International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(21s) 2026 pp. 522-535
https://doi.org/10.70917/ijcisim-2026-5328
Article

Scalable General-Purpose Cloud Compute: Architectural Innovations for Sustainable AI Infrastructure

Priyadarshni Shanmugavadivelu

Birla Institute of Technology and Science, Pilani, India

Abstract: The rapid growth of artificial intelligence (AI), machine learning, and large-scale digital services is placing unprecedented demand on cloud infrastructure, making scalable and sustainable compute increasingly important. While advances in processors and accelerators continue to improve computational capability, architectural coordination can unlock efficiency gains that hardware-generation improvements alone may not fully capture, particularly across increasingly heterogeneous compute environments (Armbrust et al., 2010). This paper will examine virtualization, workload scheduling, the provisioning of heterogeneous resources and infrastructure lifecycle management's effects on infrastructure efficiency at hyperscale. The paper extends the NIST cloud service definition (Mell & Grance, 2011) and most recent research on energy efficient resource management (Khan et al., 2022, Ilager et al., 2021) to propose a Sustainable Compute Lifecycle Framework comprised of five inter-connected phases: Platform Planning, Platform Deployment and Optimization, Platform Operations, Hardware Modernization, and Hardware Retirement and Resource Reclamation. The paper also explores new trends such as Compute Express Link (CXL) for memory pooling and disaggregation (Das Sharma et al., 2024;Chen et al., 2024) and the use of AI for infrastructure scheduling (Sanjalawe et al.,2025). The key message is that there is a new central design coordination of the cloud architecture that enables sustainable growth of the cloud, not merely increasing incremental capacity on hardware.

Keywords: Cloud Computing, Sustainable Infrastructure, Virtualization, Cloud Architecture, Resource Optimization, Hardware Lifecycle, Compute Express Link

1. INTRODUCTION

Enterprise applications, scientific computing, and large-scale AI workloads all depend on cloud platforms that must balance performance, cost, and environmental impact at once. NIST's definition of cloud computing treats rapid elasticity and resource pooling as defining characteristics of the model (Mell & Grance, 2011). Meeting those characteristics at hyperscale, however, is an operational and architectural challenge that goes well beyond adding processor capacity, and the gap between the model's stated characteristics and the operational reality of running it at scale has widened as workload diversity and demand volatility have both increased.

Infrastructure growth used to mean deploying newer hardware generations for performance headroom. That pattern is running into diminishing returns: AI workload demand is accelerating faster than generational processor improvements alone can absorb (Kachris, 2025). The industry's own efficiency metrics illustrate the stakes. The average power usage effectiveness (PUE) across the data center industry has held at approximately 1.56 for five consecutive years, according to Uptime Institute's 2024 Global Data Center Survey, meaning that a substantial share of total energy consumption continues to go toward overhead, cooling, power conversion, and distribution losses, rather than computation itself (Uptime Institute, 2024). This flat industry-wide average persists even as newer, larger facilities post meaningfully better efficiency figures, suggesting that gains achievable through architecture and operational practice are being offset by a long tail of older, less efficient infrastructure still in service. The persistence of this flat average across five consecutive survey years is itself a signal that efficiency is not simply a function of

hardware age; if it were, continuous fleet-wide hardware turnover would have moved the industry average meaningfully by now. Something else is holding the average down, and this paper argues that architectural coordination, rather than hardware refresh cadence alone, is the more plausible explanation. This reading is consistent with industry-level empirical evidence: Park et al. (2023) find, using economy-wide data across dozens of industries over two decades, that adoption of cloud-based IT services is associated with measurable improvements in users' energy efficiency, a finding that is more naturally explained by architectural and operational factors specific to how cloud platforms are run than by the underlying hardware alone, since the hardware available to any given industry adopting cloud services did not change simply because that industry adopted cloud computing.

Providers need to squeeze more efficiency from infrastructure already deployed, retire aging capacity on a structured basis, and place workloads intelligently across heterogeneous resources. General-purpose Compute sit at the center of this challenge. Beyond running customer workloads directly, it coordinates virtualization, scheduling, networking, storage, and lifecycle transitions across large server fleets (Barroso et al., 2013). This paper argues that coordination across these functions, not hardware expansion by itself, is the lever cloud providers actually have available for sustainable infrastructure growth. This is a stronger claim than simply noting that architecture matters alongside hardware; it is a claim about relative leverage; that for a hyperscale operator facing continued AI-driven demand growth, architectural coordination represents an increasingly important efficiency lever alongside hardware-generation improvements, particularly as workload diversity and infrastructure heterogeneity increase.

Existing treatments of cloud infrastructure architecture, including the foundational and resource-management literature reviewed in Section 2, address planning, deployment, scheduling, and hardware modernization as separate concerns, each with its own body of work and its own evaluation criteria. What is largely absent from that literature is an end-to-end lifecycle coordination framework that treats these concerns as stages in a single continuous process rather than as independent problems to be optimized in isolation. Three gaps follow from this absence. First, the cloud architecture models surveyed in this review stop, in practice, at deployment and operations: virtualization, scheduling, and allocation research is extensive, but it rarely extends its scope to what happens once hardware approaches the end of its useful service life. Second, and following directly from the first gap, none of the frameworks in the literature reviewed here integrates retirement as a coordinated stage rather than an isolated end-of-life event; retirement is treated as something that happens after the interesting architectural decisions have already been made, rather than as a phase whose efficiency depends on decisions made much earlier in a platform's life. Third, none of the lifecycle models surveyed in the literature reviewed here spans the full arc from planning through reclamation as a single coordinated structure; the closest analogues address two or three adjacent stages well but do not connect planning-stage decisions to retirement-stage outcomes. The Sustainable Compute Lifecycle Framework this paper proposes in Section 4 differs from that prior work specifically by closing this gap: it treats Platform Planning, Deployment and Optimization, Operational Management, Hardware Modernization, and Retirement and Resource Reclamation as five interdependent phases of one coordinated structure, in which decisions made at any phase are understood to constrain and shape what is achievable at every later phase, including the reclamation phase the frameworks reviewed here largely omit.

