TECHNICAL RESEARCH PAPER

Engineering Proactive, Resilient and Intelligent ICT Managed Services

Service management · observability · automation · incident response · security · continual improvement

Issuing organisation	Al Moammar Information Systems Co. (MIS), Kingdom of Saudi Arabia
Research domain	ICT Managed Services
Document type	Technical research paper / white paper
Author of record	Mohammed Afif – Operation and Maintenance manager
Version / status	1.0
Date of publication	[to be stamped on the day of publication]
Permanent link	[to be assigned on the official MIS domain]
Suggested citation	Al Moammar Information Systems (MIS), Engineering Proactive, Resilient and Intelligent ICT Managed Services, technical research paper v1.1. Riyadh: MIS.
Distribution	Public — may be shared and quoted with attribution to MIS.

Abstract

ICT managed services are moving from traditional reactive support toward continuously monitored, automated, resilient and increasingly predictive service delivery.

The change is driven by the complexity of the environments being managed. A single organisation may operate applications, networks, cloud services, data centres, security platforms, storage, digital services, endpoints and third-party technologies at once, and managing that estate requires more than responding to incidents after service has already degraded.

This paper examines the engineering and service-management capabilities required to

make that transition. It applies a structured qualitative methodology based on recognised service-management standards and guidance — principally ISO/IEC 20000-1:2018 and ITIL 4 — together with cybersecurity, resilience and operational-management guidance and the national cybersecurity control environment of the Kingdom.

Eight dimensions are examined — service governance, service-level management, observability, event and incident management, automation, proactive problem management, cybersecurity integration and continual improvement — expressed as sixteen operating principles.

The analysis indicates that a mature managed service cannot be judged by incidents closed or engineers assigned. It should demonstrate that service performance is measurable, that operational dependencies are visible, that abnormal behaviour is detected early, that incidents are contained efficiently, that repetitive work is automated under control, that recurring problems are analysed, and that operational experience drives improvement.

The paper proposes an Intelligent Managed Services Engineering Framework following the cycle define, observe, detect, respond, resolve, analyse, automate, improve — a vendor-neutral model for assessing managed-service maturity.

KEYWORDS *ICT managed services; IT service management; ITIL 4; ISO/IEC 20000-1; service-level management; service-level agreements; observability; automation; incident management; problem management; service resilience; proactive operations; predictive operations.*

1 Introduction

Enterprise technology environments are becoming more distributed and more interconnected. An organisation may depend simultaneously on data-centre infrastructure, cloud services, networks, applications, databases, storage, cybersecurity platforms, collaboration services, digital platforms, third-party systems and end-user technology.

A failure in one technical layer can therefore affect several business services at once. An application-performance complaint may originate in the application, in the database beneath it, in storage, in the network, in the identity service, or in a cloud dependency outside the organisation altogether — and a conventional service desk typically receives the report only after users have felt the disruption.

A mature managed-service environment sets out to identify abnormal behaviour earlier than that. The governing question is consequently not how to resolve incidents faster.

How can managed ICT services continuously identify, control and reduce operational risk before technology failures materially affect business services?

2 Research problem

Traditional support models depend heavily on user-reported incidents, following a sequence of failure, user impact, ticket, investigation and resolution. That model remains necessary, but on its own it produces recognisable limitations:

- Delayed incident detection;
- Repeated incidents with identical symptoms;
- Fragmented technical monitoring;
- Siloed operational teams;
- Dependence on individual expertise;
- Manual repetitive work;
- Limited correlation between technical events;
- Reactive capacity planning;
- Weak root-cause analysis;
- Limited learning from operational history.

As environments grow more complex, reactive support alone becomes progressively less sufficient. The research problem is therefore how ICT managed services can evolve from reactive incident handling toward measurable, proactive, automated and resilient service management.

3 Research question

What engineering, operational and governance capabilities enable ICT managed services to transition from reactive support toward proactive and intelligent service operations?

The supporting questions are:

- How should managed services be governed across the service lifecycle?
- Which operational indicators should be monitored?
- How does observability improve incident detection?
- How should incidents be integrated with cybersecurity risk management?

- Which activities are suitable for automation, and which are not?
- How can recurring incidents be converted into preventive action?
- How can service performance be improved continually rather than periodically?
- What distinguishes reactive, proactive, preventive and predictive operating models?

4 Research objectives

The research pursues eight objectives.

1. Define managed services as a continuous service-management lifecycle.
2. Examine service governance and measurable service outcomes.
3. Analyse monitoring and observability as mechanisms for early detection.
4. Examine incident and problem management as distinct disciplines.
5. Evaluate automation as an operational control rather than a convenience.
6. Integrate cybersecurity incident response into managed operations.
7. Examine continual improvement as a service characteristic.
8. Consolidate the results into an Intelligent Managed Services Engineering Framework.

5 Background

5.1 Managed services as a service lifecycle

Managed services differ from isolated technical support. Support addresses an individual issue; managed services carry ongoing responsibility for the performance and quality of defined services.

ISO/IEC 20000-1:2018 specifies requirements for establishing, implementing, maintaining and continually improving a service management system, and its scope covers the planning, design, transition, delivery and improvement of services in order to meet requirements and deliver value[1]. The lifecycle perspective matters: a managed-service provider should work through plan, design, transition, deliver, measure and improve, rather than treat managed services as operational staffing with a ticket queue attached.

[1] International Organization for Standardization, ISO/IEC 20000-1:2018, Information technology — Service management — Part 1: Service management system requirements. Geneva: ISO/IEC, 2018. Reviewed and confirmed as current in 2023.

In practical terms a managed service is defined by three commitments rather than by a headcount: an agreed scope of services and supported assets, an agreed standard of performance expressed as measurable targets, and an agreed method of governance through which performance is reported, reviewed and improved. Time-and-materials support, break/fix maintenance and project delivery may all be present inside a managed-service contract, but none of them is a managed service on its own, because none of them carries continuing accountability for the condition of the service between requests.

The service-management literature frames this as a relationship rather than a transaction. ITIL 4 describes service management as a set of organisational capabilities for enabling value through services, and treats value as co-created by the provider and the consumer through a service relationship rather than delivered unilaterally by the provider. It organises those capabilities around four dimensions — organisations and people, information and technology, partners and suppliers, and value streams and processes — and around a service value chain whose activities are plan, improve, engage, design and transition, obtain and build, and deliver and support. The practical consequence for a managed-service provider is that operational tooling is one dimension of four; a provider with excellent monitoring, undefined responsibilities and no improvement mechanism is not a mature service.[2]

The managed-service commitment therefore begins before operations do. Scope definition, service modelling, dependency discovery, baselining, runbook capture and service-level negotiation belong to the design and transition stages, and shortcomings there are inherited permanently by the operate stage. A service that enters operation without a configuration baseline, a dependency model and agreed targets cannot be measured, and what cannot be measured cannot be improved.

5.2 Managed-service delivery models and responsibility boundaries

Managed services are delivered through several models, and the model chosen determines where operational authority sits. The distinction matters because most disputes in managed-service delivery are not disputes about technical competence but about the boundary of responsibility.

- Fully managed — the provider holds end-to-end responsibility for defined services, including monitoring, incident and problem management, change execution, capacity planning and reporting.

[2] AXELOS, ITIL Foundation: ITIL 4 Edition. London: TSO, 2019.

- Co-managed — the provider and the client operate a shared model in which each party owns defined layers, practices or hours of coverage, governed by an agreed responsibility matrix.
- Monitoring and reporting only — the provider observes and reports but does not act, which limits accountability to detection and makes response times a client-side dependency.
- Resident or embedded engineering — provider personnel operate inside the client's processes, which delivers proximity but weakens the service-level construct unless outcomes remain contractually defined.
- Follow-the-sun operations — tiered network and security operations centres provide continuous coverage, with handover quality becoming a first-order service risk.

Whichever model applies, three boundaries should be documented explicitly. First, a responsibility matrix that states, for each service and each activity, who is responsible, accountable, consulted and informed. Second, the internal dependencies of the provider's own commitments — operational-level agreements with internal teams and underpinning contracts with third-party suppliers, hardware vendors, carriers and cloud providers, because an external target cannot be honoured if the internal chain supporting it is unagreed. Third, the shared-responsibility boundary in cloud and platform services, where the provider's obligations end at the interfaces the cloud provider exposes and this limit should be disclosed rather than discovered during an outage.

Two lifecycle phases deserve the same explicit treatment. Transition-in — discovery, asset and configuration baselining, dependency mapping, monitoring instrumentation, runbook capture, knowledge transfer and a bounded hyper-care period — determines whether the service can be operated to target from day one. Transition-out — documentation handover, configuration export, knowledge transfer and monitoring de-instrumentation — determines whether the client retains control of its own estate at the end of the term. A managed service that cannot be exited cleanly has transferred dependency rather than reduced risk.

