distributed across racks connected by an InfiniBand or Ethernet fabric, typically in a CLOS architecture. Each of these physical connection types has its own distinct topology. On top of this, the NVIDIA Collective Communications Library (NCCL), which manages the fundamental communication primitives between GPUs, operates with its own logical communication topology. Furthermore, the network fabric itself has its own mechanisms for detecting congestion, handling errors, and retransmitting data. This multi-layered system—from the scheduler to NCCL to the physical network—is a perfect example where strict isolation is detrimental. Optimizing for Job Completion Time (JCT) requires coordination across these layers; for instance, the scheduler's placement decisions are far more effective if it is aware of both the NCCL communication pattern and the underlying network topology. As a result, new approaches that facilitate this cross-layer knowledge sharing are beginning to emerge to tackle this complexity.

Redundant Reliability Mechanisms: In many communication stacks, reliability mechanisms are implemented independently at multiple layers, each with its own timers. For example, a wireless link layer may use a Hybrid ARQ protocol to recover from transmission errors, while the transport layer (e.g., TCP) implements its own retransmission logic. The core issue is that these timers are uncoordinated. Without a holistic view, it is difficult to ensure that the transport-layer timer is always longer than the maximum possible link-layer recovery time, often requiring careful, manual tuning by network administrators. If the timers are misaligned, a packet that is simply delayed during link-layer recovery may be presumed lost by TCP, triggering a redundant and wasteful retransmission that adds to congestion and latency.

Challenges of End-to-End Encryption: The widespread adoption of end-to-end encryption is essential for security but renders traditional deep packet inspection (DPI) ineffective. Modern protocols like TLS 1.3 (Rescorla, 2018) encrypt most of the connection metadata, preventing network devices from inspecting payloads to classify traffic for QoS or security monitoring. This forces a reliance on less precise heuristics, limiting the network's ability to differentiate between critical and non-critical flows without some form of cross-layer signaling.

These examples reveal a common pattern: for a growing and critical class of applications, rigid adherence to layer isolation is no longer a virtue but a fundamental impediment to performance. The issue is not that layering is obsolete—it remains a powerful principle for general-purpose networking. The issue is that for these specific, high-stakes use cases, a more flexible and intelligent approach is now required. The historical challenge was never the desire for cross-layer optimization, but the absence of a technology capable of managing its immense complexity in a scalable way. As the next section will argue, that technology has now arrived.

4. AI-Driven Cross-Layer Intelligence: A New Optimization Paradigm

Overcoming the limitations of strict layering requires a mechanism that can process information across architectural boundaries in a dynamic and scalable manner. While specialized cross-layer protocols have demonstrated clear benefits in niche applications, a general-purpose solution has remained elusive. Furthermore, it has been argued that while machine learning techniques have been available for over a decade, their application to complex network optimization problems has been slow, representing a significant missed opportunity to address these long-standing challenges (Vasseur, 2025).

The emergence of advanced artificial intelligence (AI) and machine learning (ML) systems provides a powerful new toolkit to address this challenge, enabling a shift from static, isolated control to dynamic, holistic optimization.

The core function of this AI-driven approach is to create a system-wide control loop that operates on three principles:

1. **Holistic Observability:** An AI model can ingest and process high-dimensional telemetry streams gathered concurrently from every layer of the stack. This includes physical-layer metrics like signal-to-noise ratio, MAC-layer retransmission statistics, network-layer latency and jitter measurements, transport-layer round-trip times, and—critically—application-layer performance indicators such as transaction success rates or video quality scores.
2. **Learning Complex System Dynamics:** The primary value of an ML model, particularly a deep neural network (DNN), is its ability to learn the complex, non-linear relationships between these disparate data points. It can discover correlations that are intractable to define with static formulas or manual heuristics—for example, how a subtle increase in PHY-layer interference patterns predicts a future drop in application-level QoE. This transforms the system from a collection of independent components into a single, observable entity.
3. **Predictive and Proactive Control:** By understanding these deep system dynamics, the model can move beyond reacting to past failures (e.g., TCP responding to a dropped packet) to predicting future states. This enables proactive control actions. The system can anticipate performance degradation and make corrective adjustments—such as rerouting traffic,

adjusting QoS parameters, or signaling an application to adapt—*before* a service-level objective is breached.

This methodology facilitates a fundamental change in the goal of network management: a-shift away from optimizing intermediate network KPIs (e.g., minimizing packet loss) and toward directly optimizing the high-level Service-Level Objectives (SLOs) that reflect true user or business outcomes.

QoE-Centric SLA Definition and Enforcement: A more profound application of this paradigm moves beyond simply predicting outcomes. Consider training a Deep Neural Network not just on network telemetry but also on explicit user feedback signals (e.g., satisfaction ratings, application usage patterns). The objective is to create a comprehensive model of user-perceived Quality of Experience (QoE). The primary function of such a a model is to dynamically *learn* the precise set of lower-layer SLAs (latency, jitter, loss) required to achieve a high level of user satisfaction for a given application context. This replaces the static, one-size-fits-all SLA with a dynamic, context-aware one derived directly from user experience. Once defined, this QoE model can be used in a cognitive control loop to actively orchestrate the network—adjusting routing, QoS queues, or even application codecs—to continually meet these bespoke SLAs, a concept central to cognitive networking (Vasseur, 2023).