What follows reviews how general-purpose cloud compute architecture evolved, synthesizes recent literature on virtualization, scheduling, and hardware lifecycle management, and introduces a lifecycle-oriented framework for coordinating these functions across a platform's operational life. The paper closes by considering how composable infrastructure and AI-assisted management extend this coordination principle into emerging architectural territory, and by examining what the sustainability implications of coordinated lifecycle management look like once retirement and reclamation are treated as an integral phase rather than an afterthought.

Research Approach

This study uses a conceptual synthesis of peer-reviewed and authoritative literature on cloud architecture, virtualization, resource scheduling, heterogeneous compute, infrastructure sustainability, and hardware lifecycle management. Prior work was selected based on its relevance to hyperscale infrastructure efficiency and used to identify recurring architectural dependencies and gaps across the compute lifecycle. These findings were synthesized

to develop the Sustainable Compute Lifecycle Framework presented in Section 4; the framework is therefore intended as a literature-grounded conceptual model rather than an empirically validated system.

2. LITERATURE REVIEW: EVOLUTION OF GENERAL-PURPOSE CLOUD COMPUTE

Cloud infrastructure architecture moved through several identifiable phases to reach its current form. Early platforms relied on hypervisor-based virtualization to consolidate workloads onto shared servers; Xen was an early proof point, showing that paravirtualization could deliver near-native performance while keeping workloads isolated from one another (Barham et al., 2003). That technical foundation made possible the elastic, multi-tenant service model later cloud platforms formalized (Armbrust et al., 2010). The core insight from this era, that isolation and resource sharing need not trade off against performance if the hypervisor layer is designed carefully, still underlies how modern cloud platforms reason about multi-tenancy today, even as the scale and workload diversity involved has grown by orders of magnitude. What has changed since Xen's introduction is less the underlying principle than the operating environment in which it is applied: a hypervisor design that performed well for a modest number of homogeneous virtual machines on a single physical host now has to perform comparably well across thousands of hosts running a far broader mix of workload types, many of which did not exist in any recognizable form when the original virtualization research was conducted.

Cloud adoption then matured into virtual machine families built for distinct workload profiles: compute-intensive, memory-intensive, storage-intensive, network-intensive. This specialization reflected a recognition that no single VM configuration serves every workload efficiently, and that matching hardware characteristics to workload demands is itself a meaningful efficiency lever, independent of any single hardware generation's raw performance. Marquez and Mondragon (2021) found that heterogeneous hardware environments need allocation frameworks accounting jointly for data placement and virtual machine assignment; treating compute and storage decisions separately misses interactions between the two. Their genetic-programming-based allocation approach makes a useful point: resource allocation on heterogeneous infrastructure is a joint optimization problem across resource types, not a scheduling problem alone. Their findings suggest that specialization at the VM-family level, while useful, is not sufficient on its own; the allocation logic governing how workloads are placed onto that specialized hardware matters just as much as the hardware itself. A platform could offer a full range of specialized VM families and still under-deliver on efficiency if the allocation logic routing workloads to those families does not account for how compute and storage resource needs interact for any given workload.

This was further accelerated by the advent of large scale AI workloads. Kachris (2025) provides an overview of hardware accelerator methods for LLM inference and observes that none of the three types of accelerators – GPUs, FPGAs or custom – is dominant in all types of inference workloads. However, general-purpose compute continues to orchestrate, manage memory, and manage networking around the accelerator workload, independent of accelerator performance (Das Sharma et al., 2024). As the variety of acceleration has grown, this orchestration burden has increased, too: An orchestration that performs training on GPUs and inference on FPGAs imposes a different set of orchestration requirements on the underlying general-purpose layer than an orchestration that uniformly applies a single type of accelerator. Consequently, the production value of an accelerator architecture cannot be evaluated solely through accelerator-level performance metrics.

Energy and resource efficiency have become central concerns throughout this evolution. Khan et al. (2022) reviewed machine-learning-centric approaches to cloud resource management and found static allocation policies increasingly giving way to data-driven methods for workload estimation, task scheduling, and VM consolidation. Their review identifies workload estimation accuracy as a persistent bottleneck: even sophisticated scheduling logic underperforms when the demand forecasts feeding it are unreliable, which ties resource-management efficiency directly back to the demand-forecasting concerns raised earlier. This is a point worth dwelling on, since it reframes what "better scheduling" actually requires. A scheduling algorithm can be theoretically optimal given accurate demand inputs and still perform poorly in production if the inputs themselves are systematically biased, a distinction that

matters when evaluating why a given scheduling approach underperforms its published benchmarks once deployed against real, noisy workload data. Ilager et al. (2021) catalog a wide range of energy-efficient resource management techniques from the preceding half-decade and observe that no single technique addresses efficiency across every dimension of hyperscale operation. That observation is itself an argument for framework-level, coordinated thinking rather than isolated point solutions; a technique that improves scheduling efficiency in isolation may still underperform if it is not coordinated with virtualization-layer decisions and hardware modernization timing, the three concerns this paper treats as jointly determining infrastructure efficiency. Read together, Khan et al. (2022) and Ilager et al. (2021) point toward the same underlying conclusion from different angles: efficiency techniques evaluated in isolation, whether at the scheduling layer or across the broader resource-management taxonomy, systematically understate what is achievable once those techniques are coordinated against one another rather than deployed independently.

Verma et al. (2024) shed additional light on this in their study on energy-efficient workflow scheduling algorithms, a study that concentrates specifically on the propagation of scheduling decisions across dependent tasks in workflow-structured workloads, a pattern that is becoming more and more prevalent in AI pipelines with a succession of data preparation, training, and inference stages. They've found that improving the scheduling efficiency of a single independent task doesn't necessarily carry over to these types of workloads that are structured into a workflow, as scheduling one stage affects the scheduling of other stages. Given that coordination of scheduling across stages is important for maximizing efficiency within a single workflow, it is much more compelling that scheduling across a fleet should also be coordinated with virtualization and hardware modernization decisions.