5.3 Standards and frameworks that govern managed services

Managed services are not governed by a single document. ISO/IEC 20000-1:2018 and ITIL 4 are complementary rather than competing: the standard states auditable requirements for a service-management system and is certifiable, while ITIL 4 supplies practice-level guidance on how those

requirements are met in day-to-day operation. A provider may adopt ITIL practices without certification, or hold certification while operating its practices poorly; maturity is demonstrated by evidence in both directions.

Managed operations also intersect four adjacent control domains. Information-security management requirements are specified in ISO/IEC 27001:2022[3]; outcome vocabulary for the govern, identify, protect, detect, respond and recover functions is provided by the NIST Cybersecurity Framework 2.0[4]; governance and management objectives for enterprise information and technology are set out in COBIT 2019[5]; and service-continuity commitments depend on the business-continuity management requirements of ISO 22301:2019[6]. A managed-service design that cites only a service-management reference will meet its process obligations while leaving its security, governance and continuity obligations unstated.

Reference	Nature	Contribution to a managed service
ISO/IEC 20000-1:2018	Certifiable requirements	Service-management system: planning, design, transition, delivery, measurement, improvement
ITIL 4	Best-practice guidance	Practice-level guidance for service level management, incident, problem, change, monitoring and event management, continual improvement
ISO/IEC 27001:2022	Certifiable requirements	Information-security management system governing access, logging, supplier and operational security
NIST CSF 2.0	Outcome framework	Common language for govern, identify, protect, detect, respond and recover outcomes
NIST SP 800-61r3	Guidance	Incident-response activity placed inside cybersecurity risk management
COBIT 2019	Governance framework	Governance and management objectives, and the separation of governance from management
ISO 22301:2019	Certifiable requirements	Business-continuity requirements underpinning recovery and continuity commitments
NCA ECC / DCC	National controls	Mandatory control baseline for logging, access, configuration and data handling in the

[3] International Organization for Standardization, ISO/IEC 27001:2022, Information security, cybersecurity and privacy protection — Information security management systems — Requirements. Geneva: ISO/IEC, 2022.

[4] National Institute of Standards and Technology, The NIST Cybersecurity Framework (CSF) 2.0, NIST CSWP 29. Gaithersburg, MD: NIST, February 2024.

[5] ISACA, COBIT 2019 Framework: Governance and Management Objectives. Schaumburg, IL: ISACA, 2018.

[6] International Organization for Standardization, ISO 22301:2019, Security and resilience — Business continuity management systems — Requirements. Geneva: ISO, 2019.

Reference	Nature	Contribution to a managed service
		Kingdom

The framework proposed in clause 9 is therefore not an alternative to these references. It is an operating synthesis of them, expressed in the sequence in which a managed service actually performs the work.

6 Literature review

6.1 Service management

The service-management model covers planning, design, transition, delivery, monitoring and continual improvement as set out in the standard cited above, and its central implication for managed services is that delivery should be governed through repeatable process rather than through individual technical action. A service-management model accordingly defines service scope, responsibilities, service requirements, performance indicators, incidents, changes, risks and improvements.

ITIL 4 expresses the same intent as a service value system in which demand and opportunity are converted into value through the service value chain, supported by thirty-four management practices and directed by seven guiding principles — focus on value, start where you are, progress iteratively with feedback, collaborate and promote visibility, think and work holistically, keep it simple and practical, and optimise and automate. Three of those principles carry particular weight in managed operations. Think and work holistically warns against optimising a single technical tower while the end-to-end service degrades. Progress iteratively with feedback describes how observability data should be used, which is to change the operating model rather than only to populate a report. Optimise and automate states the sequence explicitly: optimisation of the process precedes its automation, never the reverse.

The practices most directly relevant to a managed-service provider are service level management, monitoring and event management, incident management, problem management, change enablement, service configuration management, service continuity management, information security management, knowledge management, service desk, supplier management and continual improvement. This paper treats those practices as the vocabulary of the analysis in clause 8, while remaining vendor-neutral and framework-neutral in the framework it proposes.

6.2 Continual improvement

Continual improvement is treated as a core element of the service-management system[7], which produces an important distinction. A provider may meet every service-level target while failing to reduce recurring technical problems. A mature model therefore asks two questions rather than one: whether the service was restored, and why the failure occurred at all.

6.3 Cybersecurity incident response

NIST finalised SP 800-61 Revision 3 in April 2025. The publication integrates incident response into broader cybersecurity risk management and emphasises the effectiveness of detection, response and recovery activity[8]. The relevance to managed services is direct: cybersecurity incidents do not occupy a separate operational universe. Managed services may contribute detection, triage, escalation, containment support, recovery, evidence preservation and service restoration, and operational and security processes therefore require defined coordination rather than goodwill.

6.4 Zero trust and managed operations

Zero trust withholds implicit trust on the basis of network location, and requires authentication and authorisation before access to enterprise resources[9]; the same principle has been extended to application and service identities in cloud-native and multi-cloud environments[10]. For a managed-service provider this bears on privileged access, remote administration, service accounts, third-party access, administrator identity, device trust and the operational tooling itself. Managed-service access should be governed as an identity and resource problem rather than as network connectivity.

6.5 The national control context

Managed operations in the Kingdom also sit inside a defined control environment. Controls issued by the National Cybersecurity Authority bear on logging, access management, configuration and change control, and on the handling of data within managed environments[11], alongside the essential

[7] International Organization for Standardization, ISO/IEC 20000 IT Service Management — A Practical Guide. Geneva: ISO, 2019.

[8] A. Nelson, S. Rekhi, M. Souppaya and K. Scarfone, Incident Response Recommendations and Considerations for Cybersecurity Risk Management: A CSF 2.0 Community Profile, NIST Special Publication 800-61 Revision 3. Gaithersburg, MD: National Institute of Standards and Technology, April 2025.

[9] S. Rose, O. Borchert, S. Mitchell and S. Connelly, Zero Trust Architecture, NIST Special Publication 800-207. Gaithersburg, MD: NIST, 2020. DOI: 10.6028/NIST.SP.800-207.

[10] R. Chandramouli and Z. Butcher, A Zero Trust Architecture Model for Access Control in Cloud-Native Applications in Multi-Cloud Environments, NIST Special Publication 800-207A. Gaithersburg, MD: NIST, 2023.

[11] National Cybersecurity Authority, Data Cybersecurity Controls. Riyadh: NCA, Kingdom of Saudi Arabia, 2022.

cybersecurity controls[12]. For a provider operating on behalf of client organisations, these are contractual and architectural inputs to the service design rather than obligations discovered at audit.

6.6 Service-level management and reliability engineering

Two bodies of practice inform how service levels should be constructed. The ITIL 4 service level management practice[13] sets clear, business-based targets for service utility, warranty and experience, and requires that service delivery and use be assessed, monitored and managed against those targets. The emphasis on business-based targets is the operative part: a target that measures a component rather than a business outcome will be met while the consumer of the service continues to experience failure.

Site reliability engineering[14] supplies the measurement discipline. It separates the service level indicator, which is the measured quantity, from the service level objective, which is the internal target for that indicator, and from the service level agreement, which is the external commitment with consequences attached. It also introduces the error budget — the permitted shortfall implied by an objective — as an operational control that governs the rate of change to a service. The construct is directly applicable to managed services: an internal objective set tighter than the contractual agreement creates the margin in which degradation can be corrected before a breach occurs, and error-budget consumption is a leading indicator that a change programme is outrunning the reliability of the estate.

Experience-level measurement extends the same logic to the consumer's perception of the service, using digital-experience and user-journey indicators in place of component availability. It does not replace technical targets; it prevents a service from reporting health that its users do not recognise.

7 Research methodology

The paper applies a structured service-maturity and operational-risk methodology in six stages. The analytical vocabulary is taken from the service-management system requirements of ISO/IEC 20000-1:2018 and the management practices of ITIL 4, with cybersecurity, zero-trust and national-control references applied where operations intersect security. No single

[12] National Cybersecurity Authority, Essential Cybersecurity Controls (ECC). Riyadh: NCA, Kingdom of Saudi Arabia. Edition and year of issue to be confirmed before publication.

[13] PeopleCert, ITIL 4 Practice Guide: Service Level Management. London: PeopleCert International (AXELOS), current edition 2025.

[14] B. Beyer, C. Jones, J. Petoff and N. R. Murphy (eds.), Site Reliability Engineering: How Google Runs Production Systems. Sebastopol, CA: O'Reilly Media, 2016.

framework is adopted as authoritative; each stage draws on the reference that governs the activity under examination.

Stage 1 — Service lifecycle decomposition

The lifecycle was decomposed into definition, transition, operation, measurement and improvement.