Workload-Aware Data Center Optimization: In a distributed ML environment, an AI model can learn to recognize the unique network traffic "fingerprints" of different workloads and their phases. This allows the system to optimize directly for high-level KPIs like Job Completion Time (JCT) for training or Time to First Token (TTFT) for inference. For example, a reinforcement learning agent could tune job scheduling policies based on their observed impact on network congestion and end-to-end throughput, orchestrating the fabric to anticipate bandwidth-intensive phases and minimize expensive compute stalls.

Importantly, this AI-driven approach does not mandate the abandonment of modularity. Instead, cross-layer intelligence can be introduced with minimal disruption through a form of **"soft layering."** This involves creating **telemetry fusion layers** that aggregate and unify metrics from across the stack into a common data plane for model consumption. The AI model's output can then be fed back into the system via **intelligent control loops**, guiding the decision-making of existing protocols at specific layers (e.g., adjusting congestion control parameters or traffic shaping policies) through well-defined APIs or metadata channels.

In essence, AI provides a scalable and adaptive mechanism to intelligently violate the strict information boundaries of the layered model without discarding its architectural benefits. This approach is not intended as a universal replacement for intra-layer optimization, as many technologies are most effectively managed within their own domain. Instead, it offers a powerful, complementary strategy for complex scenarios where system-level performance is clearly constrained by a lack of cross-layer context. It allows the system to retain its modular structure while operating with a level of holistic awareness that was previously unattainable.

5. Conclusion

The layered architecture, a foundational principle of networking, provided the modularity and standardization necessary to build the global Internet. Its core design trade-off—the strict isolation

of layers—was instrumental to this success. However, as this paper has argued, while the layering principle remains effective for general-purpose networking, its strict enforcement of isolation has become a structural impediment for a critical class of modern systems. In these dynamic, resource-constrained, and performance-sensitive environments, each layer operates with an architectural blind spot, optimizing its behavior using only the information available within its own domain, which prevents globally optimal behavior.

Artificial intelligence offers a practical and powerful paradigm to transcend these limitations. By ingesting and analyzing holistic, multi-layer telemetry, ML models can learn the complex dynamics of an entire system. This enables a transformative shift from reactive, KPI-driven management to predictive, SLO-focused optimization, where the primary goal is to directly enhance end-to-end outcomes such as user Quality of Experience or Job Completion Time. This can be achieved without a disruptive architectural overhaul, using "soft layering" techniques that augment existing modular systems with intelligent control loops.

Moving forward, the integration of AI-driven cross-layer intelligence represent a necessary evolution in system design. Realizing its full potential will require continued innovation in areas such as standardized, high-resolution telemetry frameworks and open, programmable control interfaces. Embracing this paradigm is not about abandoning the layered model, which remains essential for many tasks, but about augmenting it with the holistic intelligence required to meet the demands of the next generation of applications.

The views and opinions expressed in this paper are solely those of the author and do not necessarily reflect the official policy or position of NVIDIA or any of its affiliates. The information and analysis presented are based exclusively on publicly available data, documentation, and reports accessible on the web.

6. References

InfiniBand Trade Association. (2022). *InfiniBand™ architecture specification, volume 1, release 1.5*.

Iyengar, J., & Thomson, M. (Eds.). (2021, May). *QUIC: A UDP-based multiplexed and secure transport* (RFC 9000). IETF.

Krifa, A., Barakat, C., & Spyropoulos, T. (2008). An optimal joint routing and scheduling policy for Delay Tolerant Networks. In *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*.

Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H., & Zhao, W. (2017). A survey on cross-layer solutions for wireless sensor networks. *IEEE Communications Surveys & Tutorials*, *19*(1), 571-595.

NVIDIA. (2024). *NVIDIA DGX SuperPOD: The blueprint for AI factories*.

Postel, J. (1981a, September). *Internet protocol* (RFC 791). IETF.

Postel, J. (1981b, September). *Transmission control protocol* (RFC 793). IETF.

Rescorla, E. (2018, August). *The Transport Layer Security (TLS) protocol version 1.3* (RFC 8446). IETF.

Vasseur, J. P. (2023). *Cognitive networking: The new networking paradigm*. Retrieved from https://assets.zyrosite.com/A1aglvGGy1F64BNz/wp-cognitive-networks-mv05jQOpq0CJ7eNd.pdf

Vasseur, J. P. (2025, July). *Beyond protocol: A decade of missed opportunities in network AI*. Retrieved from https://assets.zyrosite.com/A1aglvGGy1F64BNz/beyondprotocol-july2025-m7VbXp2BLKTx071B.pdf

Winter, T., Thubert, P., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, J. P., & Alexander, R. (2012, March). *RPL: IPv6 routing protocol for low-power and lossy networks* (RFC 6550). IETF.