Buyya et al. (2024) approach the same efficiency question from a broader management-integration perspective, arguing that energy efficiency and sustainability in cloud computing depend on integrated management of data center resources and workloads rather than on any single optimization technique applied in isolation. Their vision paper is notable for treating energy efficiency not as a separate sustainability objective sitting alongside performance and cost objectives, but as an outcome that emerges from how well those other objectives are jointly managed; a platform that manages performance and cost well, in their framing, tends toward better energy efficiency almost as a byproduct, whereas a platform that manages energy efficiency as an isolated goal, disconnected from performance and cost decisions, tends to under-deliver on all three. This management-integration perspective aligns closely with the coordination argument this paper develops, and it extends that argument from the scheduling and resource-allocation techniques discussed in Sections 2 and 3 to the organizational question of how a cloud provider structures decision-making across teams responsible for performance, cost, and sustainability, which are frequently managed as separate functions with separate incentives even when the underlying technical decisions are deeply interdependent.

A broader systematic mapping of energy-efficiency techniques across more than a hundred primary studies reaches a similar conclusion from a review-level vantage point, finding that virtualization and consolidation, power-aware methods, and thermal-management techniques each contribute measurable but individually partial efficiency gains, with no single technique category dominating across the full space of hyperscale operating conditions surveyed (Bharany et al., 2022). This review-level convergence with Khan et al. (2022) and Ilager et al. (2021), despite covering a different and larger set of primary studies, strengthens the case that the coordination gap this paper identifies is a structural feature of the field's accumulated evidence rather than an artifact of any single research group's framing.

It is worth situating this coordination gap against the major cloud architecture models most prominent in the literature reviewed above. NIST's model (Mell & Grance, 2011) defines cloud computing by its service and deployment characteristics, on-demand self-service, resource pooling, rapid elasticity, but is deliberately silent on how a provider should sequence infrastructure decisions across a platform's operational life; it describes what a cloud platform is, not how its underlying compute fleet should be managed from procurement through retirement. Warehouse-scale computing as Barroso et al. (2013) characterize it goes further by treating the datacenter itself as the unit of design, but its treatment of the operational lifecycle centers on deployment and steady-state operation, with modernization and retirement addressed only implicitly as a background condition of fleet management rather than as a coordinated design phase. The energy-efficiency and sustainability literature, Buyya et al. (2024) most directly, comes closest to this paper's coordination argument by insisting that efficiency emerges from integrated management

rather than isolated technique, but that literature stops short of proposing a phase-structured lifecycle model that names retirement and reclamation as a coordinated stage on equal footing with planning and deployment. Set against these three reference points, general NIST-style service characteristics, warehouse-scale operational design, and integrated sustainability management, the Sustainable Compute Lifecycle Framework proposed here occupies a distinct position: it borrows the operational rigor of warehouse-scale design and the integration principle of the sustainability literature, but applies both across a full five-phase lifecycle that treats hardware retirement and resource reclamation as a first-class coordinated phase rather than an implicit background condition or an afterthought outside the model's scope.

Taken as a whole, the literature reviewed here traces a consistent arc: virtualization established that isolation and performance need not trade off against one another; workload specialization and heterogeneous allocation established that matching resources to workload characteristics is itself an efficiency lever independent of hardware generation; and the more recent resource-management and sustainability literature has converged on the observation that no single technique, however well designed, captures the efficiency available from coordinating multiple techniques together. This paper's contribution is to take that convergent observation and translate it into a lifecycle-structured framework, presented in Section 4, that gives cloud providers a concrete way to organize that coordination across the full operational life of their infrastructure rather than leaving it as a general principle without an operational structure to enact it.

3. ANALYTICAL FRAMEWORK: ARCHITECTURAL TECHNIQUES FOR SUSTAINABLE INFRASTRUCTURE

Several architectural techniques jointly determine how efficiently a hyperscale cloud platform runs. This section treats each in turn before Section 4 synthesizes them into a single lifecycle framework.

Virtualization and Dynamic Resource Allocation

Virtualization remains foundational to multi-tenant efficiency. Modern hypervisors support live migration, dynamic resource allocation, and workload isolation, extending the core capabilities Barham et al. (2003) established two decades ago into environments of far greater scale and heterogeneity today. Live migration in particular lets operators rebalance load without interrupting service, though migration itself consumes resources and can affect concurrent workload performance when scheduled poorly. The efficiency case for virtualization has shifted somewhat since Xen's introduction: where the original motivation was primarily consolidation, fitting more workloads onto fewer physical machines, modern hypervisor capabilities are increasingly valued for the operational flexibility they provide during hardware transitions, letting operators shift workloads off aging hardware ahead of a modernization cycle without service disruption. This flexibility is what connects virtualization, discussed here as a standalone technique, to the hardware modernization discussion later in this section: a platform with mature live-migration capability can execute a modernization cycle far more gradually and with far less workload disruption than one without it, which changes the calculus for how aggressively an operator can pursue workload-aware modernization criteria in the first place.

Workload-Aware and NUMA-Aware Scheduling

Workload-aware scheduling builds on virtualization's efficiency gains by placing applications according to processor topology, memory locality, and networking requirements. NUMA-aware scheduling cuts cross-node memory access latency, a consideration that matters more as server core counts climb and memory access patterns account for a growing share of overall application latency. Evaluations of NUMA-aware scheduling policies in warehouse-scale environments have reported reductions in execution time of up to roughly a quarter relative to default schedulers, alongside a meaningfully higher proportion of tasks placed within their optimal memory domain, which underscores that scheduling intelligence produces measurable efficiency gains independent of any hardware refresh. This is particularly relevant for AI inference workloads, where memory bandwidth and locality often constrain throughput more directly than raw processor clock speed, making scheduling decisions a meaningful lever even when the underlying hardware generation stays fixed. In practice, this means two servers with identical processor

generations and identical nominal specifications can deliver meaningfully different effective throughput for the same inference workload, purely as a function of how memory-aware the scheduler placing that workload happens to be. That gap is invisible to any analysis that treats hardware generation as the primary efficiency variable, which is part of why this paper treats scheduling as a distinct architectural lever rather than folding it into a generic discussion of hardware capability.