Each stage was examined for its inputs, its controls and its evidence. Definition takes the service catalogue, business-criticality ratings and contractual scope as inputs, and produces the service model, the responsibility matrix and the agreed target set. Transition takes that model and produces configuration baselines, monitoring instrumentation, runbooks and a tested escalation tree. Operation consumes signals and produces incident, event, change and request records. Measurement converts those records into performance against target, and improvement converts performance into a change to the service design. A stage that produces no evidence cannot be audited, and a stage whose evidence is never read produces no improvement.

Stage 2 — Operational signal identification

Signals were grouped into availability, performance, events, incidents, capacity, security, user experience, configuration and service dependencies.

For each group the study recorded the source system, the collection method, the retention period required to support trend analysis, and the decision the signal is intended to inform. This last criterion proved the most discriminating: a signal that informs no decision is cost without control. Availability and performance signals inform restoration and service-level reporting; capacity signals inform procurement and architecture; configuration and dependency signals inform diagnosis and change risk; security signals inform escalation into the cybersecurity process; user-experience signals inform whether the technical picture and the business picture agree.

Stage 3 — Failure and incident analysis

The study examined how operational failures are detected, categorised, prioritised, diagnosed, resolved and escalated.

Failures were classified by origin — change-induced, capacity-induced, configuration drift, dependency failure, third-party failure, security-induced and latent defect — because origin determines the preventive control that would have avoided the failure. Change-induced failure is addressed by

change control and post-implementation verification, not by faster restoration; capacity-induced failure is addressed by forecasting; dependency failure is addressed by architecture and supplier management. Detection latency was recorded separately from restoration duration, since the two are governed by different capabilities.

Stage 4 — Repetition analysis

Recurring events and incidents were examined to establish whether automation or preventive action could reduce recurrence, and which of the two is appropriate.

Repetition was measured on three axes: frequency, effort per occurrence and variance in handling. High frequency with low variance indicates an automation candidate. High frequency with high variance indicates a process or knowledge gap that must be closed before automation is attempted. Low frequency with high impact indicates a candidate for a rehearsed runbook rather than for automation, because rarely used automation decays silently between uses.

Stage 5 — Automation mapping

Operational tasks were categorised according to their suitability for automation, which is not the same as their frequency.

Suitability was scored against six criteria: whether the trigger is deterministic, whether the outcome is verifiable, whether the action is reversible, whether the permissions required are narrowly scoped, whether the task is idempotent when repeated, and whether the failure mode of the automation itself is contained. Tasks meeting all six are automation candidates. Tasks failing reversibility or containment remain human-authorised regardless of how repetitive they are, which is the distinction developed in clause 8.10.

Stage 6 — Maturity framework construction

The findings were consolidated into the framework presented in clause 9.

Consolidation applied two tests to each candidate principle. First, whether the principle is observable — whether an assessor could find evidence of it in a live service without relying on assertion. Second, whether its absence is consequential — whether services lacking it exhibit a recognisable failure pattern. Principles that were merely good practice, without observable evidence or a distinct failure signature, were excluded so that the framework remains an assessment instrument rather than a checklist.

8 Technical analysis: sixteen operating principles

8.1 Define the service before managing it

A managed service requires a defined scope: supported assets, applications, locations, service hours, responsibilities, dependencies, exclusions, escalation paths and service-level targets. Without a service boundary, operational accountability is ambiguous and every dispute becomes a matter of interpretation.

A service cannot be measured reliably until its scope is defined.

8.2 Service-level management

Service-level management converts operational expectation into measurable target — availability, incident response time, restoration time, resolution time, change success, service performance, backup success and capacity thresholds.

Compliance with those targets should not become the sole measure of quality, because an environment may meet every threshold while repeatedly suffering avoidable incidents. Service-level compliance measures performance against an agreed threshold; it does not by itself demonstrate operational maturity.

8.2.1 The three-layer target structure

A defensible service-level construct has three layers, and conflating them is the most common cause of unmeasurable agreements. The service level indicator is the measured quantity — the percentage of successful transactions, the availability of a business service, the time to first meaningful response. The service level objective is the internal target for that indicator, set deliberately tighter than the contract so that the provider detects drift before the client does. The service level agreement is the external commitment, and it is the only layer that carries consequence. Beneath all three sit the operational-level agreements with internal teams and the underpinning contracts with suppliers, carriers and cloud providers on which the external commitment actually depends.

The rule that follows is simple to state and frequently ignored: no external target may be tighter than the weakest underpinning commitment supporting it. A four-hour restoration target for a device whose vendor contract provides next-business-day hardware replacement is not a service level but an expression of hope.

8.2.2 Constructing a target that can be measured

Every target requires five components before it can be operated: the indicator, the calculation, the measurement source, the measurement window, and the exclusions. Availability, for example, is meaningless until it is stated whether it is measured at the component, the business service or the user journey; whether the calculation is agreed uptime divided by agreed service hours or by calendar time; whether the source is the monitoring platform, the service-management tool or a synthetic probe; whether the window is calendar month, rolling thirty days or quarter; and which periods are excluded — approved maintenance windows, client-caused outages, force majeure and failures in third-party services outside the provider's control.

Exclusions should be enumerated rather than described in general terms, because an undefined exclusion is resolved during a dispute rather than during design. The same applies to the measurement source: where the client and the provider measure the same indicator with different instruments, the agreement should name the authoritative source in advance and state how discrepancies are reconciled.

Service-level metric	Definition	Illustrative target	Measurement source
Business-service availability	Agreed uptime as a percentage of agreed service hours, measured at the business service	99.5 – 99.9 % monthly	Monitoring platform / synthetic probe
Time to detect	Interval from onset of abnormal behaviour to alert raised	≤ 5 minutes for critical services	Monitoring and event platform
Time to respond	Interval from alert or ticket to acknowledged ownership by a qualified engineer	15 min (P1) – 8 h (P4)	Service-management tool
Time to restore	Interval from detection to restoration of agreed service, workaround permitted	4 h (P1) – 5 business days (P4)	Service-management tool
Time to resolve	Interval to permanent removal of cause, tracked as a problem record	Agreed per problem record	Problem management records
First-time-fix rate	Incidents resolved at first tier without escalation	≥ 65 %	Service-management tool
Change success rate	Changes	≥ 95 %	Change records

Service-level metric	Definition	Illustrative target	Measurement source
	implemented without incident, rollback or emergency remediation		
Backup and recovery success	Successful backup jobs and verified restore tests in the period	≥ 99 % jobs; quarterly restore test	Backup platform / test evidence
Recurring-incident ratio	Proportion of incidents matching an existing known-error record	Declining trend period on period	Problem and known-error records
Automation coverage	Proportion of eligible repetitive tasks executed by automation with logging and rollback	Increasing trend period on period	Automation platform logs
Experience level (XLA)	User-perceived performance of a defined journey, sampled continuously	Agreed per journey	Digital-experience monitoring

The target values above are illustrative of common commercial practice and are not drawn from any standard. Targets should be derived from business impact, criticality and the underpinning supplier commitments available for each service.

8.2.3 Priority, response and restoration

Priority is not the same as severity, and neither is the same as the order in which work is convenient. Priority should be derived from a published matrix of business impact against urgency, so that classification is repeatable between engineers and between shifts. The matrix should be agreed with the client, applied at the point of detection as well as at the point of report, and reviewed when the service catalogue changes.

The clock rules matter as much as the targets. An agreement should state when each clock starts — at detection by the monitoring platform, or at receipt of a user report — because a provider that monitors proactively will otherwise be penalised for finding faults early. It should state whether the clock runs against calendar time or agreed service hours, whether it pauses while the provider is legitimately waiting for client input, vendor dispatch or a scheduled maintenance window, and how the pause is evidenced. It should also distinguish restoration, which may rely on a workaround, from resolution, which removes the cause and belongs to problem management as clause 8.8 sets out.

8.2.4 Commercial consequence and chronic failure

Service credits give a target its force, and their structure should reflect that they are a governance mechanism rather than compensation. Four elements are usually required: the credit scale applied per breach or per band of shortfall, the aggregate cap in any period, whether credits may be earned back through sustained performance, and the chronic-failure threshold at which repeated breaches become a termination or step-in right rather than another credit. A provider that pays credits and continues to breach is not being governed; it is being invoiced.

The counterpart obligation belongs to the client. Service levels are conditional on client-side dependencies — access, approvals, change windows, environment information, third-party escalation and asset accuracy — and those dependencies should be stated as client responsibilities with their own timescales. An agreement that imposes measured obligations on one party and unmeasured expectations on the other will be renegotiated at the first serious incident.

8.2.5 From service levels to experience levels

Component-level targets can all be met while the business service is unusable, which is why experience-level measurement has become part of mature service-level design. An experience-level target measures a defined user journey — authentication, transaction submission, report generation, remote-desktop responsiveness — and samples it continuously from the position of the user rather than from the position of the infrastructure. Experience-level agreements should supplement technical targets rather than replace them, because a journey measurement identifies that the service is failing without identifying which component is responsible.

