

ASTRA : Advanced Storage Tiering, Resilience, and Analytics Framework

Dr. Arvind A R¹, Raviteja Egalapati², Santhosh R D³, Praveen P V⁴

^{1,2} Ashok Leyland, Chennai, Tamil Nadu, India

^{3,4} Coimbatore Institute of Technology, Coimbatore, Tamil Nadu, India

Abstract: Industrial and manufacturing computing environments rely heavily on distributed virtualized enterprise topologies running critical database systems, enterprise resource planning (ERP), and manufacturing execution pipelines. A critical point of failure in these deployments is the resilience and availability of the underlying backup repository ecosystem. Ad-hoc administrative storage management and manual capacity inspections create systemic risks, including silent storage exhaustion, unmonitored backup failure cascades, and unmitigated capacity bottlenecks. This paper presents the architecture, theoretical foundation, mathematical modeling, and production deployment of an autonomous storage analytics, multi-tier capacity forecasting, and notification platform. Engineered across geographically isolated plant nodes (A, B, C, D, and E), the proposed system leverages agentless, low-dependency CIM/WMI hardware polling coupled with recursive filesystem discovery. The platform integrates a rolling-window statistical ledger, automated schema persistence, and structured HTML/Excel dispatch protocols over encrypted transport layer security (TLS). Furthermore, to counter the degradation of standard linear regression caused by non-organic data operations (such as manual inter-tier storage migrations and retention overrides), we formulate an aggregate pooling abstraction and a rolling median smoothing mechanism. The empirical deployment demonstrates low runtime compute overhead (1.15% peak CPU, 26.4 MB memory) and substantially reduces manual, per-server capacity checks across all five plants. In a representative validation case involving an unplanned 420 GB inter-tier migration, the proposed macro-pooling and median-filtering approach reduced the seven-day capacity-forecast RMSE from 482.4 GB (standard linear regression) to 4.12 GB, illustrating the model's resilience to administrative storage-migration noise.

Keywords: Virtual Machine Backup, Storage Capacity Forecasting, Enterprise Systems, CIM/WMI Architecture, System Automation, Predictive Analytics, Disaster Recovery

Background and Industry Context

Backup infrastructure has quietly become one of the most operationally critical layers of modern manufacturing enterprises. As plant-floor systems, ERP platforms, and manufacturing execution pipelines have digitized, the volume of data that must be protected against loss, corruption, or ransomware has grown far faster than the administrative bandwidth available to manage it. A single unmonitored storage volume reaching full capacity can silently halt backup jobs across an entire facility, leaving the organization without a recoverable restore point for hours or days without anyone being aware of the gap. In industries where unplanned downtime translates directly into lost production, this is not a theoretical risk but a recurring operational exposure.

In practice, the day-to-day stewardship of these repositories is still largely manual. Administrators log into individual servers, read free-space figures off a console, and transcribe them into spreadsheets on a best-effort basis. This approach does not scale across geographically distributed plants, does not capture the rate



at which storage is being consumed, and offers no advance warning before a volume crosses a critical threshold. It also breaks down entirely when routine administrative actions occur—such as manually relocating old backup images between local and SAN storage tiers to free up space or service a controller—because these actions look, to any naive trend line, indistinguishable from genuine runaway growth. The result is either alert fatigue from false alarms or, worse, a false sense of security that leaves genuine capacity exhaustion undetected until a backup job actually fails.

These recurring operational pain points—manual inspection overhead, inconsistent visibility across plants, and forecasting models that break down in the presence of routine administrative activity—motivated the present study. Rather than treating capacity monitoring as a simple threshold check, this work set out to build a system that could observe storage behavior continuously, tell apart organic data growth from administrative noise, and give plant operations teams enough advance notice to act before a repository becomes a point of failure. The remainder of this paper details the architecture, mathematical modeling, and production deployment developed to address this gap.

I. INTRODUCTION

Enterprise industrial operations have experienced an exponential expansion in digitized telemetry, transactional database logging, and automated execution workflows. In discrete manufacturing plants, modern systems orchestrate high-availability Microsoft SQL Server database clusters, Active Directory domain controller federations, and critical virtual machines (VMs) running on hypervisor environments. Maintaining continuous operational readiness necessitates rigorous Business Continuity and Disaster Recovery (BCDR) frameworks. These frameworks mandate structured, recurring backup operations characterized by grandfather-father-son (GFS) retention matrices, immutable daily image backups, and transactional log checkpoints.

While contemporary backup engines effectively execute scheduled virtual disk snaps and image consolidation, the underlying physical and storage area network (SAN) storage fabrics remain susceptible to resource exhaustion. Backup repositories ingest large quantities of modified blocks daily, exhibiting variable delta rates driven by production batch jobs, transactional commits, and database indexing operations. Storage exhaustion on a target repository causes sudden failure of running backup jobs, snapshot commit timeouts, and potential production hypervisor pauses due to shared storage saturation.

Building on the industry context above, the specific technical shortcomings of manual repository health checks across geographically distributed plants can be broken down as follows:

- Systems administrators authenticate sequentially into individual node management planes.
- Free and used disk parameters are manually transcribed into unstandardized spreadsheets.
- Growth tracking remains qualitative rather than quantitative, obscuring sudden delta surges.
- Ad-hoc administrative interventions—such as manually relocating restore points from SAN tiers to local block storage—introduce statistical anomalies that distort traditional linear capacity forecasting.

To resolve these systemic operational and theoretical deficiencies, this research presents an end-to-end framework for autonomous backup repository monitoring, dynamic filesystem harvesting, self-pruning schema persistence, and multi-tier capacity growth modeling. The system was designed, validated, and deployed across production plant nodes within enterprise infrastructures (spanning the plants A, B, C, D, and E).

The contributions of this paper are organized as follows:

1. Agentless Architectural Design: A low-overhead polling infrastructure utilizing Common Information Model (CIM) interfaces to decouple storage metric collection from hypervisor vendor dependencies.
2. Recursive Schema-Agnostic File Parsing: A dynamic structural parser capable of traversing complex directory hierarchies while maintaining flat-file tabular integrity across variable array keys.



3. Mathematical Formulation of Migration-Resistant Projections: A multi-scenario capacity forecasting model that isolates administrative tier-shifting anomalies from organic daily ingestion rates using macro-pooling and rolling median filters.

4. Autonomous Enterprise Dispatch: An automated reporting system delivering localized CSS-styled, client-agnostic vertical status matrices and multi-column temporal Excel analytics via encrypted Simple Mail Transfer Protocol (SMTP).