Heterogeneous Resource Allocation

Heterogeneous resource allocation has grown correspondingly more important as AI and enterprise workloads increasingly share infrastructure. Marquez and Mondragon (2021) show allocation frameworks that account jointly for storage placement and compute assignment outperform frameworks optimizing each independently, a finding with direct implications for platforms running both traditional enterprise workloads and AI training or inference pipelines side by side. As the range of workload types sharing a single fleet grows, the case for joint rather than siloed allocation logic strengthens correspondingly; a scheduler optimized purely for compute placement risks stranding storage or network capacity that a joint approach would have used more effectively. This has a direct bearing on how cloud providers should think about accelerator deployment specifically: an accelerator-heavy fleet built to serve AI workloads still depends on the same joint compute-storage-network allocation logic as a general-purpose fleet, and treating accelerator placement as a separate allocation problem from the rest of the fleet risks reproducing the siloed-optimization failure mode Marquez and Mondragon (2021) describe, only at the accelerator layer rather than the storage layer. Emerging memory-disaggregation research adds a further layer to this picture: large-scale characterization of CXL-attached memory finds meaningfully higher tail latency than local DRAM, on the order of hundreds of nanoseconds rather than the sub-nanosecond variation typical within a single memory chip (Liu et al., 2024). Allocation logic that treats pooled and local memory as interchangeable without accounting for this latency difference risks placing latency-sensitive workloads onto disaggregated memory where they will underperform, which is itself a heterogeneous-allocation problem in the same family Marquez and Mondragon (2021) describe for compute and storage.

Hardware Modernization Strategy

Hardware modernization strategy has shifted from age-based replacement toward criteria weighing workload demand, operational cost, and performance improvement together. Server and storage hardware typically carries a useful lifecycle of three to five years before replacement becomes cost-effective, though this figure varies considerably by workload intensity and by how aggressively an operator extends hardware life through selective component upgrades. Providers evaluating modernization holistically, rather than replacing hardware on a fixed schedule alone, can extend useful life for underutilized capacity while steering newer hardware toward the workloads that benefit most from it. This selectivity matters more as fleets grow larger and more heterogeneous: a uniform replacement schedule applied across an entire fleet wastes the residual useful life of hardware serving lighter workloads while potentially under-serving compute-intensive workloads that would benefit from earlier replacement. Ahmed et al. (2021) note in their review of data center energy consumption and reliability modeling that IT-load energy consumption, power-conditioning overhead, and cooling-load energy each follow distinct consumption patterns that age-based replacement schedules do not account for individually, which supports treating modernization timing as a workload- and component-specific decision rather than a single fleet-wide rule. Workload-aware modernization criteria also interact directly with the scheduling and allocation techniques discussed above; a fleet with mature workload-aware scheduling can more precisely identify which specific machines are approaching a genuine performance ceiling for their assigned workloads, rather than relying on age as a rough proxy for that ceiling.

Automation as a Cross-Cutting Enabler

Automation cuts the operational overhead tied to provisioning, monitoring, scaling, and retirement. As fleets grow, manual operational processes simply don't scale at the same rate; automated management becomes a practical necessity, not an optional efficiency add-on. Automation's role extends beyond simple task execution; it also supports the kind of continuous, data-driven decision-making that workload-aware scheduling and selective hardware

modernization both depend on, since neither can operate effectively at fleet scale if the underlying monitoring and provisioning processes require constant manual intervention. Recent autoscaling research illustrates this dependency concretely: Chouliaras and Sotiriadis (2023) show that adaptive auto-scaling frameworks calibrated to actual workload behavior outperform static threshold-based provisioning, while Pintye et al. (2024) find that different applications require different monitoring metrics to estimate resource needs accurately, meaning a one-size-fits-all automation policy underperforms a workload-differentiated one even when both are equally automated. A broader review of auto-scaling techniques similarly concludes that machine-learning-based and time-series approaches increasingly outperform simpler threshold-based rules as fleet heterogeneity grows, reinforcing that automation's value depends on how well it is matched to the workload mix it manages, not merely on whether automation exists at all (Alharthi et al., 2024). Automation is therefore best understood not as a fifth technique alongside the four above but as a cross-cutting enabler that determines how consistently and how quickly the other four can actually be applied across a fleet of meaningful size; a workload-aware scheduling policy that depends on manual monitoring to trigger rebalancing decisions will, in practice, rebalance far less often than fleet scale would ideally require.

The four techniques reviewed above, virtualization, workload-aware scheduling, heterogeneous resource allocation, and hardware modernization, are typically evaluated and improved one at a time in the literature, and each shows measurable gains when optimized on its own terms. But optimizing each independently understates what is available, because the techniques interact: a scheduling policy tuned in isolation from modernization criteria cannot exploit the flexibility that mature live-migration capability provides during a hardware transition, and a modernization policy tuned in isolation from workload-aware scheduling data has no reliable way to identify which specific machines are genuinely approaching a performance ceiling rather than merely aging on a fixed calendar. Marquez and Mondragon's (2021) finding that joint compute-storage allocation outperforms allocating each independently is a narrow instance of a broader pattern this paper argues holds across all four techniques: the efficiency lost to siloed optimization is unlikely to stay confined to a single layer, so that a platform optimizing virtualization, scheduling, allocation, and modernization as four separate problems plausibly captures less total efficiency than one that treats their interactions as part of the optimization target from the start. This is the reasoning that motivates treating integration itself, not any single technique, as the primary lever, and it is the premise Section 4 builds on in proposing a framework that coordinates all four techniques, together with automation as the enabler that makes such coordination operationally feasible, across a platform's full lifecycle rather than within any single phase alone.