8.2.6 Service-level reporting and governance cadence

A service level that is not reported is not managed. The reporting construct should operate on four cadences: an operational review each day covering incidents, alerts and pending changes; a weekly review covering trends, backlog and outstanding problem records; a monthly service review covering performance against every agreed target, breaches with cause, credit position, capacity and risk; and a quarterly service review covering the service improvement plan, cost, roadmap alignment and forward risk. Each cadence answers a different question, and collapsing them into a single monthly report removes both the operational detail and the strategic view.

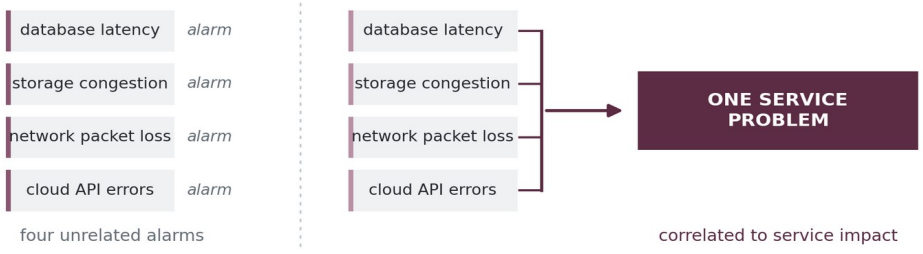
The report itself should carry more than compliance percentages. A mature service-level report states performance against target, the cause of every

breach, the improvement actions arising, the status of open problem records, the automation delivered in the period, capacity and headroom trends, security-relevant events and coordination activity, and the current service risks with owners and dates. That report is the primary evidence of continual improvement under both ISO/IEC 20000-1 and the ITIL continual-improvement practice, and it is the artefact by which a client can distinguish a service that is managed from a service that is merely staffed.

8.3 Observability

Monitoring answers whether a known metric has left its threshold. Observability attempts the broader question of what is happening inside the service, and why, by correlating metrics, logs, traces, events, configuration and dependency data. Figure 2 shows the practical difference.

Figure 2 — Four alarms, or one service problem



Correlation across metrics, logs, traces, events, configuration and dependency data is what converts signals into an actionable service diagnosis.

A siloed model produces four unrelated alarms from one underlying condition; an observability model interprets them as a single service problem. The difference is not the number of tools but whether the signals are related to a service.

Observability rests on five telemetry classes, and a gap in any one of them limits what the others can explain. Metrics give aggregated numeric state over time and answer how much and how often. Logs give discrete event records and answer what happened. Traces follow a single transaction across components and answer where the time went. Events record state change deployments, configuration edits, failovers — and answer what altered the system. Synthetic and real-user measurement answer whether the service works from the position of the consumer. A managed service instrumented on metrics alone can detect that a threshold moved but cannot explain why, which is the difference between an alarm and a diagnosis.

Four signals are worth monitoring for every service regardless of technology: latency, throughput, error rate and saturation. Latency should be reported at percentiles rather than as an average, because an average conceals the minority of slow transactions that generate the complaints. Error rate should be expressed against total requests rather than as an absolute count. Saturation should be measured against the resource that binds first — connections, IOPS, threads, licences — which is rarely the resource most easily graphed.

Three practical constraints govern observability design. Coverage: every component carrying a service-level obligation must be instrumented, and the gap list should be a named risk rather than an assumption. Retention: trend analysis, capacity forecasting and problem management require history, so retention periods should be derived from the longest analytical window required, not from default tool settings. Signal volume: high-cardinality data is expensive to collect and slow to query, so instrumentation should be deliberate. Observability is an engineering design decision, not a licensing decision.

8.4 Service dependency mapping

A business service rarely depends on one system. It may run through a user channel, an application, a database, a network, storage, an identity service and infrastructure beneath all of them. Managed services should hold an understanding of those relationships, because without dependency context an operations team can know that a device has failed without knowing which business services are affected.

The objective therefore shifts from device monitoring to service-aware monitoring, and the shift is one of information rather than of instrumentation.

A usable dependency model records four relationships for each business service: the components it consumes, the direction and criticality of each dependency, the redundancy present at each layer, and the external parties involved. Three sources build it, and all three are needed. Automated discovery finds what is technically connected. Configuration and change records state what should be connected. Practitioner knowledge explains what matters — which of the connections carries the business transaction and which is incidental.

The model earns its cost at three moments. During an incident it converts a component alarm into a statement of business impact, which is what allows

prioritisation to be defended. During change assessment it defines the blast radius of a proposed change, which is what allows risk to be classified rather than guessed. During capacity and continuity planning it identifies single points of failure and concentration risk, including shared dependencies that appear independent on an architecture diagram but resolve to the same physical or contractual source.

A dependency model decays. It should be maintained as an output of change management rather than as a periodic project, because a model that is accurate at onboarding and unmaintained thereafter is more dangerous than no model at all — it invites confident decisions on stale information.

8.5 Event management

Not every event is an incident. Events include normal state changes, warnings, capacity thresholds, failures and security alerts, and an effective model filters and correlates them. Without filtering, operations teams experience alert overload and begin to discount the alerts that matter. The objective is not more alerts but more actionable information.

Event handling should apply a defined sequence: collection, normalisation into a common format, deduplication of repeats, suppression of events from components already known to be down or inside a maintenance window, correlation into a single service-level condition, enrichment with dependency, ownership and runbook context, and only then notification. Every alert that reaches a human should carry three things — what is affected in service terms, what the expected action is, and where the procedure for that action is recorded.

Alert quality is measurable, and should be measured. Useful indicators include the proportion of alerts that result in an action, the proportion closed as noise, the ratio of alerts to incidents, the number of alerts per engineer per shift, and the proportion of incidents detected by monitoring rather than reported by a user. That last ratio — the proactive detection rate — is the clearest single indicator of whether a managed service operates ahead of its users or behind them.

Thresholds deserve as much design attention as the alerts they raise. Static thresholds suit binary and contractual conditions; rate-of-change thresholds detect degradation earlier than absolute limits; baseline-relative thresholds accommodate services with strong daily or seasonal patterns. Any alert

routinely acknowledged without action for a month should be redesigned or retired, because a monitoring configuration that trains its operators to ignore it has itself become a service risk.

8.6 Incident management

Incident management restores normal service as efficiently as possible, through a lifecycle of detection, recording, classification, prioritisation, diagnosis, resolution, restoration and closure. Severe incidents attract additional governance: incident command, stakeholder communication, technical workstreams, executive escalation and recovery coordination.

Incident handling should remain focused on restoration. Eliminating the cause belongs to problem management, and conflating the two delays the first without reliably achieving the second.

Priority should be derived from impact and urgency rather than from the identity of the caller, and the resulting classes should carry defined behaviour rather than only defined timescales. A P1 implies continuous work, a named incident manager, a communications cadence and executive visibility; a P3 implies scheduled work within business hours. Where those behavioural differences are not written down, priority becomes a negotiation at the moment of highest pressure.

Major incidents require an explicit command structure, and the roles should be separated even when the team is small: an incident manager who owns coordination and communication but performs no technical work, technical leads per workstream, a scribe who maintains the timeline, and a business liaison who translates technical state into service impact. Communication should run to a published cadence — an initial notification within a defined period of declaration, updates at fixed intervals whether or not there is progress, and a closure notice stating impact, cause where known, and the follow-up actions raised.

Every major incident should produce a written post-incident review within a defined period, covering the timeline, the detection path, the decisions taken, what delayed restoration, and the actions arising with owners and dates. The review should be blameless in method and specific in output: its purpose is to generate problem records, monitoring changes, runbook updates and architectural findings. A review that produces narrative without actions has documented an outage rather than learned from it.

8.7 Cybersecurity incident integration

Cybersecurity incidents frequently begin as operational events — unusual authentication, unexpected configuration change, abnormal traffic, an endpoint alert, unexplained service degradation. The current incident-response guidance discussed in clause 6.3 emphasises integrating response across cybersecurity risk-management activity, which for a managed service means defined integration with security teams and defined criteria by which an operational incident carrying security indicators is escalated. Criteria written in advance are the only ones that operate under pressure.

The escalation criteria should be enumerated, not described. Indicators that ordinarily justify immediate security escalation include authentication anomalies such as impossible-travel patterns or repeated privilege failures, unexplained privilege escalation or newly created administrative accounts, configuration change outside the change record, unexpected outbound traffic or connections to unfamiliar destinations, endpoint detection alerts on servers, disabled logging or security agents, unexplained encryption or mass file modification, and any service degradation coinciding with one of the above.

Operational behaviour changes once a security indicator is present. Actions that are routine in an availability incident — rebuilding a host, clearing logs, restarting a service, restoring from backup — can destroy evidence or reinfect the environment. The managed-service procedure should state which actions are suspended pending security triage, how volatile evidence is captured before remediation, who authorises containment that will itself cause an outage, and how the incident record is handled where confidentiality is required.