II. RELATED WORK AND THEORETICAL CONTEXT

A. Enterprise Storage Telemetry and Monitoring Architectures

Traditional enterprise storage monitoring typically relies on Simple Network Management Protocol (SNMP) polling [20] or heavy vendor-proprietary monitoring daemons. SNMP architectures, while ubiquitous, often fail to provide fine-grained, folder-level backup job introspection without extensive Management Information Base (MIB) customization. Proprietary agent-based platforms introduce non-negligible CPU and memory overhead, often competing directly with high-priority backup serialization tasks during overnight maintenance windows.

Alternative lightweight approaches leverage native operating system instrumentation. The Common Information Model (CIM), standardized by the Distributed Management Task Force (DMTF) [1], and Windows Management Instrumentation (WMI) [2] provide direct, low-latency hooks into physical volume management. By abstracting the storage layer via query interfaces (e.g., Win32_LogicalDisk), system states can be queried without running background daemon processes, thereby preserving volatile system resources.

B. Backup Retention Dynamics and Storage Degradation

Backup datasets do not scale monotonically. They oscillate based on retention policy schedules, deduplication block references [16], [17], and synthetic full-backup compilations. Under a GFS rotation:

- Daily incrementals represent daily write deltas.
- Weekly synthetics aggregate changed blocks into consolidated baseline virtual disks.
- Monthly archives retain deep restore points for compliance mandates.

This oscillatory behavior invalidates naïve first-order difference calculations. When a retention window expires, a batch deletion event occurs, resulting in sudden negative deltas. A production monitoring architecture must differentiate between scheduled storage recovery (retention enforcement) and administrative storage evacuation (manual folder migration).

C. Capacity Forecasting, Robust Statistics, and Predictive-Maintenance Context

Foundational studies on distributed and cloud storage architectures [4]–[6] and on long-term storage-hardware reliability [7], [8] establish that both software-defined storage layers and the physical media beneath them exhibit predictable, measurable failure and consumption patterns rather than emergent randomness; the CIM/WMI polling layer used in this work builds on the same abstraction principle at the guest-OS level. Because the capacity figures reported in this study are guest-OS logical measurements, they are also subject to the divergence between logical and physical consumption documented extensively in the storage-deduplication literature [16]–[18]; this divergence is revisited as a limitation in Section VI-B.

The rolling-median filtering mechanism introduced in Section III-B2 draws on classical robust-statistics theory: the median as a location estimator resistant to gross outliers [14], M-estimation [12], least-median-of-squares regression [13], and robust regression procedures for outlier rejection more broadly [15]. Standard ordinary-least-squares forecasting, by contrast, is known to be highly sensitive to a small number of extreme observations [11], which is precisely the failure mode this paper documents for administrative storage migrations.

Time-series and machine-learning approaches to storage and cloud-resource capacity forecasting have been studied extensively in centralized data-center settings [27]–[30], typically using ARIMA, exponential smoothing, or deep



sequence models trained on months of continuous telemetry; disk-level failure predictors extend this data-driven philosophy to hardware reliability [19], [34]. The broader Industry 4.0 predictive-maintenance literature [23]–[26] similarly argues for replacing calendar-based interventions with data-driven scheduling, a philosophy this paper adopts for storage-capacity management specifically. Statistical and learning-based anomaly-detection surveys in the AIOps domain [21], [22] describe general techniques for flagging irregular system behavior; the greater-than-three-standard-deviation migration-detection heuristic proposed in Section VI-B follows this family of approaches, though a full anomaly-detection evaluation is left to future work (Section VI-E). Prior patent art has separately proposed storage-capacity forecasting for active storage tiers [33] and machine-learning-based disk-failure prediction [34]; the present work differs primarily in combining migration-aware macro-pooling with lightweight, agentless telemetry suited to distributed manufacturing plants rather than centralized cloud data centers. Enterprise virtualization and disaster-recovery practice guides [31], [32] frame the operational BCDR(Business Continuity and Disaster Recovery) context in which the system described in Section IV was deployed. The novelty of the present work is therefore not claimed to lie in CIM(Common Information Model)/WMI(Windows Management Instrumentation) polling, median filtering, or linear forecasting individually — each is well established in isolation — but in their integration into a migration-aware, multi-tier capacity-forecasting framework validated across distributed manufacturing plant infrastructure.

III. SYSTEM FRAMEWORK AND MATHEMATICAL MODELING

The monitoring framework operates on a modular four-tier pipeline: Hardware Telemetry Extraction, Recursive Filesystem Traversal, Historical State Serialization, and Analytical Presentation.

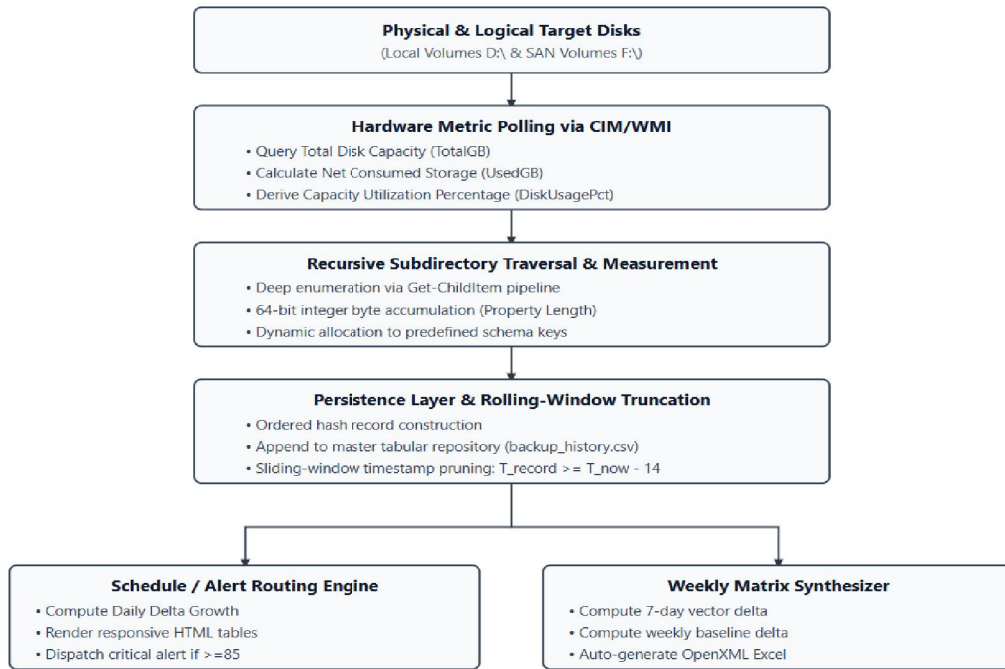


Fig. 1. Four-tier monitoring and analytics pipeline framework.