4. PROPOSED SUSTAINABLE COMPUTE LIFECYCLE FRAMEWORK AND DISCUSSION

Figure 1. Sustainable Compute Lifecycle Framework

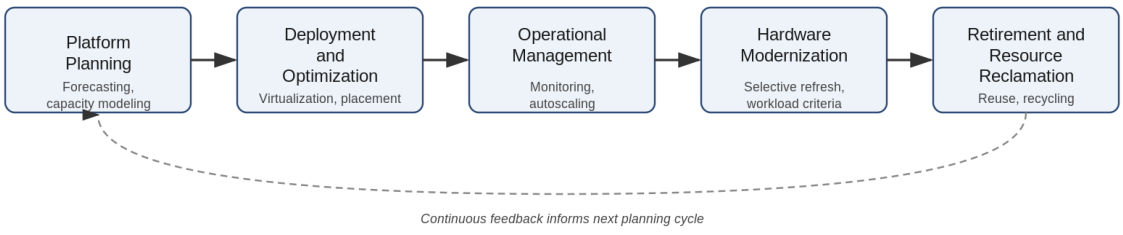

Figure 1. Sustainable Compute Lifecycle Framework. (Author's original synthesis based on the literature reviewed in Sections 2 and 3.)

Pulling together the architectural techniques reviewed above, this paper proposes the Sustainable Compute Lifecycle Framework, a model for coordinating infrastructure decisions across a platform's operational life rather than treating planning, deployment, and retirement as separate, uncoordinated activities.

Five interconnected phases make up the framework. Platform Planning uses workload forecasting and capacity modeling to guide hardware selection before deployment starts; the quality of forecasting at this stage constrains every phase that follows, since a platform provisioned against an inaccurate demand forecast cannot be fully corrected through operational tuning later. Deployment and Optimization applies virtualization and workload-aware placement to maximize utilization from day one, drawing directly on the scheduling and allocation techniques discussed in Section 3. Operational Management sustains that efficiency through continuous monitoring, autoscaling, and automation, consistent with the resource-management approaches both Khan et al. (2022) and Ilager et al. (2021) describe as increasingly central to cloud operations. Hardware Modernization introduces newer processor generations selectively, guided by the workload-driven criteria discussed in Section 3.4 rather than a fixed replacement calendar. Retirement and Resource Reclamation closes the cycle, removing legacy infrastructure in a structured way that lets datacenter resources be repurposed rather than simply decommissioned.

Because this framework is presented here as a conceptual synthesis rather than as the output of an empirical deployment, the discussion that follows characterizes expected operational benefits rather than measured results. Applied consistently, the framework's coordination logic points toward four benefits that follow directly from the interdependencies described above rather than from any single novel technique. Coordinating Platform Planning with Hardware Modernization should reduce the incidence of premature replacement, since modernization decisions informed by workload-aware scheduling data can distinguish hardware genuinely approaching a performance ceiling from hardware merely aging on a fixed calendar, the same distinction Ahmed et al. (2021) identify as missing from age-based replacement schedules. Coordinating Deployment and Optimization with Operational Management should shorten the gap between a capacity plan's assumptions and a fleet's observed workload behavior, consistent with the continuous-refinement role AI-assisted scheduling plays in Section 5. Extending live-migration capability from a deployment-time convenience into a deliberate enabler of gradual hardware transitions should reduce the service disruption typically associated with modernization cycles. Finally, treating Retirement and Resource Reclamation as a planned phase rather than an end-of-life afterthought should improve component recovery and reuse rates along the lines Microsoft's Circular Centers program demonstrates operationally, since standardization decisions made earlier in the lifecycle directly affect how efficiently components can later be recovered. None of these four expected benefits has been validated against operational telemetry in this paper; they follow from the coordination logic the framework formalizes and from the individual technique-level findings reviewed in Sections 2 and 3, and validating them empirically against a deployed fleet is the natural next step this paper's synthesis calls for.

Table 1. Comparison of Age-Based and Framework-Based Hardware Modernization Decision Criteria

Dimension	**Age-Based Approach**	**Framework-Based Approach**
Trigger for replacement	Fixed time interval (e.g., 3-5 years)	Workload demand, operational cost, and performance improvement, evaluated jointly
Treatment of underutilized hardware	Replaced regardless of utilization	Extended in service if still meeting workload needs
Coordination with scheduling	Independent decision	Informed by workload-aware scheduling data identifying genuine performance ceilings
Coordination with retirement/reclamation	Retirement treated as separate end-of-life step	Retirement and component reuse planned jointly with modernization timing

This lifecycle framing differs from conventional hardware-refresh planning in one important respect: modernization stops being an isolated procurement decision and becomes one stage in a continuous optimization process running from initial capacity planning through resource reclamation. Seen this way, the efficiency gains available at any single stage are bounded by decisions made earlier; a poorly planned platform can't be fully corrected

through operational tuning alone, and a platform modernized without workload-aware criteria risks replacing hardware that didn't need it while under-serving workloads that did. This interdependence across phases is precisely what distinguishes a lifecycle framework from a collection of independently optimized best practices: the value of getting Hardware Modernization right, for instance, is partly determined by how well Platform Planning anticipated the workload mix that modernization now has to accommodate. The same logic runs in the other direction too: a Platform Planning process that assumes an idealized, frictionless modernization cycle will systematically underestimate the operational cost of executing that modernization later, if the Deployment and Optimization phase never built the live-migration and workload-aware placement capability modernization actually depends on.

Table 2. Qualitative Directional Shift in Efficiency-Gain Sources, per Literature Reviewed

Period (approximate)	Dominant Efficiency-Gain Source (per literature reviewed)
2003-2010	Virtualization-driven consolidation; isolation without performance loss
2011-2020	Workload-specialized VM families; early energy-aware scheduling and consolidation surveys
2021-2023	Joint/heterogeneous resource allocation; industry-level empirical energy-efficiency evidence
2022-2025	Data-driven and ML-centric resource management; adaptive autoscaling; integrated sustainability management
Emerging	Composable/disaggregated memory (CXL); AI-assisted scheduling

Table 3. Architectural Techniques Summary

Technique	Primary Efficiency Mechanism	Infrastructure Layer Affected
Virtualization / live migration	Multi-tenant consolidation, workload rebalancing without downtime	Hypervisor / compute layer
Workload-aware / NUMA-aware scheduling	Reduced cross-node memory latency, improved effective throughput	Scheduler / compute-memory interface
Heterogeneous resource allocation	Joint optimization across compute, storage, and network resources	Cross-layer (compute, storage, network)
Hardware modernization (workload-aware)	Selective refresh targeting genuine performance ceilings	Physical hardware layer
Automation / adaptive autoscaling	Reduces manual overhead, enables continuous data-driven decisions	Cross-cutting (all layers)