Coordination should be rehearsed rather than assumed. A joint exercise between operations and the security function, run at least annually against a realistic scenario, tests the handover under the only conditions that matter: time pressure and incomplete information. The output of that exercise is a shorter escalation path, not a certificate.

8.8 Problem management

Incident management asks how service is restored; problem management asks why the incident occurred. A recurring incident should generate analysis — trend analysis, root-cause analysis, known-error records, dependency

analysis, configuration review — and the purpose of that analysis is to convert operational history into preventive action rather than into a report.

Problem records should be opened from defined triggers rather than at discretion: any major incident, any incident recurring above an agreed threshold within a period, any incident restored by an unexplained workaround, and any near-miss detected before user impact. Without triggers, problem management becomes an activity that happens when the queue is quiet, which is precisely when it is least needed.

Analysis technique should match the failure. Chronological timeline reconstruction establishes sequence and detection latency. Five-whys and causal-chain analysis suit single-path failures. Fault-tree or ishikawa analysis suits failures with multiple contributing conditions. Pareto analysis across the incident history identifies where the recurring volume actually sits, which is frequently not where the attention is. Change correlation tests the most common single cause, since a substantial proportion of incidents follow a change.

Output discipline matters more than technique. Each closed problem record should state the cause, the corrective action, the preventive action, the owner and date, and the verification method — how the organisation will know the failure has stopped recurring rather than merely stopped being reported. Known-error records should carry the symptom, the diagnostic test that confirms it, the approved workaround and its expected duration, so that the next occurrence is a two-minute recognition rather than a fresh investigation.

8.9 Automation

Managed services contain a large volume of repetitive activity: health checks, ticket enrichment, alert correlation, service restarts, log collection, report generation, account workflows, backup validation, configuration checks and common remediation. Automation delivers speed, repeatability and reduced manual variation across all of them.

> **Automation also carries risk. Poorly controlled automation executes the wrong action quickly, consistently and at scale.**

Automation therefore requires defined triggers, authorisation, testing, rollback, logging and exception handling. An automated action without a rollback path is an unreviewed change with a scheduler attached.

Automation maturity progresses through five recognisable levels, and each level should be earned rather than skipped. Level one is scripted assistance, where an engineer runs a tested script manually. Level two is scheduled execution of routine tasks. Level three is triggered diagnostic automation, which gathers evidence and enriches records but changes nothing. Level four is triggered remediation within a narrow, reversible scope. Level five is closed-loop remediation with automated verification and automatic rollback on failed verification. Most managed services derive the majority of their benefit at levels two and three, where risk is minimal and the volume of manual effort removed is largest.

Six guardrails should be present before any automation is permitted to change state. A scoped service account with the minimum permissions required and no interactive login. Idempotency, so that repeated execution is harmless. A blast-radius limit, capping how many objects a single run may affect before it stops and escalates. Verification of the intended outcome after execution. An automatic rollback path, with the rollback itself tested. And complete logging of trigger, decision, action, result and identity, retained for the same period as change records.

Automation should also be governed as a change. Each automated action is a standing change to the environment, so it belongs in the configuration record, carries a version, has a named owner, is reviewed when the platform it acts on changes, and is retired deliberately rather than left running unattended. Unowned automation is the most quietly dangerous asset in an operations estate.

8.10 Human-in-the-loop operations

Not every decision should be automated. Actions that are irreversible, that affect several services at once, that touch security boundaries, or that depend on business context should retain human authorisation, while diagnosis, enrichment and evidence gathering can be automated freely because they change nothing. The distinction to hold is between automating information and automating action.

A practical test for the boundary asks four questions of any candidate action. Is the action reversible within the recovery objective? Is its effect confined to one service? Does it require business context — timing, commercial consequence, regulatory obligation — that the automation does not hold? And would a wrong execution be detected quickly? Two or more unfavourable

answers indicate that the action should be prepared automatically and executed on human authorisation, which captures most of the speed benefit while retaining accountability.

The pattern that works in practice is automated preparation with human commitment: the automation detects, gathers evidence, identifies the probable cause, drafts the remediation and presents it with a single approval step. Detection-to-decision time falls sharply, while the decision itself, and the accountability for it, remain with a named engineer.

8.11 Capacity and performance management

Capacity failures are among the most predictable of all operational failures, because they arrive along a trend rather than as an event. Managed services should track utilisation, growth, saturation points and performance headroom, and should treat a capacity threshold as a lead indicator rather than as an alarm to be silenced.

Capacity management requires four measurements per resource: current utilisation, the trend over a period long enough to distinguish growth from noise, the saturation point at which performance degrades rather than the point at which the resource is exhausted, and the lead time required to add capacity. The last is the one most often omitted and the one that determines how early a forecast must be produced: a resource with a twelve-week procurement lead time must be forecast at least a quarter ahead, whatever its current utilisation.

Forecasting need not be sophisticated to be useful. Linear projection of observed growth, seasonal adjustment for known business cycles, and event-driven modelling for planned changes such as a migration, an acquisition or a campaign will cover most cases. Thresholds should be set to trigger a planning action rather than an alarm — a documented practice is to plan at a defined utilisation, procure at a higher one, and treat arrival at the saturation point as a service failure that should never have been reached.

Capacity should also be read as a performance discipline, not only a volumetric one. Response-time degradation under load, queue depth, connection saturation, licence exhaustion, thread-pool starvation and storage-latency inflation are all capacity conditions that appear long before any resource reads as full. Reporting capacity purely as percentage utilisation

of processor, memory and disk is the most common reason a service is surprised by a predictable failure.

8.12 Configuration and change management

A substantial proportion of incidents originate in change. Configuration records, change approval, testing, scheduling, rollback and post-change verification are therefore operational controls, and a service without configuration records cannot reliably determine whether an environment is in its approved state — which makes every diagnosis an investigation from first principles.

Change should be classified before it is assessed, because the class determines the control. A standard change is pre-approved, low-risk and repeatable, and should be executed without ceremony. A normal change requires assessment, scheduling and authorisation proportionate to its risk. An emergency change compresses the approval path but not the record, and should always generate a retrospective review. Where every change follows the same heavyweight route, the routine work slows down and the risky work receives no additional scrutiny.

Risk assessment should be evidence-based rather than declarative. The dependency model states what the change can affect; the incident history states what has failed before under similar conditions; the configuration baseline states whether the target is in its approved state. Each change should carry a defined test approach, a rollback plan with a stated time to execute, a verification step performed after implementation, and a named person accountable for the outcome — not merely for the approval.

Configuration control is the counterpart discipline. A managed service should hold a baseline for each managed platform, detect drift from that baseline on a defined cycle, and treat unexplained drift as a security and reliability finding rather than as an administrative exception. Post-change verification should confirm both that the intended change took effect and that the service returned to its expected performance envelope, because a change that succeeds technically and degrades the service has still failed.

8.13 Knowledge management

Operational knowledge held only by individuals is a service risk. Known errors, diagnostic procedures, runbooks, escalation criteria and resolution history should be recorded so that service quality does not depend on which engineer

is on shift. Knowledge management is what makes a managed service transferable, and transferability is what makes it a service rather than an arrangement.

A runbook is only useful if it is written to be executed under pressure by someone who did not write it. A workable standard states the service and component affected, the symptom and the diagnostic test that confirms it, the prerequisites and access required, the steps in order with expected output at each step, the verification that the action worked, the rollback if it did not, and the escalation path with named criteria. Runbooks should be versioned, owned, and reviewed on a defined cycle and after every incident that exposed a gap in them.

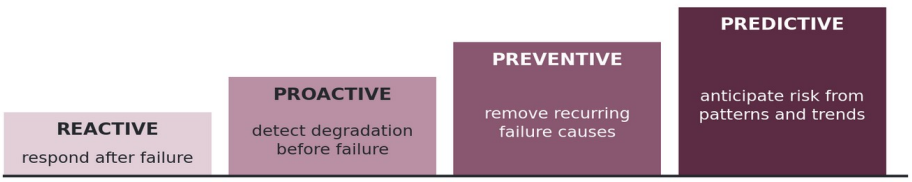
Knowledge requires a lifecycle, not a repository. Articles should be created from incident and problem output, reviewed for accuracy on a schedule, measured by use — how often an article is opened, and how often it resolved the case — and retired when the platform they describe is decommissioned. Two indicators reveal whether the discipline is real: the proportion of incidents resolved using an existing article, and the median time for new operational knowledge to reach the knowledge base after the incident that produced it.

Knowledge management is also the practical test of service transferability. If a competent engineer unfamiliar with the account cannot operate the service from the recorded material, the service depends on individuals rather than on process — which is a continuity risk carried by the client as much as by the provider.

8.14 From reactive to predictive operations

The preceding principles describe a progression rather than a menu. Figure 1 sets out the four maturity stages and the behaviour that characterises each.

Figure 1 — Managed-service maturity stages



Each stage rests on the one before it. Predictive capability is an advanced stage built on the others, never a substitute for them.