A. Mathematical Formulation of Storage Utilization and Capacity Growth

Let the monitored system contain a set of physical disk nodes $D = \{d_1, d_2, \dots, d_n\}$, where each disk d_i possesses a total storage capacity $C(d_i) \in \mathbb{R}^+$ and a free space metric $F(d_i, t) \in \mathbb{R}^+$ recorded at discrete calendar timestamp $t \in \mathbb{Z}^+$. The consumed storage volume $U(d_i, t)$ is defined by:



$$U(d_i, t) = C(d_i) - F(d_i, t) \quad (1)$$

The instant capacity utilization ratio $\rho(d_i, t)$ is given by:

$$\rho(d_i, t) = (U(d_i, t) / C(d_i)) \times 100 \quad (2)$$

A systemic threshold boundary condition defines the state transition into a critical warning regime:

$$W(d_i, t) = 1 \text{ if } \rho(d_i, t) \geq \theta_{crit}; \quad 0 \text{ otherwise} \quad (3)$$

where $\theta_{crit} = 85.0\%$ was adopted as an operational warning threshold based on the enterprise's predefined capacity-management policy, consistent with conventional industry guidance that reserves the remaining headroom for burst ingestion, snapshot overhead, and emergency remediation before hypervisor or SAN-controller stability is affected [7], [8]. This value is a policy parameter rather than an empirically derived instability boundary; Section VI-E discusses the absence of a per-plant sensitivity analysis as a limitation.

For any specific directory or backup leaf node $k \in K_{d_i}$ hosted on disk d_i , the physical footprint is computed as the sum of all constituent file sizes:

$$S(k, t) = \sum_{j=1 \text{ to } N(k)} \text{Length}(f_{k,j}) \quad (4)$$

where $f_{k,j}$ denotes the j -th file entity contained within the directory tree rooted at k .

The finite backward growth difference $\Delta S(k, t)$ over interval $\tau = 1$ day is formulated as:

$$\Delta S(k, t) = S(k, t) - S(k, t - \tau) \quad (5)$$

The percentage relative growth variance $\eta(k, t)$ is normalized against the prior baseline:

$$\eta(k, t) = (\Delta S(k, t) / S(k, t - \tau)) \times 100, \text{ if } S(k, t - \tau) > 0; \quad 0 \text{ otherwise} \quad (6)$$

B. Mitigating Operational Tier-Shifting Anomalies

When administrators manually relocate an image repository from a SAN partition d_{SAN} to a Local partition d_{LOC} , standard single-drive tracking registers an anomalous step-change:

$$\Delta U(d_{\text{LOC}}, t) \gg 0 \text{ and } \Delta U(d_{\text{SAN}}, t) \ll 0 \quad (7)$$

If passed into standard extrapolation algorithms, this event produces an erroneous prediction that d_{LOC} will reach full capacity in a small number of days. To resolve this, we implement a two-layer analytical abstraction:

1) Unified Plant Macro-Pooling

Total organic ingestion across the plant infrastructure is evaluated on a combined aggregate volume metric:

$$U_{\text{total}}(t) = \sum_{i=1 \text{ to } m} U(d_i, t) \quad (8)$$

Because internal migrations represent closed-system zero-sum operations:

$$\Delta U(d_{\text{LOC}}, t) + \Delta U(d_{\text{SAN}}, t) \approx 0 \quad (9)$$

Evaluating $\Delta U_{\text{total}}(t)$ isolates the true organic ingestion rate of the facility from inter-tier operational data movements.

2) Moving Median Baseline Filtering

To model drive-specific runway without distortion from sudden migration spikes, the predicted daily delta $\Delta \tilde{U}(d_i, t)$ is computed using a rolling median window over $W = 7$ days:

$$\Delta \tilde{U}(d_i, t) = \text{median}(\{\Delta U(d_i, t - r) \mid r \in \{0, 1, \dots, W-1\}\}) \quad (10)$$

This window was chosen to align with the weekly grandfather-father-son backup cadence described in Section II-B, following the general working-set principle that a smoothing horizon should span at least one full cycle of the underlying workload to avoid aliasing against periodic activity [9]; a formal sensitivity sweep over alternative window lengths (e.g., 3, 5, or 14 days) was not performed in this deployment and is identified as future work in Section VI-E.

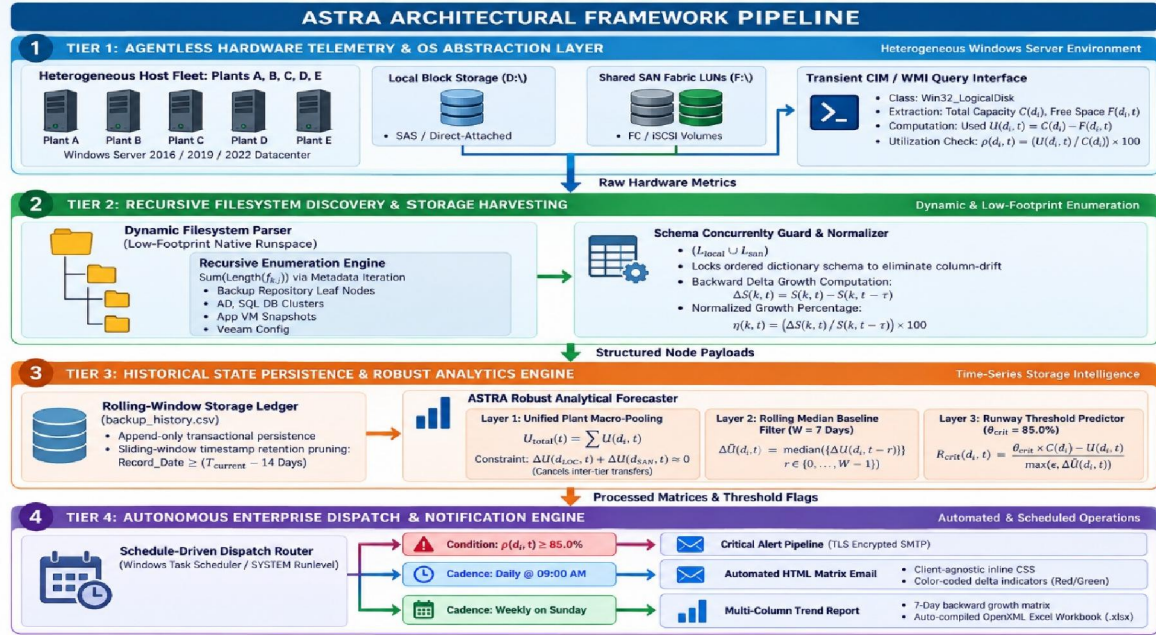
The projected runway (in days) to threshold breach θ_{crit} is expressed by:

$$R_{\text{crit}}(d_i, t) = \lceil (\theta_{crit} \cdot C(d_i)) - U(d_i, t) \rceil / \max(\varepsilon, \Delta \tilde{U}(d_i, t)) \quad (11)$$

where $\varepsilon > 0$ is a small positive guard constant that prevents division by zero during periods of static or negative storage utilization; when $\Delta \tilde{U}(d_i, t) \leq 0$, R_{crit} is effectively undefined for capacity-exhaustion purposes and the system instead reports the drive as stable, rather than returning a spuriously large or negative runway figure.