The framework's practical value comes from this coordination. It makes explicit that virtualization efficiency, scheduling intelligence, and modernization discipline are interdependent, not separable levers, a conclusion that echoes the joint-optimization findings Marquez and Mondragon (2021) report at the resource-allocation level, extended here across the full infrastructure lifecycle. Retirement and Resource Reclamation, the framework's final phase, is not merely an end-of-life formality; structured component reuse and recycling programs demonstrate that this phase can itself generate measurable operational value. Microsoft's Circular Centers program, for example, reported a 90.9% reuse and recycling rate for servers and components in 2024, exceeding its own 90% target roughly a year ahead of schedule, with more than 3.2 million components reused during the year (Microsoft, 2024). A retirement phase treated as an afterthought forfeits this kind of value; a retirement phase planned as part of the same coordinated framework governing earlier lifecycle stages can capture it deliberately, since decisions made during Platform Planning and Hardware Modernization about which components are standardized and how consistently they

are deployed across the fleet directly affect how efficiently those same components can later be recovered, refurbished, and redeployed at the Retirement and Resource Reclamation stage.

Viewed across all five phases together, the framework's central claim is not that any individual phase introduces a novel technique the literature has not already documented in some form, but that treating the five phases as a coordinated sequence, rather than as five separately optimized problems, surfaces efficiency opportunities that phase-by-phase optimization misses. This is consistent with the broader pattern observed across the literature reviewed in Section 2: Khan et al. (2022) and Ilager et al. (2021) both note that no single resource-management technique addresses efficiency comprehensively, and Marquez and Mondragon (2021) demonstrate that even within a single lifecycle phase, joint optimization across resource types outperforms independent optimization. The lifecycle framework proposed here extends that same joint-optimization logic from within a single phase to across all five phases of a platform's operational life.

The framework also has implications for how organizations structure the teams responsible for infrastructure efficiency, a point that follows directly from Buyya et al.'s (2024) observation that energy efficiency emerges from integrated management rather than isolated optimization. If Platform Planning, Deployment and Optimization, Operational Management, Hardware Modernization, and Retirement and Resource Reclamation are owned by separate teams with separate performance metrics and no shared visibility into how decisions at one phase affect outcomes at another, the coordination benefits this framework identifies are unlikely to materialize regardless of how well each individual team executes its own phase. Conversely, an organization that gives a single accountable function visibility across all five phases, even if execution remains distributed across specialized teams, is better positioned to capture the interdependencies this section has described: the connection between Platform Planning's forecast quality and Hardware Modernization's ability to apply workload-aware criteria, or between Deployment and Optimization's live-migration maturity and the operational cost of executing a modernization cycle later. This organizational implication is a natural extension of the framework's technical argument rather than a separate claim requiring independent justification, since a framework built around cross-phase interdependency has little practical value if no part of the organization is positioned to observe or act on that interdependency.

5. FUTURE DIRECTIONS

The three directions discussed in this section, composable infrastructure, AI-assisted scheduling, and structured hardware reuse, share a common feature worth making explicit before turning to each individually: none of them replaces the coordination logic developed in Sections 3 and 4, and each instead extends that logic into new technical territory that did not exist, or existed only in early form, when the foundational virtualization and scheduling literature reviewed in Section 2 was first developed. This continuity matters for how the framework proposed in this paper should be read going forward. A framework built purely around today's specific techniques, virtualization, NUMA-aware scheduling, age-independent modernization criteria, would risk becoming outdated as those specific techniques are superseded by newer ones. A framework built instead around the coordination principle those techniques currently instantiate is more durable, since new techniques such as composable memory pooling or AI-assisted scheduling can be incorporated into the same five-phase structure without requiring the framework itself to be redesigned.

Composable infrastructure, enabled by interconnects such as Compute Express Link, extends the coordination principle behind the proposed framework considerably. CXL lets memory resources be pooled and dynamically allocated across servers rather than statically bound to individual machines (Das Sharma et al., 2024), and recent surveys trace CXL's evolution from a memory-expansion technology toward more fine-grained, composable resource-sharing architectures (Chen et al., 2024). For AI workloads with large and variable memory footprints, that flexibility extends the framework's Deployment and Optimization phase directly, decoupling memory capacity planning from individual server procurement decisions. Where a server's memory capacity has historically been fixed at the point of procurement, forcing operators to provision for peak memory demand on every machine, composable memory pooling allows capacity to follow demand instead, which has direct implications for the Platform Planning phase as well: capacity models no longer need to assume static per-server memory ceilings, which in turn changes what an accurate

demand forecast at the planning stage actually needs to predict. Rather than forecasting peak memory demand per server, planners under a composable-memory model need to forecast aggregate memory demand across a pool, a materially different and in some respects more tractable forecasting problem.

AI-assisted infrastructure management is a second significant direction. Sanjalawe et al. (2025) reviewed AI-driven approaches to cloud job scheduling and found machine learning and hybrid optimization techniques increasingly outperforming static heuristic scheduling, particularly under the variable, multi-tenant workload conditions typical of AI inference and training pipelines. Applied within the lifecycle framework proposed here, predictive scheduling mainly strengthens the Operational Management phase, letting capacity decisions made during planning get continuously refined against observed workload behavior rather than remaining fixed until the next planning cycle. This continuous-refinement capability is particularly valuable given the workload-estimation bottleneck Khan et al. (2022) identify: if demand forecasting at the planning stage is imperfect, an operationally adaptive scheduling layer can partially compensate by adjusting placement decisions as actual workload behavior diverges from the original forecast. It is worth being precise about the limits of this compensation, however: an adaptive scheduling layer can correct for forecasting error within the capacity envelope that Platform Planning already provisioned, but it cannot conjure capacity that was never provisioned in the first place. AI-assisted scheduling therefore strengthens the framework's resilience to forecasting error without eliminating the underlying importance of getting Platform Planning right.