Predictive operation is the most demanding stage because it depends on accurate historical and real-time data rather than on tooling. An organisation

that has not established visibility cannot predict, whatever analytics it has purchased.

The four stages can be distinguished by observable evidence rather than by claim. A reactive service detects through user reports, measures itself by ticket volume and closure, and holds knowledge in individuals. A proactive service detects through monitoring, measures service-level performance, and holds documented procedures. A preventive service analyses recurrence, operates problem management with corrective and preventive actions, forecasts capacity, and can demonstrate a declining recurrence trend. A predictive service holds sufficient historical and real-time data to model failure conditions, and intervenes on trajectory before a threshold is crossed.

Progression is constrained by data rather than by ambition. Predictive operation requires instrumentation coverage across the service, retention long enough to contain the pattern being predicted, consistent event and incident classification so that history is comparable, an accurate dependency model so that correlation is meaningful, and a feedback loop in which predictions are scored against outcomes. An organisation that has not established these conditions will not obtain prediction by purchasing analytics; it will obtain a dashboard whose confidence is unearned.

The honest position for a provider is therefore stage-aware. Each stage has a defined entry condition, and the value of naming the current stage is that it identifies the next capability to build rather than the next product to buy.

8.15 Run-book automation as the operational baseline

Run-book automation is the codified execution of a documented operational procedure by a platform rather than by a person. It is the practical bridge between the automation principle in clause 8.9 and the knowledge principle in clause 8.13: the runbook states what should be done, and the automation executes it identically every time, under identity control, with a complete record of what it did. Run-book automation is therefore not a tool category but an operating baseline — the point at which routine operational work stops depending on which engineer is on shift.

Three adjacent technologies are frequently conflated with it and should be distinguished, because each carries a different failure mode. Run-book automation drives systems through their supported interfaces — application programming interfaces, management endpoints, orchestration platforms — and fails visibly when an interface changes. Robotic process automation

drives human interfaces instead, replaying keystrokes and screen interactions; it is legitimate where no programmable interface exists, but it is brittle, breaks silently on a screen change, and should be treated as a temporary bridge rather than an architecture. Infrastructure as code declares a desired state and reconciles the environment toward it, which makes it the correct instrument for provisioning and configuration rather than for event response. A mature managed service uses all three deliberately and does not substitute one for another.

The work that repays automation earliest in a managed service is consistent across environments: alert triage and enrichment, scheduled health verification, evidence and log collection at the moment of an incident, ticket classification and routing, restart or failover of stateless components, certificate and licence renewal, patch orchestration within approved windows, backup and restore verification, account and access workflows within an approved catalogue, standard changes, and the assembly of service-level and capacity reporting. None of these require judgement; all of them consume disproportionate engineer time and all of them are performed inconsistently when performed by hand.

Eight design rules make run-book automation safe enough to trust. Write and prove the procedure before automating it, because automation encodes a process rather than improving it. Parameterise rather than hard-code, so one automation serves many targets. Implement pre-checks that confirm the environment is in the expected state before acting, and post-checks that confirm the intended outcome afterwards. Provide a dry-run mode that reports what would change without changing it. Make execution idempotent. Hold credentials in a secrets vault and execute under a least-privilege service account, consistent with the zero-trust position in clause 6.4. Throttle scope so a single run cannot exceed a defined blast radius. And log trigger, decision, action, outcome and identity to the retention period applied to change records.

Governance is what separates an automation estate from an accumulation of scripts. Each automation should appear in a catalogue with an owner, a version, a purpose, the systems it touches, its authorisation model, its rollback path and a review date. Changes to automations follow the change process, because an automation is a standing change to the environment. Automations whose platform has been replaced, whose owner has left, or which have not

executed successfully within a defined period should be retired deliberately. Unowned automation with valid credentials is one of the more serious latent risks an operations estate can carry.

Progress should be measured, not asserted. Useful indicators are the proportion of eligible repetitive tasks executed by automation, manual effort removed per period, automation success rate, the rate at which rollback is invoked, the rate at which post-execution verification fails, and — the indicator that governs trust — the number of incidents caused by automation. A programme that cannot report the last of these is not yet governing what it has built.

8.16 Applied artificial intelligence in modern operations

Artificial intelligence in managed operations is not a single capability, and treating it as one is the most common cause of disappointed adoption. Three classes of application carry materially different accuracy characteristics, risk profiles and control requirements, and each should be evaluated on its own terms.

The first class is analytic and machine-learning operations intelligence, commonly described as AIOps. It applies statistical and learned models to operational telemetry for anomaly detection against learned baselines rather than static thresholds, event correlation and noise reduction, clustering of related alerts into a single probable condition, retrieval of similar historical incidents with their resolutions, capacity and failure forecasting, and change-risk scoring based on the outcome history of comparable changes. This class is where most managed services obtain measurable value first, because its outputs are quantitative and its performance can be evaluated with precision, recall and lead time.

The second class is generative and assistive intelligence. Applied to operations, it summarises long incident timelines into a stakeholder update, drafts post-incident reviews from the record, explains unfamiliar log or configuration output, drafts runbooks and knowledge articles for human verification, translates technical state into business language for service reporting, and answers operational questions from the provider's own knowledge base and configuration data through retrieval-grounded generation. Its output is a draft, never an authority: every generated artefact that leaves the provider or drives an action should carry a named human reviewer.

The third class is agentic operation, in which a model plans a sequence of steps and invokes tools to execute them. This is the highest-value and highest-risk class, and the safe adoption pattern is asymmetric: grant broad autonomy for read-only diagnosis — querying telemetry, correlating configuration, assembling an evidence pack, proposing a probable cause — and require the same six guardrails and human authorisation from clause 8.9 for any action that changes state. An agent that can gather everything and change nothing is a significant capability with a contained failure mode.

All three classes are constrained by the same precondition, which is data rather than model quality. Instrumentation coverage determines what can be observed; retention determines what patterns can be learned; consistent event, incident and change classification determines whether history is comparable; the dependency model determines whether correlation is meaningful; and recorded outcomes determine whether the system can be scored at all. An estate that has not completed the instrumentation and analysis phases will obtain confident output from incomplete data, which is more dangerous than no output because it will be acted upon.

Operational use case	Suitable technique	Autonomy	Control required
Health checks, evidence collection, report assembly	Run-book automation	Full	Pre/post-checks, logging, catalogue entry
Alert deduplication and correlation	Analytic / ML (AIOps)	Full	Precision and recall monitoring, suppression audit
Anomaly detection against learned baseline	Analytic / ML	Advisory	False-positive tracking, threshold review
Capacity and failure forecasting	Analytic / ML	Advisory	Forecast scored against outcome
Change-risk scoring	Analytic / ML	Advisory	Human authorisation retained (clause 8.12)
Incident summary, stakeholder update, RCA draft	Generative / assistive	Draft only	Named human reviewer before release
Knowledge retrieval and runbook	Generative, retrieval-grounded	Draft only	Grounded in own knowledge base;

Operational use case	Suitable technique	Autonomy	Control required
drafting			citations required
Diagnosis, evidence assembly, probable cause	Agentic, read-only	Broad	No write permissions; full action log
Remediation of a stable, reversible condition	Agentic with approval gate	Gated	Six guardrails of clause 8.9 plus human commitment
Irreversible or multi-service action	Human decision	None	Automated preparation only

Governance should be established before deployment rather than after the first incident. Two references are directly applicable: the NIST AI Risk Management Framework[15], which organises AI risk into the govern, map, measure and manage functions, and ISO/IEC 42001:2023[16], which specifies requirements for an artificial-intelligence management system and can be operated alongside the service-management and information-security management systems already in place. For a managed-service provider operating on client estates, five controls carry most of the weight: a defined lawful and contractual basis for processing client operational data, including whether it may leave the client environment or the Kingdom; data-residency and confidentiality constraints consistent with the national controls discussed in clause 6.5; logging of inputs, outputs and the identity that acted on them; treatment of model, prompt and tool-configuration changes as changes under clause 8.12; and a named human accountable for every AI-influenced decision, because accountability cannot be delegated to a model.

Value should be demonstrated on operational indicators rather than on capability claims. The measures that matter are the change in mean time to detect, the reduction in alert volume reaching a human without a corresponding fall in the proactive detection rate, first-time-fix rate, mean time to restore, the false-positive and false-negative rates of any predictive model, and the proportion of AI-generated drafts accepted by reviewers without material correction. Applications that do not move these numbers within an agreed evaluation window should be retired rather than defended, and the evaluation window should be defined before the pilot begins.