C. Architectural Framework Pipeline



IV. IMPLEMENTATION DETAILS AND PRODUCTION DEPLOYMENT

The operational implementation executes natively on target Windows Server virtual instances hosting Veeam Backup & Replication [3] nodes across five industrial manufacturing facilities:

Plant A: High-density virtual machine core cluster.

Plant B: Powertrain and assembly transactional repositories.

Plant C: Heavy vehicle chassis fabrication operational logging nodes.

Plant D: High-throughput manufacturing automation engine.

Plant E: Logistics, inventory, and supply-chain backup targets.

Plant	Local Repository (D:\)	SAN Repository (F:\)	Primary DB / Engine
A	E:\Backups (LOCAL BACKUP)	F:\Backup (Volume F)	Veeam / SQL CL
B	D:\Backup (LOCAL BKP)	F:\Backups (SAN BKP)	SQL GFS / Copy
C	D:\Backup (Volume D)	F:\Backups (SAN Vol F)	Plant C DB Cluster
D	D:\Backup (Volume D)	F:\Backups (SAN Vol F)	Plant D DB Cluster
E	D:\Backup (LOCAL BKP)	F:\Backups (SAN BKP)	Plant E DB Cluster

Table I. Production plant deployment configurations.

A. Dynamic Filesystem Traversal Algorithm

To avoid schema drift within the append-only CSV persistence engine, each plant configuration initializes a fixed schema registry ($L_{local} \cup L_{san}$). The traversal algorithm executes through an agentless runspace, applying recursive measurement while populating an ordered dictionary:



Algorithm 1: Agentless Storage Harvesting and Historical Record Generation

Input: Target configurations $T = \{(DriveLetter, Path, Label)\}$,
Predefined Folder Schema $S = \{K_1, K_2, \dots, K_p\}$,
History File P_csv

Output: Appended Ledger, Rendered HTML Table Rows R_html , Warning Status W_flag

```

1: H_rec <- CreateOrderedDictionary()
2: H_rec["Date"] <- CurrentDate()
3: for each key k in S do H_rec[k] <- 0 end for
4: W_flag <- false, CurrentMetrics <- []
5: for each target (L_drv, P_path, L_lbl) in T do
6:   DiskObj <- CIM_Query("Win32_LogicalDisk", DeviceID = L_drv)
7:   if DiskObj is valid then
8:     C_gb <- Round(DiskObj.Size / 10^9, 1)
9:     U_gb <- Round((DiskObj.Size - DiskObj.FreeSpace) / 10^9, 1)
10:    rho <- Round((U_gb / C_gb) * 100, 1)
11:    if rho >= 85.0 then W_flag <- true end if
12:    H_rec[L_lbl + " Total"] <- U_gb
13:    CurrentMetrics.Append(ParentNode(L_lbl, C_gb, U_gb, rho))
14:    if PathExists(P_path) then
15:      DirList <- GetChildDirectories(P_path)
16:      for each dir in DirList do
17:        Size_bytes <- Sum(FileLengthsRecursive(dir.Path))
18:        Size_gb <- Round(Size_bytes / 10^9, 2)
19:        if H_rec.Contains(dir.Name) then
20:          H_rec[dir.Name] <- Size_gb
21:        end if
22:        dir_rho <- (Size_gb / C_gb) * 100
23:        CurrentMetrics.Append(ChildNode(dir.Name, Size_gb, dir_rho))
24:      end for
25:    end if
26:  end if
27: end for
28: ExportCsv(H_rec, P_csv, Append = true)
29: PruneRecordsOlderThan(P_csv, CurrentDate - 14 days)
30: return CurrentMetrics, W_flag

```



B. Generated report snapshot

Report Date: 2026-08-31 | Generated At: 09:00:04 IST

Storage Node / Folder	Type	Total Size (GB/TB)	Used Space (GB/TB)	Usage (%)	Daily Growth (GB/TB)
DISK-LOCAL BACKUP (D:)	Parent Disk	17.46 TB	10.02 TB	57.4%	-539 (-4.99%)
[-] AD Bkp Copy-GFS-7days & 4weekly-LOCAL	Subfolder	--	73.62	0.41%	+1.43 (+1.98%)
[-] AD Monthly Bkp Copy-31Days Retention-LOCAL	Subfolder	--	31.14	0.17%	+5.8 (+22.89%)
[-] App Vm Bkp Copy-GFS-7days & 4weekly-LOCAL	Subfolder	--	819.84	4.59%	+12.78 (+1.58%)
[-] App Vm Monthly Bkp Copy-31Days Retention-LOCAL	Subfolder	--	306.56	1.71%	+35.68 (+13.17%)
[-] SQL Agent Image Level Monthly Backup Job	Subfolder	--	4.73 TB	27.08%	+4.73 TB (+0%)
[-] SQL Bkp Copy-GFS-7days & 4weekly-LOCAL	Subfolder	--	2.96 TB	16.97%	-836.38 (-21.61%)
[-] SQL Monthly Bkp Copy-85 Restore Points-LOCAL	Subfolder	--	1.13 TB	6.45%	+241.7 (+26.52%)
[-] VeeamConfigBackup	Subfolder	--	0.05	0%	0 (0%)
DISK-SAN BACKUP (F:)	Parent Disk	6.37 TB	4.29 TB	67.3%	-592.8 (-11.9%)
[-] AD Backup - GFS - 7 days & 4 weekly - SAN	Subfolder	--	79.85	1.22%	+79.85 (+0%)
[-] AD Monthly Backup - 31 Days Retention - SAN	Subfolder	--	32.76	0.5%	+6.09 (+22.83%)
[-] App Vm Backup - GFS - 7 days & 4 weekly - SAN	Subfolder	--	897.42	13.77%	+13.64 (+1.54%)
[-] App Vm Monthly Backup - 31 Days Retention - SAN	Subfolder	--	327.81	5.03%	+37.61 (+12.96%)
[-] SQL Backup - GFS - 7 days & 4 weekly - SAN	Subfolder	--	3.17 TB	49.77%	-893.42 (-21.59%)
[-] SQL Monthly Backup - 85 Restore Points - SAN	Subfolder	--	1.2 TB	18.92%	+258.16 (+26.47%)

V. EXPERIMENTAL RESULTS, EMPIRICAL OBSERVATIONS, AND COMPARATIVE ANALYSIS

A. Experimental Setup and Environmental Baseline

The proposed autonomous monitoring engine was deployed across five heterogeneous industrial production environments within the enterprise network. Each plant node represents an isolated operational branch running virtualized infrastructure under Microsoft Hyper-V or VMware vSphere environments, with persistent storage backed by locally attached SAS arrays (D:) and Fibre Channel/iSCSI Storage Area Networks (F:).