A further, more speculative extension of AI-assisted infrastructure management concerns agentic AI specifically, as distinct from the predictive and classification-style machine learning Sanjalawe et al. (2025) survey above. The literature reviewed in this paper does not yet address agentic AI's infrastructure implications directly, so what follows is this paper's own extrapolation from the coordination logic developed above rather than a synthesis of established findings, and it should be read with that caveat. Agentic AI systems, built around large language models that plan multi-step actions and invoke tools or APIs to carry them out, plausibly introduce reasoning workloads with a resource profile that differs from the training and inference workloads this paper has otherwise discussed: an agentic workload's compute demand can be expected to depend on how many reasoning steps a task requires and how many tool calls it issues, which would vary with task complexity in a way that conventional batch or request-response inference patterns do not. If that expectation holds, it would carry implications for the framework's Operational Management phase, since capacity sized for average agentic workload behavior could be poorly matched to a minority of unusually complex, many-step tasks. Orchestration is a second dimension worth flagging speculatively: an agentic system may invoke multiple specialized models, retrieval systems, and external tools within a single task, which would place additional demands on the general-purpose compute layer's ability to orchestrate heterogeneous, short-lived workloads, extending the orchestration burden this paper has argued grows as accelerator diversity increases. To the extent this pattern materializes at scale, it would also reinforce the case for heterogeneous compute specifically, since a single agentic task combining large-model inference, retrieval, and conventional application logic would span the compute-storage-network interactions Marquez and Mondragon (2021) argue must be allocated jointly rather than independently. These are reasoned extensions of the coordination argument this paper develops, not conclusions drawn from an agentic-AI-specific evidence base, and confirming them would require the kind of workload characterization studies that exist for training and inference workloads but do not yet exist, to this paper's knowledge, for agentic systems specifically.

Sustainability considerations will likely shape hardware modernization practice further still. The scale of Microsoft's component reuse program, more than 3.2 million components recovered in a single year (Microsoft, 2024), indicates that structured retirement practices can operate at a scale relevant to hyperscale fleets, not merely as a pilot-scale sustainability initiative. As more providers formalize retirement and reclamation processes along these lines, the Retirement and Resource Reclamation phase of frameworks like the one proposed here may increasingly be evaluated on its own efficiency terms, alongside the more commonly scrutinized Deployment and Operational Management phases. A plausible trajectory, consistent with the coordination argument developed throughout this paper, is that component-level standardization decisions made during Platform Planning and Hardware Modernization will

increasingly be evaluated partly on how well they support efficient recovery and reuse later, rather than purely on their upfront performance and cost characteristics, effectively extending the framework's coordination logic to include reclamation-readiness as a design consideration from the earliest lifecycle stage onward.

Several practical implications follow from the analysis developed in this paper for organizations operating or planning hyperscale infrastructure. First, capacity planning processes should be evaluated not only on forecast accuracy in isolation but on how well that forecast translates into modernization and scheduling decisions downstream; a highly accurate forecast that is not operationally connected to modernization criteria and workload-aware scheduling policy captures less value than a moderately accurate forecast that is well integrated with both. Second, organizations investing in accelerator capacity for AI workloads should evaluate that investment jointly with the general-purpose compute layer's orchestration capacity, rather than treating accelerator procurement as a standalone decision, since the coordination burden documented throughout this paper falls disproportionately on the general-purpose layer as accelerator diversity increases. Third, retirement and reclamation practices merit earlier and more deliberate design attention than they conventionally receive; treating reclamation readiness as a consideration only once hardware nears end of life forecloses efficiency gains that earlier, coordinated component-standardization decisions could otherwise have preserved.

These implications are necessarily general, since the specific weighting of each lifecycle phase's importance will vary by provider scale, workload mix, and organizational structure. A smaller provider operating a narrower range of workload types may find that the coordination benefits documented here are achievable through less formal mechanisms than a hyperscale operator running a highly heterogeneous fleet across compute-intensive, memory-intensive, and accelerator-heavy workload categories simultaneously. The lifecycle framework proposed in this paper is intended as a structuring device for reasoning about that coordination problem, not as a prescriptive specification that applies identically regardless of scale or workload composition.

LIMITATIONS

This paper's contribution is conceptual synthesis rather than empirical validation, and that scope carries several limitations worth stating explicitly. First, the Sustainable Compute Lifecycle Framework has not been tested against operational telemetry from a deployed fleet; the expected benefits described in Section 4 follow from the coordination logic the framework formalizes and from technique-level findings reported elsewhere in the literature, not from measurement of the framework's own performance in production. Second, the framework synthesizes findings drawn primarily from literature addressing large hyperscale operators, and the coordination mechanisms it describes, particularly the organizational recommendation that a single accountable function maintain visibility across all five phases, may be harder to justify or implement for smaller providers whose infrastructure and organizational scale do not generate the same coordination costs that motivate the framework at hyperscale. Third, the paper's treatment of emerging directions, composable infrastructure, AI-assisted scheduling, and agentic AI workloads, draws on early-stage research and survey literature rather than mature, extensively validated deployment experience, since these technologies are themselves still evolving; conclusions about how well the framework accommodates them should be treated as provisional pending further practical experience. Fourth, this paper does not quantify the magnitude of the efficiency gains coordination should be expected to deliver relative to independent optimization; it argues for the direction and mechanism of that gain rather than its size, and closing that gap requires the kind of operational validation the paper identifies as a natural next step. These limitations do not undermine the coordination argument itself, which rests on convergent findings across the technique-level literature reviewed in Sections 2 and 3, but they do mean the framework should be read as a structuring device for organizing that argument rather than as a validated operational specification.

6. CONCLUSION

Sustainable cloud infrastructure growth takes more than processor and accelerator performance gains. As AI and enterprise workloads keep expanding, coordinated architectural decisions across virtualization, workload-aware scheduling, heterogeneous resource allocation, and hardware lifecycle management determine whether infrastructure

efficiency keeps pace with demand. This paper traced the evolution of general-purpose cloud compute architecture and proposed the Sustainable Compute Lifecycle Framework as a model for coordinating infrastructure decisions across a platform's full operational life, rather than treating planning, operation, and retirement as separate concerns. Emerging directions, composable infrastructure enabled by CXL, AI-assisted scheduling, and structured hardware reuse, extend this coordination principle rather than replace it. Cloud providers pursuing sustainable growth under continued AI-driven demand will likely find that architectural coordination, applied consistently across the compute lifecycle, delivers efficiency gains that hardware expansion alone cannot.