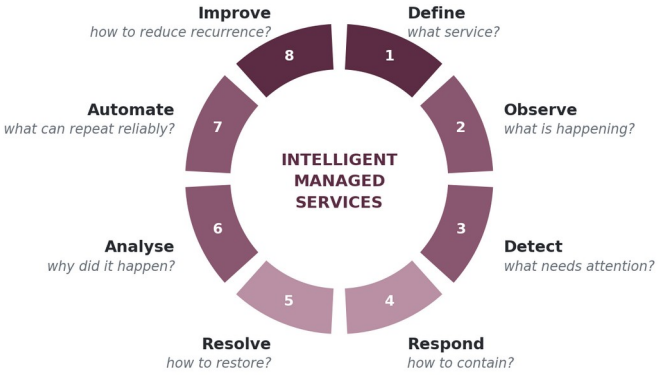
[15] National Institute of Standards and Technology, Artificial Intelligence Risk Management Framework (AI RMF 1.0), NIST AI 100-1. Gaithersburg, MD: NIST, January 2023.
[16] International Organization for Standardization, ISO/IEC 42001:2023, Information technology — Artificial intelligence — Management system. Geneva: ISO/IEC, 2023.

The sequencing conclusion is the same one that governs automation. Applied AI amplifies the instrumentation, analysis and automation phases described in clause 12; it does not substitute for them. Deployed onto an uninstrumented estate with inconsistent records, it produces plausible narrative without operational effect. Deployed onto a service that already observes, classifies, analyses and automates under control, it compresses detection and diagnosis time materially — which is precisely why it belongs at the end of the adoption sequence rather than at the beginning.

9 The Intelligent Managed Services Engineering Framework

The sixteen principles consolidate into eight stages, each carrying a question and a desired outcome. Figure 3 shows the stages as a closed cycle rather than a sequence with an end.

Figure 3 — The eight assurance stages as a closed cycle



The cycle does not terminate at resolution; improvement returns to definition.

STAGE	CORE QUESTION	DESIRED OUTCOME	PRINCIPLES
1 Define	What service is being managed?	Clear accountability	8.1, 8.2
2 Observe	What is happening?	Operational visibility	8.3, 8.4
3 Detect	What requires attention?	Early identification	8.5, 8.11
4 Respond	How should impact be contained?	Controlled response	8.6, 8.7
5 Resolve	How is service restored?	Service recovery	8.6, 8.13

STAGE	CORE QUESTION	DESIRED OUTCOME	PRINCIP LES
6 Analyse	Why did it happen?	Root-cause insight	8.8, 8.12
7 Automate	What can be repeated reliably?	Operational efficiency	8.9, 8.10, 8.15
8 Improve	How can recurrence and risk be reduced?	Continual improvement	8.8, 8.14, 8.16

Expressed as a single statement, intelligent managed services are the combination of service context, observability, response, root-cause learning, controlled automation and continual improvement. This is a conceptual outcome of the present research and is not presented as an official standard or regulatory formula.

9.1 Alignment with ITIL 4 practices and ISO/IEC 20000-1 requirements

The framework is not a substitute for either reference. It is an operating sequence assembled from them, and the mapping below states that relationship explicitly so that a reader already working to ITIL 4 or certified to ISO/IEC 20000-1 can locate each principle inside the practice or clause that governs it. Where a principle has no direct counterpart, that is stated rather than approximated.

Principle	ITIL 4 practice	ISO/IEC 20000-1:2018 requirement
8.1 Define the service	Service catalogue management; business analysis	8.2.4 service catalogue management; 8.2.2 plan the services
8.2 Service-level management	Service level management	8.3.3 service level management; 8.3.4 supplier management
8.3 Observability	Monitoring and event management; measurement and reporting	9.1 monitoring, measurement, analysis and evaluation
8.4 Service dependency mapping	Service configuration management	8.2.6 configuration management; 8.2.5 asset management
8.5 Event management	Monitoring and event management	9.1 monitoring, measurement, analysis and evaluation
8.6 Incident management	Incident management; service desk	8.6.1 incident management, including major incidents
8.7 Cybersecurity integration	Information security management; incident management	8.7.3 information security management; 8.6.1 incident management
8.8 Problem management	Problem management	8.6.3 problem management; 10.1 nonconformity and corrective action
8.9 Automation	Deployment management; infrastructure and platform	8.5.1 change management (automation as a controlled change)

Principle	ITIL 4 practice	ISO/IEC 20000-1:2018 requirement
	management	
8.10 Human-in-the-loop	No dedicated practice — governed by the guiding principle optimise and automate	8.5.1 change management (authorisation requirements)
8.11 Capacity and performance	Capacity and performance management; availability management	8.4.3 capacity management; 8.7.1 service availability management
8.12 Configuration and change	Change enablement; service configuration management; release management	8.5.1 change management; 8.5.3 release and deployment; 8.2.6 configuration
8.13 Knowledge management	Knowledge management	Clause 7 support of the SMS — knowledge and documented information
8.14 Reactive to predictive	Continual improvement; measurement and reporting	9.4 service reporting; 10.2 continual improvement
8.15 Run-book automation	Deployment management; infrastructure and platform management; workforce and talent management	8.5.1 change management; 8.6.2 service request management
8.16 Applied artificial intelligence	No dedicated practice — governed externally by NIST AI RMF 1.0 and ISO/IEC 42001:2023	No direct requirement — governed as change (8.5.1), security (8.7.3) and improvement (10.2)

Three observations follow from the mapping. First, the framework adds no new process vocabulary: every principle from 8.1 to 8.14 has a named counterpart in both references, which is deliberate, since a provider should not be asked to maintain a private taxonomy alongside the standard it is certified against. Second, the framework differs from both references in sequence rather than in content — ITIL 4 organises practices by capability and ISO/IEC 20000-1 organises requirements by management-system clause, whereas the eight stages here are ordered in the sequence in which operational work actually occurs. Third, the two newest principles sit outside the service-management references and are therefore governed by adjacent ones: run-book automation is controlled as change and request fulfilment, and applied artificial intelligence is controlled through the AI management-system and risk-framework references cited in clause 8.16.

For a provider already certified to ISO/IEC 20000-1, the practical use of this mapping is gap analysis in the opposite direction to an audit: the clause is satisfied by definition, and the question is whether the operational behaviour described by the corresponding principle is actually present. Conformity with 9.1 can be demonstrated with a monitoring report; the observability capability described in clause 8.3 is a higher bar than that, and the distinction is where most managed-service improvement is available.

10 Findings

The study produces eight findings.

Finding 1. **Ticket closure is not a measure of service maturity.** A service may close every ticket within target and still be deteriorating, because the measure records response rather than condition.

Finding 2. **Observability is the precondition of proactive operation.** Detection ahead of failure requires signals related to a service, not alarms related to devices.

Finding 3. **Dependency context converts a device failure into a service decision.** Without it, prioritisation is guesswork dressed as triage.

Finding 4. **Incident and problem management are different disciplines.** Merging them produces slower restoration and shallower analysis at the same time.

Finding 5. **Automation follows process maturity; it does not create it.** An unstable process, once automated, fails faster and more consistently than before.

Finding 6. **Security coordination must be defined before it is needed.** Escalation criteria written during an incident are written too late.

Finding 7. **Prediction rests on data quality, not on tooling.** Visibility precedes correlation, correlation precedes prevention, and prevention precedes prediction.

Finding 8. **A service level governs behaviour only when it is business-based, measurable and consequential.** A target expressed against a component, measured by an unnamed source or unsupported by an underpinning supplier commitment will be reported as compliance while the business service continues to fail.

11 Discussion

The analysis suggests that managed services are best understood as a continuous operational-engineering capability rather than as a support function with a contract. Three shifts carry most of the difference: from device monitoring to service-aware observability; from incident response to incident response combined with problem elimination; and from manual operation to controlled automation with human authority retained where consequence is high.

Each shift also changes what a client should be shown. A reactive service reports what it fixed. A proactive service reports what it prevented, what it automated, and what it learned — and the second report is considerably harder to produce, which is precisely why it distinguishes one provider from another.

12 Practical applications

The framework applies to infrastructure operations, network operations, cloud operations, application support, security operations coordination, service-desk modernisation, end-user services, hybrid infrastructure and government digital services. It also functions as an assessment instrument.

Adoption is sequential rather than simultaneous. The order below reflects dependency: each phase produces the input that the next phase consumes, and attempting a later phase without its predecessor produces activity without measurable effect. The durations are indicative for a single managed service of moderate complexity, not a commitment.

Phase	Focus	Key outputs	Indicative duration
1 Define	Service scope, ownership, targets	Service model, responsibility matrix, agreed target set, underpinning-commitment review	4–6 weeks
2 Instrument	Observability and dependency context	Telemetry coverage per service, dependency model, configuration baseline, alert design	6–10 weeks
3 Operate	Event, incident and change discipline	Priority matrix, escalation tree, runbooks, major-incident procedure, change classes	8–12 weeks
4 Analyse	Problem management and reporting	Problem triggers, known-error records, four-cadence reporting pack, recurrence baseline	one full quarter
5 Automate	Controlled automation of stable work	Automation candidate register, guardrails, level 2–3 automation in	ongoing, reviewed quarterly

Phase	Focus	Key outputs	Indicative duration
		production	
6 Predict	Data-driven forecasting	Retention and classification quality, capacity forecasts, scored predictions	after phases 1–5 are evidenced

Two sequencing errors are common enough to name. Automating before phase 3 encodes an unstable process, and the automation then fails at the same rate as the process but faster. Purchasing predictive analytics before phase 2 produces confident output from incomplete data, which is worse than no output because it is acted upon.