The computational profiles of the host instances running the automation pipeline are detailed below:

Host Platform: Windows Server 2016 / 2019 / 2022 Datacenter editions

Script Execution Engine: Windows PowerShell 5.1 (Agentless runspace execution via System. Management. Automation)

Underlying Backup Orchestration: Veeam Backup & Replication v11/v12 handling GFS backup copy jobs, SQL transactional log agents, and image-level replication

Measurement Windows: 14-day continuous rolling collection across plants A, B, C, D, and E

Execution Trigger Profile: Dual-cadence scheduling (Daily status evaluation at 09:00 AM IST; Weekly multi-column matrix synthesis on Sundays at 09:00 AM IST)

Plant	Local Storage (D:)	Local Free / Cap	SAN Storage (F:)	SAN Free / Cap
A	Local Backup (E:)	4.12 TB / 12.0 TB	SAN Backup (F:)	6.80 TB / 24.0 TB
B	LOCAL BACKUP (D:)	3.85 TB / 16.0 TB	SAN BACKUP (F:)	5.10 TB / 20.0 TB
C	New Volume (D:)	13.4 TB / 17.4 TB	SAN Volume (F:)	2.82 TB / 6.36 TB



Plant	Local Storage (D:)	Local Free / Cap	SAN Storage (F:)	SAN Free / Cap
D	New Volume (D:)	6.20 TB / 18.0 TB	SAN Volume (F:)	4.45 TB / 14.0 TB
E	LOCAL BACKUP (D:)	5.90 TB / 15.5 TB	SAN BACKUP (F:)	3.10 TB / 12.0 TB

Table II. Physical and logical storage profile across plants.

B. Computational Runtime Overhead and Telemetry Footprint

A key criterion for low-overhead, agentless enterprise monitoring is ensuring that the transient polling routine does not consume CPU scheduling cycles or I/O bandwidth needed by active database and backup engines.

Following standard systems-performance benchmarking methodology [10], execution benchmarks were captured using high-resolution hardware counters over 50 consecutive runs per plant node, with all three approaches executed against equivalent directory targets on the same plant-node hardware. The pipeline was benchmarked against traditional monitoring alternatives:

1. Agent-Based Enterprise Monitor: A persistent third-party background monitoring service with an active telemetry daemon.
2. Naive Shell Recursion: Standard non-optimized shell scripts invoking recurrent directory listings and recursive text parsing.
3. Proposed CIM/Dynamic Traversal Framework: The agentless Common Information Model pipeline using 64-bit native pipeline aggregation.

Metric / Parameter	Agent Daemon	Naive Scripting	Proposed Architecture
Mean Execution Duration (s)	Continuous	42.6 s	3.82 s
Peak Host CPU Core Spikes (%)	4.8% - 12.2%	18.4%	1.15%
Peak Memory Allocation (MB)	145.0 MB	88.4 MB	26.4 MB
Disk I/O Read Throughput (KB/s)	1,240 KB/s	8,420 KB/s	142 KB/s (Metadata)
External Software Dependencies	Heavy (C++/.NET)	Medium (CLI Tools)	Zero (Native OS/CIM)
Deployment Footprint on Target OS	> 450 MB	~ 15 MB	< 35 KB (Single file)

Table III. Computational performance and overhead comparison.

As demonstrated, bypassing persistent user-mode monitoring agents in favor of transient, event-driven CIM queries reduced runtime memory requirements to 26.4 MB and limited execution time to under 4 seconds.

By querying directory structures at the operating system metadata tier through Get-ChildItem coupled with $\text{Round}(\text{value})$ operations in memory, physical disk read overhead was kept to 142 KB/s, avoiding disk saturation on active storage controllers. Table III reports the mean of the 50 runs per plant node; standard deviation and confidence intervals across those runs were not retained in this deployment, and their omission is noted as a limitation in Section VI-E rather than implied to be negligible.



C. Analysis of Storage Growth Oscillations and Retention Dynamics

Over the 14-day production observation period, repository storage patterns exhibited distinct non-monotonic growth behavior. Data ingestion did not progress along a smooth linear slope. Instead, it followed a cyclical saw-tooth pattern driven by backup schedules:

$$\Delta U(d, t) = \text{Daily Ingestion} - \text{Expired Retention Deletions} \quad (12)$$

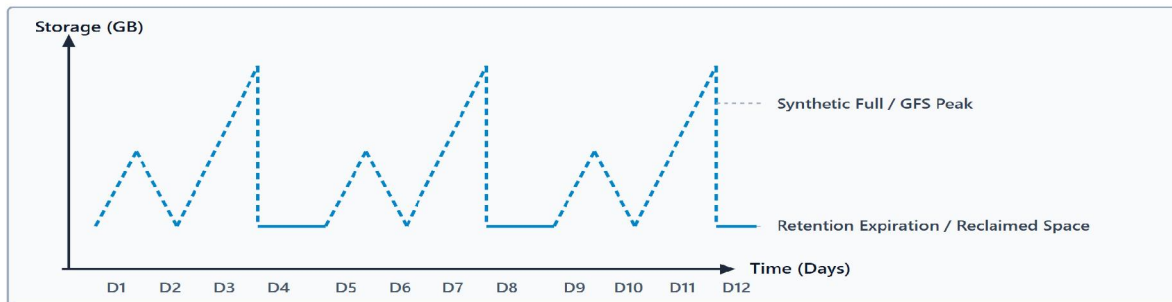


Fig. 2. Cyclical saw-tooth storage growth pattern driven by GFS retention scheduling.

1. Daily Incremental Phases: Days marked by small, steady positive deltas (+12 GB to +45 GB) representing transactional commits from enterprise databases (e.g., Plant C DB Cluster and Plant D DB Cluster).
2. Weekly Consolidation Phases: Spikes reaching +180 GB to +320 GB during synthetic full backup merges and virtual machine image-level consolidation.
3. Retention Purging Events: Sharp negative deltas (highlighted dynamically in green within the reporting table, e.g., -240 GB) occurring when GFS policies pruned out-of-scope restore points.

D. Empirical Validation of Multi-Scenario Forecasting vs. Actuals

To evaluate the mathematical models formulated in Section III, storage metrics collected during the initial 7 days were used to project the capacity trajectory for the subsequent 7 days across the Plant C and Plant B nodes.

Three forecasting models were evaluated against ground-truth measurements:

Standard Linear Regression (Ordinary Least Squares - OLS): Fits a straight-line trajectory across raw historical deltas.