ETHICS STATEMENT

Conflict of Interest: The author declares no conflict of interest.Statement of Human and Animal Rights: This study did not involve human participants, animal subjects, or identifiable personal data requiring ethics board review.Informed Consent: Not applicable.

CREDIT AUTHOR CONTRIBUTIONS

Priyadarshni Shanmugavadivelu: Conceptualization, Investigation, Writing - Original Draft, Writing - Review and Editing.

REFERENCES

1. Ahmed, K. M. U., Bollen, M. H. J., & Alvarez, M. (2021). A review of data centers energy consumption and reliability modeling. IEEE Access, 9, 152536-152563. https://doi.org/10.1109/ACCESS.2021.3125092
2. Bharany, S., Sharma, S., Khalaf, O. I., Abdulsahib, G. M., Al Humaimeedy, A. S., Aldhyani, T. H. H., Maashi, M., & Alkahtani, H. (2022). A systematic survey on energy-efficient techniques in sustainable cloud computing. Sustainability, 14(10), 6256. https://doi.org/10.3390/su14106256
3. Alharthi, S., Alshamsi, A., Alseiari, A., & Alwarafy, A. (2024). Auto-scaling techniques in cloud computing: Issues and research directions. Sensors, 24(17), 5551. https://doi.org/10.3390/s24175551
4. Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Lee, G., Patterson, D. A., Rabkin, A., Stoica, I., & Zaharia, M. (2010). A view of cloud computing. Communications of the ACM, 53(4), 50-58. https://doi.org/10.1145/1721654.1721672
5. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., & Warfield, A. (2003). Xen and the art of virtualization. Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03), 164-177. https://doi.org/10.1145/945445.945462
6. Barroso, L. A., Clidaras, J., & Holzle, U. (2013). The datacenter as a computer: An introduction to the design of warehouse-scale machines (2nd ed.). Morgan & Claypool Publishers. https://doi.org/10.2200/S00516ED2V01Y201306CAC024
7. Buyya, R., Ilager, S., & Arroba, P. (2024). Energy-efficiency and sustainability in new generation cloud computing: A vision and directions for integrated management of data centre resources and workloads. Software: Practice and Experience, 54(1), 24-38. https://doi.org/10.1002/spe.3248
8. Chen, C., Zhao, X., Cheng, G., Xu, Y., Deng, S., & Yin, J. (2024). Next-gen computing systems with Compute Express Link: A comprehensive survey (arXiv:2412.20249). https://doi.org/10.48550/arXiv.2412.20249
9. Chouliaras, S., & Sotiriadis, S. (2023). An adaptive auto-scaling framework for cloud resource provisioning. Future Generation Computer Systems, 148, 173-183. https://doi.org/10.1016/j.future.2023.05.017
10. Das Sharma, D., Blankenship, R., & Berger, D. S. (2024). An introduction to the Compute Express Link (CXL) interconnect. ACM Computing Surveys, 56(11), Article 290. https://doi.org/10.1145/3669900
11. Ilager, S., Muralidhar, R., Rameshan, N., & Buyya, R. (2021). Recent advances in energy efficient resource management techniques in cloud computing environments (arXiv:2107.06005). https://arxiv.org/abs/2107.06005
12. Kachris, C. (2025). A survey on hardware accelerators for large language models. Applied Sciences, 15(2), 586. https://doi.org/10.3390/app15020586
13. Khan, T., Tian, W., Zhou, G., Ilager, S., Gong, M., & Buyya, R. (2022). Machine learning (ML)-centric resource management in cloud computing: A review and future directions. Journal of Network and Computer Applications, 204, Article 103405. https://doi.org/10.1016/j.jnca.2022.103405
14. Liu, J., Hadian, H., Xu, H., Berger, D. S., & Li, H. (2024). Dissecting CXL memory performance at scale: Analysis, modeling, and optimization (arXiv:2409.14317). https://arxiv.org/abs/2409.14317
15. Marquez, J., & Mondragon, O. H. (2021). An intelligent approach to resource allocation on heterogeneous cloud infrastructures. Applied Sciences, 11(21), 9940. https://doi.org/10.3390/app11219940
16. Mell, P., & Grance, T. (2011). The NIST definition of cloud computing (NIST Special Publication 800-145). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-145

17. Microsoft. (2024). Learn how Microsoft Circular Centers are scaling cloud supply chain sustainability. Microsoft Azure Blog. https://azure.microsoft.com/en-us/blog/learn-how-microsoft-circular-centers-are-scaling-cloud-supply-chain-sustainability/
18. Park, J., Han, K., & Lee, B. (2023). Green cloud? An empirical analysis of cloud computing and energy efficiency. Management Science, 69(3), 1639-1664. https://doi.org/10.1287/mnsc.2022.4442
19. Pintye, I., Kovacs, J., & Lovas, R. (2024). Enhancing machine learning-based autoscaling for cloud resource orchestration. Journal of Grid Computing, 22(4), 31. https://doi.org/10.1007/s10723-024-09783-1
20. Sanjalawe, Y., Al-E'mari, S., Fraihat, S., & Makhadmeh, S. (2025). AI-driven job scheduling in cloud computing: A comprehensive review. Artificial Intelligence Review. https://doi.org/10.1007/s10462-025-11208-8
21. Uptime Institute. (2024). Uptime Institute global data center survey results 2024. Uptime Institute. https://uptimeinstitute.com/resources/research-and-reports/uptime-institute-global-data-center-survey-results-2024
22. Verma, P., Kumar, R., & Sharma, A. (2024). A survey on energy-efficient workflow scheduling algorithms in cloud computing. Software: Practice and Experience. https://doi.org/10.1002/spe.3292