OBSERVED CONDITION	INDICATED GAP
Many alerts but slow diagnosis	Correlation and observability gap
Repeated incidents with identical symptoms	Problem-management gap
Manual repetitive remediation	Automation opportunity
Service levels met while outages recur	Service-improvement gap
No business-service dependency map	Service-context gap
Shared unrestricted administrator accounts	Access-governance risk
No configuration baseline or change record	Change-control gap

13 Applied context

Al Moammar Information Systems operates managed and support services for enterprise and government organisations, across infrastructure, networks, applications, cloud and security environments. That practice is the applied context of this paper: it is where the operational tensions described above are encountered under service-level obligation rather than in principle.

The paper presents no engagement as a research experiment and makes no claim that any particular service was designed according to the framework proposed here. It also names no client. Both choices are deliberate: they keep the paper within the boundary of research, and they keep client information where it belongs.

Placeholder for MIS input — delete this box before publication. One verified operational case may be inserted here: the service type; the client sector, anonymised unless written consent to name the client has been obtained; the service scope; the technologies monitored; the service-level structure; the

observability approach; the incident-management model; one automation use case; a recurring issue that was addressed; and the measurable improvement achieved. Only facts the responsible MIS team can substantiate should be published.

14 Limitations

Five limitations should be stated. The study is a qualitative service-management analysis rather than a controlled operational experiment. It analyses no confidential ticketing, monitoring or client-performance data. Managed-service models vary with contractual scope, criticality, service hours and technology estate, so the framework describes capability rather than prescribing a service design. Predictive operation depends on historical datasets of sufficient quality and volume, and predictive maturity cannot be inferred from the presence of analytics or artificial-intelligence tooling. And the principal service-management reference provides a management-system framework rather than a technology-specific managed-service architecture.

The framework should accordingly be read as a maturity model rather than as an operating manual.

15 Future research

Useful extensions include AIOps and its measurable effect on detection time; predictive incident detection; automated root-cause analysis; inference of service dependencies from observed traffic; intelligent event correlation; artificial-intelligence-assisted service desks; automated remediation with verifiable rollback; the convergence of managed services and security operations; digital-experience monitoring; predictive capacity management; and service-level metrics expressed in terms of user experience rather than component availability.

The natural continuation of this study is a Managed Services Predictive Operations Maturity Model, assessing organisations across visibility, correlation, automation, prevention and prediction, and defining measurable indicators for each stage.

16 Research conclusions

The study reaches eight conclusions, followed by an explicit answer to the research question.

16.1 Managed services should be governed as services, not as collections of technical activity

Defined scope, named ownership and measurable outcomes are the foundation. Without them, technical teams optimise individual systems without visibility of service impact, and the sum of well-run components is not a well-run service.

16.2 Observability is the foundation of proactive operation

Maturity depends on understanding the condition of a service across its components. Monitoring assets is useful; relating those signals to a service is what produces operational context.

16.3 Early detection buys controlled intervention

Detecting degradation before complete failure creates time, and time is what converts an outage into a maintenance action. Monitoring should therefore carry leading indicators as well as failure indicators.

16.4 Incident response must be integrated with cybersecurity risk management

Operational and security incidents require defined coordination mechanisms rather than informal escalation, a conclusion consistent with the current incident-response guidance reviewed in clause 6.3, which places response inside cybersecurity risk management.

16.5 Automation is most effective on stable, understood processes

Automation follows process maturity. Its value is highest where the activity has predictable triggers, defined outcomes, controlled permissions, a clear rollback path and reliable logging; a poorly understood process should be understood before it is automated, not instead.

16.6 Recurring incidents are inputs to engineering improvement

A mature service does not resolve the same failure repeatedly without investigating its cause. The chain runs from incident history through problem analysis and preventive action to service improvement, and a service that never completes it will continue to meet its targets while its clients continue to suffer the same outages.

16.7 Predictive operation requires operational-data maturity

Predictive analytics cannot operate reliably without accurate historical and real-time data. Visibility precedes prediction, and data quality precedes any use of artificial intelligence in operations.

16.8 Continual improvement is the defining characteristic of maturity

Continual improvement is embedded in the service-management lifecycle itself, as clause 6.2 sets out, and a managed service should therefore evolve on the evidence of measured performance, incidents, risks, capacity and client requirements rather than at contract renewal.

16.9 Answer to the research question

The research question asked what engineering, operational and governance capabilities enable ICT managed services to transition from reactive support toward proactive and intelligent service operations. On the analysis conducted here, the answer is as follows.

> **The transition requires defined service governance, service-aware observability, structured and separated incident and problem management, defined cybersecurity coordination, controlled automation with human authority retained where consequence is high, recorded operational knowledge, and continual improvement driven by measurement. Predictive capability is an advanced maturity stage built upon these foundations, never a substitute for any of them.**

17 Conclusion

Managed ICT services are moving from a support function toward a continuous operational-engineering capability. This paper identified seven foundational capabilities — service governance, observability, incident management, problem management, cybersecurity integration, automation and continual improvement — and set them out as a progression from reactive through proactive and preventive to predictive operation.

The central conclusion is that managed-service maturity cannot be measured by ticket closure, engineer availability or service-level compliance. A mature service demonstrates that it can observe service behaviour, detect emerging risk, restore service efficiently, understand recurring failure, automate appropriate activity under control, and improve on the evidence it collects.

The practical question consequently evolves from how quickly an incident can be resolved to how effectively operational disruption can be prevented, detected, contained, resolved and learned from. That shift is what separates conventional ICT support from intelligent managed services.

References

Each source is cited in a footnote at first mention. The footnote numbers below correspond to those citations.

1. International Organization for Standardization, ISO/IEC 20000-1:2018, Information technology — Service management — Part 1: Service management system requirements. Geneva: ISO/IEC, 2018. Reviewed and confirmed as current in 2023.
2. AXELOS, ITIL Foundation: ITIL 4 Edition. London: TSO, 2019.
3. International Organization for Standardization, ISO/IEC 27001:2022, Information security, cybersecurity and privacy protection — Information security management systems — Requirements. Geneva: ISO/IEC, 2022.
4. National Institute of Standards and Technology, The NIST Cybersecurity Framework (CSF) 2.0, NIST CSWP 29. Gaithersburg, MD: NIST, February 2024.
5. ISACA, COBIT 2019 Framework: Governance and Management Objectives. Schaumburg, IL: ISACA, 2018.
6. International Organization for Standardization, ISO 22301:2019, Security and resilience — Business continuity management systems — Requirements. Geneva: ISO, 2019.
7. International Organization for Standardization, ISO/IEC 20000 IT Service Management — A Practical Guide. Geneva: ISO, 2019.
8. A. Nelson, S. Rekhi, M. Souppaya and K. Scarfone, Incident Response Recommendations and Considerations for Cybersecurity Risk Management: A CSF 2.0 Community Profile, NIST Special Publication 800-61 Revision 3. Gaithersburg, MD: NIST, April 2025.
9. S. Rose, O. Borchert, S. Mitchell and S. Connelly, Zero Trust Architecture, NIST Special Publication 800-207. Gaithersburg, MD: NIST, 2020. DOI: 10.6028/NIST.SP.800-207.
10. R. Chandramouli and Z. Butcher, A Zero Trust Architecture Model for Access Control in Cloud-Native Applications in Multi-Cloud Environments, NIST Special Publication 800-207A. Gaithersburg, MD: NIST, 2023.
11. National Cybersecurity Authority, Data Cybersecurity Controls. Riyadh: NCA, Kingdom of Saudi Arabia, 2022.

12. National Cybersecurity Authority, Essential Cybersecurity Controls (ECC). Riyadh: NCA, Kingdom of Saudi Arabia. Edition and year of issue to be confirmed before publication.
13. PeopleCert, ITIL 4 Practice Guide: Service Level Management. London: PeopleCert International (AXELOS), current edition 2025.
14. B. Beyer, C. Jones, J. Petoff and N. R. Murphy (eds.), Site Reliability Engineering: How Google Runs Production Systems. Sebastopol, CA: O'Reilly Media, 2016.
15. National Institute of Standards and Technology, Artificial Intelligence Risk Management Framework (AI RMF 1.0), NIST AI 100-1. Gaithersburg, MD: NIST, January 2023.
16. International Organization for Standardization, ISO/IEC 42001:2023, Information technology — Artificial intelligence — Management system. Geneva: ISO/IEC, 2023.