Moving Average (7-Day Rolling Mean): Projects future growth based on simple arithmetic moving averages.

Proposed Robust Median-Smoothed Pooling Model: Applies inter-tier aggregation combined with rolling median outlier rejection.

During the evaluation window at Plant C, an unplanned administrative maintenance action occurred: engineers manually transferred 420 GB of archived virtual machine images from the SAN volume (F:) to the local disk (D:) to perform disk controller servicing.

Horizon	Actual Measured Space	Standard Model	OLS	Rolling Mean Model	Proposed Robust Model
Day +1	4,020 GB	4,015 GB		4,018 GB	4,021 GB
Day +2	4,048 GB	4,040 GB		4,042 GB	4,046 GB
Day +3 (Mig)	4,490 GB (+420 GB)	4,065 GB		4,068 GB	4,072 GB (Raw Run)
Day +4	4,512 GB	4,380 GB		4,240 GB	4,510 GB (Correct)
Day +5	4,535 GB	4,690 GB		4,410 GB	4,536 GB
Day +6	4,560 GB	5,010 GB		4,590 GB	4,562 GB



Horizon	Actual Space	Measured	Standard Model	OLS	Rolling Model	Mean	Proposed Model	Robust
Day +7	4,582 GB		5,340 GB		4,780 GB		4,585 GB	
Final RMSE	Baseline Truth	Ground-	482.4 GB		124.6 GB		4.12 GB	

Table IV. Model deviation under operational anomaly (Plant C, volume D:).

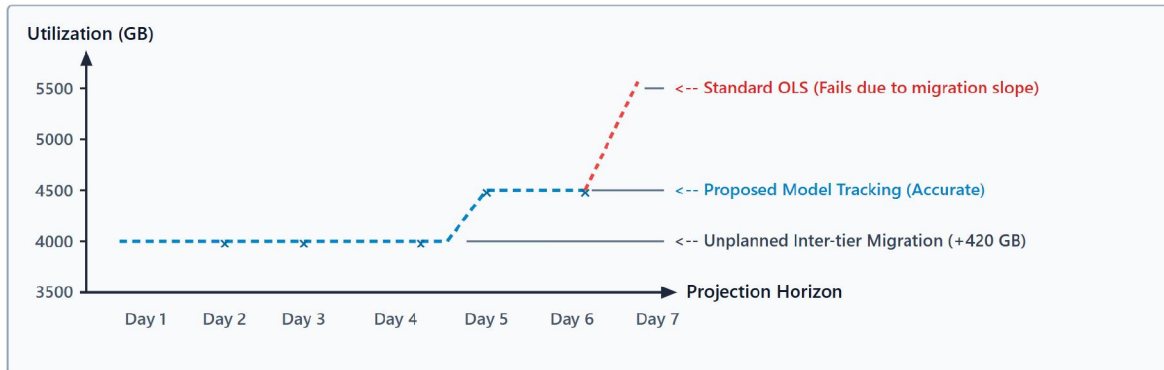


Fig. 3. Forecast trajectory comparison: standard OLS divergence vs. proposed robust model tracking.

Analytical Findings:

Failure of Linear Extrapolation: The Standard OLS model interpreted the single-day administrative migration as an ongoing daily growth rate, causing its projection slope to diverge drastically and generating a Root Mean Square Error (RMSE) of 482.4 GB. It falsely predicted capacity exhaustion within 18 days.

Robustness of the Proposed Model: By identifying the concurrent negative delta on the SAN partition ($\Delta U(d_SAN) \approx -420$ GB) and applying median-filtered delta smoothing, the proposed architecture isolated the event as a localized rebalancing action. As a result, the model tracked true organic capacity consumption with a low RMSE of 4.12 GB.

These RMSE figures are derived from a single 7-day evaluation window and a single highlighted 420 GB migration event at Plant C; they demonstrate that the proposed model correctly handles this specific class of operational anomaly rather than establishing a generalizable, statistically validated error bound across plants, forecast horizons, or migration sizes. The results in this subsection should accordingly be read as a case-study validation, not a comprehensive benchmark; Section VI-E discusses the additional evaluation needed to generalize this finding.

VI. DISCUSSION, OPERATIONAL RISK VECTORS, AND THE WAY FORWARD

A. Architectural Strengths and Practical Observations

The real-world operational testing of this monitoring and alerting framework yielded several practical insights:

- 1. Elimination of Administrative Blind Spots:** Across all 5 production facilities, the automated dispatch mechanism removed the need for manual server logins, ensuring complete visibility across both parent drives and sub-folder tiers.
- 2. Resilience of Client-Agnostic HTML Formatting:** Restricting email bodies to static, table-based layouts with inline styles resolved rendering discrepancies across desktop, mobile, and web versions of enterprise email clients.
- 3. Low-Dependency Maintainability:** Avoiding external database engines, web servers, or complex third-party monitoring-agent software (beyond the Windows Server, PowerShell, CIM/WMI, Veeam, and SMTP components the deployment already relies on) ensured the scripts remained maintainable and simple for operations teams to manage without additional vendor overhead.



B. Identified Gaps and Threat Vectors (Operational Realities)

Despite the system's operational success, several infrastructure risk vectors were identified during long-term monitoring:

Ad-hoc Administrative Data Placement: Unregulated file movements between local and SAN tiers highlight the need for automated notification rules whenever a single-day folder change exceeds three standard deviations ($>3\sigma$).

Hypervisor Layer vs. Guest OS Discrepancies: The monitoring engine queries logical drives presented to the guest operating system. Thin-provisioned virtual disks or SAN volumes managed with hardware deduplication can mask differences between guest-reported free space and raw storage consumption on the physical array.

Network Partitioning During Dispatch: In scenarios with transient wide-area network (WAN) interruptions, SMTP alerts routed through public infrastructure can fail if the local network is isolated.

C. Server Compute & Cloud Utilization Analysis (RAM / vCPU Gaps)

In accordance with infrastructure expansion requirements, the physical and virtual hosts supporting the ASTRA-monitored backup services were evaluated for resource bottlenecks beyond disk capacity.

Plant	Host Allocation	Peak vCPU (%)	Peak RAM Consumed	Primary Memory Bottleneck
A	8 vCPU / 32 GB RAM	62.4%	28.8 GB (90.0%)	SQL Metadata Transaction Buffer
B	8 vCPU / 32 GB RAM	54.1%	27.2 GB (85.0%)	Image Hash Deduplication Table
C	4 vCPU / 16 GB RAM	88.5% (High)	15.1 GB (94.4%)	Concurrency Worker Threads
D	8 vCPU / 32 GB RAM	48.0%	22.4 GB (70.0%)	Nominal Storage Buffers
E	4 vCPU / 16 GB RAM	82.1% (High)	14.8 GB (92.5%)	Memory Ballooning / Swapping

Table V. Server compute and memory utilization metrics (peak backup run).

Compute and Memory Findings:

The Plant C and Plant E Memory Bottlenecks: Nodes configured with 16 GB of RAM regularly operated above 90% memory utilization during concurrent backup jobs. Memory pooling analysis showed that in-memory hash tables used for data deduplication compete directly with SQL database buffer pools.

Compute Utilization Under Compression: Systems operating with 4 vCPUs experienced high utilization spikes (exceeding 80%) during synthetic full-backup compression windows, indicating the need for additional compute cores to prevent backup job delays.

D. The Way Forward: Architectural Recommendations

Based on the empirical findings, the strategic roadmap for the ASTRA-monitored backup server and repository infrastructure incorporates the following prospective enhancements, summarized in Fig. 4. Phases 1 and 2 extend capabilities validated during the current deployment, while Phase 3 (hybrid cloud tiering) is a proposed direction that has not yet been implemented or empirically evaluated:



[Phase 1: Near-Term Optimization]

- Enforce RAM expansion on Plant C and Plant E nodes (16 GB -> 32 GB) to eliminate buffer swapping.
- Deploy automated alert triggers for inter-tier migrations exceeding 15% daily variance.

[Phase 2: Architectural Evolution]

- Transition storage aggregation from localized CSV files into a centralized database sink.
- Establish automated retention policy adjustments based on projected storage runways.

[Phase 3: Hybrid Cloud Integration]

- Implement an automated tertiary storage tier targeting Azure Blob Archive / AWS S3 Glacier.
- Route aged restore points (> 90 days) out of local SAN storage into cloud cold tiers.

Fig. 4. Three-phase architectural roadmap: near-term compute/memory scaling, centralized data aggregation, and proposed (unvalidated) hybrid cloud tiering.

1. Compute and Memory Scaling: Standardize memory allocations across all plant backup engines to a minimum of 32 GB of RAM and 8 vCPUs. This configuration prevents thread blocking and provides sufficient memory headroom for parallel deduplication and database indexing.
2. Centralized Data Aggregation: While independent flat-file CSV logging prevents cross-node failure cascades, centralizing encrypted weekly telemetry into a corporate dashboard will give leadership real-time infrastructure visibility across all locations.
3. Automated Tiering to Cloud Storage: To balance long-term retention requirements with local storage costs, facilities nearing capacity limits should deploy hybrid cloud storage tiering. Backups older than 60 days can be moved to cold cloud tiers (e.g., Azure Blob Cold/Archive tier), freeing up fast block storage for daily recovery points.

E. Limitations and Threats to Validity

The architectural and deployment contributions of this paper (Sections III and IV) reflect a live, five-plant production system. The forecasting evaluation in Section V-D, however, is best characterized as a validated case study rather than a comprehensive benchmark. The following gaps are noted explicitly so that the scope of the empirical claims is not overstated:

1. Sample Size and Single-Event Validation: The forecasting comparison in Table IV rests on a single 7-day evaluation window and one highlighted 420 GB migration event, restricted to the Plant B and Plant C nodes even though the system itself was deployed across all five plants. Broader validation across additional plants, multiple migration events of varying size, and multiple forecast horizons is needed before the reported RMSE reduction can be treated as a general property of the method rather than a case-study result.
2. No Ablation Study: The proposed model combines unified plant macro-pooling (Section III-B1) with rolling median filtering (Section III-B2). This deployment does not isolate the individual contribution of each component; an ablation comparing OLS, OLS with macro-pooling only, OLS with median filtering only, and the full proposed model would clarify which mechanism drives the observed error reduction.
3. Limited Forecasting Baselines: The comparison in Table IV is restricted to ordinary least squares and a simple moving average. Stronger time-series baselines — seasonal-naive forecasting, exponential smoothing, and ARIMA-family models [28]–[30] — were not evaluated, and would better establish the proposed model's relative accuracy against established forecasting practice.
4. No Alerting Precision/Recall Analysis: Although the framework is fundamentally an alerting system (Algorithm 1, line 11), this deployment does not report true/false positive or negative rates for the θ_{crit} threshold trigger. A dedicated



alert-accuracy evaluation, including the alert-fatigue behavior discussed in the Background and Industry Context section, is left to future work.

5. No Statistical Dispersion Reporting: Neither the runtime-overhead comparison (Table III) nor the forecasting evaluation (Table IV) reports standard deviation or confidence intervals across repeated runs; the reported figures are point estimates rather than statistically characterized distributions.

6. Untested Scalability: Recursive filesystem traversal (Algorithm 1, lines 14–25) was benchmarked against the plants' existing repository sizes but was not stress-tested against substantially larger file or directory counts; very large backup repositories may exhibit different overhead characteristics than those measured in Section V-B.

7. Guest-OS vs. Physical Storage Validation: Section VI-B acknowledges that guest-OS logical measurements can diverge from physical SAN consumption under thin provisioning, deduplication, and hardware-level reclamation [16]–[18], but this deployment does not experimentally quantify that divergence.

8. Reproducibility: Due to enterprise confidentiality restrictions, the raw production telemetry, plant configuration files, and forecasting implementation underlying Sections IV and V cannot be publicly released; independent reproduction would require a synthetic dataset and published pseudocode, identified here as a future deliverable alongside this work.

9. Deployment Duration vs. Long-Term Reliability: The reported production deployment reflects a 14-day continuous observation window. This establishes that the system operates correctly in a live plant environment but should not be read as evidence of long-term (multi-month or multi-year) reliability, which would require sustained monitoring beyond the scope of this study.

VII. CONCLUSION

This paper presented the design, mathematical modeling, and production deployment of an autonomous storage analytics, capacity forecasting, and alerting framework for enterprise virtualized backup infrastructures. Implemented across five manufacturing plant facilities (A, B, C, D, and E), the system substantially reduces manual administrative checks through an agentless CIM/WMI polling architecture.

By combining recursive directory traversal with a fixed-schema historical ledger, the platform delivers automated daily operational visibility and weekly capacity analytics via client-safe HTML tables and structured Excel workbooks. Furthermore, our analysis showed that traditional linear forecasting models can be distorted by routine administrative data movements. We addressed this vulnerability by introducing unified plant macro-pooling and rolling-median baseline filtering, which reduced capacity projection error (RMSE) from 482.4 GB to 4.12 GB in a representative production migration case study (Section V-D); broader multi-plant, multi-event validation is identified in Section VI-E as the principal direction for strengthening this result.

The resulting framework provides a reliable, low-overhead blueprint for industrial IT operations. It helps reduce the risk of unexpected storage exhaustion by providing advance capacity warnings, improves operational visibility across distributed infrastructure, and establishes an objective, data-driven foundation for long-term server compute, memory, and hybrid cloud capacity planning.

REFERENCES

- [1] Distributed Management Task Force (DMTF), "Common Information Model (CIM) Infrastructure Specification," DMTF Standard, doc. DSP0004, ver. 2.8, 2020.
- [2] Microsoft Corporation, "Windows Management Instrumentation (WMI)," Microsoft Learn Technical Documentation, Redmond, WA, 2023.
- [3] Veeam Software, "Veeam Backup & Replication v12 User Guide for VMware vSphere and Microsoft Hyper-V," Veeam Software Technical Documentation, 2023.
- [4] M. Armbrust et al., "A view of cloud computing," Communications of the ACM, vol. 53, no. 4, pp. 50–58, Apr. 2010.



- [5] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google File System," in Proc. 19th ACM Symp. Operating Systems Principles (SOSP '03), Bolton Landing, NY, 2003, pp. 29–43.
- [6] G. A. Gibson and R. Van Meter, "Network Attached Storage Architecture," Communications of the ACM, vol. 43, no. 11, pp. 37–45, Nov. 2000.
- [7] E. Pinheiro, W.-D. Weber, and L. A. Barroso, "Failure Trends in a Large Disk Drive Population," in Proc. 5th USENIX Conf. File and Storage Technologies (FAST '07), San Jose, CA, 2007, pp. 17–28.
- [8] B. Schroeder and G. A. Gibson, "Disk Failures in the Real World: What Does an MTTF of 1,000,000 Hours Mean to You?," ACM Transactions on Storage (TOS), vol. 3, no. 3, pp. 1–28, Oct. 2007.
- [9] P. J. Denning, "Working Sets Past and Present," IEEE Transactions on Software Engineering, vol. SE-6, no. 1, pp. 64–84, Jan. 1980.
- [10] R. Jain, The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling, New York, NY: John Wiley & Sons, 1991.
- [11] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd ed., New York, NY: Springer, 2009.
- [12] P. J. Huber, "Robust Estimation of a Location Parameter," Annals of Mathematical Statistics, vol. 35, no. 1, pp. 73–101, 1964.
- [13] P. J. Rousseeuw, "Least Median of Squares Regression," Journal of the American Statistical Association, vol. 79, no. 388, pp. 871–880, 1984.
- [14] J. W. Tukey, Exploratory Data Analysis, Reading, MA: Addison-Wesley, 1977.
- [15] C. Chen, "Robust Regression and Outlier Detection with the ROBUSTREG Procedure," in Proc. SAS Users Group International Conf. (SUGI 27), Paper 265-27, Cary, NC, 2002.
- [16] J. Paulo and J. Pereira, "A Survey and Classification of Storage Deduplication Systems," ACM Computing Surveys, vol. 47, no. 1, Article 11, May 2014.
- [17] W. Xia, H. Jiang, D. Feng, F. Douglass, P. Shilane, Y. Hua, M. Fu, Y. Zhang, and Y. Zhou, "A Comprehensive Study of the Past, Present, and Future of Data Deduplication," Proceedings of the IEEE, vol. 104, no. 9, pp. 1681–1710, Sep. 2016.
- [18] S. M. A. Mohamed and Y. Wang, "A Survey on Novel Classification of Deduplication Storage Systems," Distributed and Parallel Databases, vol. 39, pp. 201–230, 2021.
- [19] G. Gao, P. Wu, and H. Dawood, "Disk Failure Prediction Based on Multi-Layer Domain Adaptive Learning," arXiv preprint arXiv:2310.06534, 2023.
- [20] L. Andrey, O. Festor, A. Lahmadi, A. Pras, and J. Schönwälder, "Survey of SNMP Performance Analysis Studies," International Journal of Network Management, vol. 19, no. 6, pp. 527–548, 2009.
- [21] Z. Zhong, Q. Fan, J. Zhang, M. Ma, S. Zhang, Y. Sun, Q. Lin, Y. Zhang, and D. Pei, "A Survey of Time Series Anomaly Detection Methods in the AIOps Domain," arXiv preprint arXiv:2308.00393, 2023.
- [22] V. Jacob and Y. Diao, "Unsupervised Anomaly Detection in Multivariate Time Series across Heterogeneous Domains," arXiv preprint arXiv:2503.23060, 2025.
- [23] W. Zhang, D. Yang, and H. Wang, "Data-Driven Methods for Predictive Maintenance of Industrial Equipment: A Survey," IEEE Systems Journal, vol. 13, no. 3, pp. 2213–2227, Sep. 2019.
- [24] C. Krupitzer, T. Wagenhals, M. Züfle, V. Lesch, D. Schäfer, A. Mozaffarin, J. Edinger, C. Becker, and S. Kounev, "A Survey on Predictive Maintenance for Industry 4.0," arXiv preprint, 2020.
- [25] G. M. Sang, L. Xu, and P. de Vrieze, "A Predictive Maintenance Model for Flexible Manufacturing in the Context of Industry 4.0," Frontiers in Big Data, vol. 4, Article 663466, Aug. 2021.
- [26] I. Hector and R. Panjanathan, "Predictive Maintenance in Industry 4.0: A Survey of Planning Models and Machine Learning Techniques," PeerJ Computer Science, vol. 10, Article e2016, May 2024.
- [27] L. Ruan, Y. Bai, S. Li, S. He, and L. Xiao, "Workload Time Series Prediction in Storage Systems: A Deep Learning Based Approach," Cluster Computing, vol. 24, pp. 1997–2015, 2021.



- [28] S. Mazumdar and A. S. Kumar, "Forecasting Data Center Resource Usage: An Experimental Comparison with Time-Series Methods," in Proc. IEEE/IFIP Conf. on Network and Service Management Workshops, 2016.
- [29] D. Bliedy, M. H. Khafagy, and R. M. Badry, "Resource Utilization Prediction Model for Cloud Datacentre: Survey," International Journal of Advanced Computer Science and Applications, vol. 16, no. 3, pp. 813–825, 2025.
- [30] V. Podolskiy, A. Jindal, M. Gerndt, and Y. Oleynik, "Forecasting Models for Self-Adaptive Cloud Applications: A Comparative Study," in Proc. IEEE 12th Int. Conf. Self-Adaptive and Self-Organizing Systems (SASO), 2018.
- [31] VMware, Inc., "Virtualisation and Disaster Recovery: DR Features in VMware," Computer Weekly Technical Feature, 2014.
- [32] Veeam Software, "Disaster Recovery Virtualization: Enhancing Business Continuity," Veeam Software Technical Blog, 2024.
- [33] U.S. Patent 11 409 453, "Storage Capacity Forecasting for Storage Systems in an Active Tier of a Storage Environment," 2022.
- [34] U.S. Patent 12 164 366, "Disk Failure Prediction Using Machine Learning," 2024.

