# 8.5 Service Level Management im Zusammenspiel mit Incident, Problem, Change und Configuration Management

> **Kurz erklärt**
>
> Service Level Management kann Servicequalität nicht allein steuern.
>
> Es benötigt Informationen aus Incident Management, Problem Management, Change Enablement, Service Configuration Management, Knowledge Management, Supplier Management sowie Measurement and Reporting.
>
> Gleichzeitig liefert Service Level Management Ziele, Prioritäten und Anforderungen zurück, an denen diese Practices ihre Arbeit ausrichten können.
>
> Erst durch dieses Zusammenspiel werden Servicequalität, Risiken und Verbesserungsbedarf aus Sicht der Benutzer und des Geschäfts sichtbar.

---

**Warum das Zusammenspiel wichtig ist**

Ein Service Level beschreibt, welche Qualität ein Service erreichen soll.

Ob diese Qualität tatsächlich erreicht wird, hängt jedoch von vielen Bereichen ab.

Beispiele:

- Incident Management stellt gestörte Services wieder her.
- Problem Management untersucht wiederkehrende Ursachen.
- Change Enablement steuert Änderungen am Service.
- Service Configuration Management zeigt technische Abhängigkeiten.
- Knowledge Management stellt Lösungen und Workarounds bereit.
- Supplier Management steuert externe Lieferanten.
- Monitoring and Event Management liefert Zustandsdaten.
- Measurement and Reporting bereitet Kennzahlen auf.
- Continual Improvement verfolgt Verbesserungsmaßnahmen.

Service Level Management verbindet diese Informationen mit den vereinbarten Servicezielen.

---

**Grundidee des Zusammenspiels**

```
Serviceziele und Erwartungen festlegen
        ↓
Service Level Management definiert messbare Ziele
        ↓
Incident Management liefert Störungs- und Wiederherstellungsdaten
        ↓
Problem Management untersucht wiederkehrende Ursachen
        ↓
Change Enablement setzt Verbesserungen kontrolliert um
        ↓
Service Configuration Management zeigt Abhängigkeiten
        ↓
Knowledge Management stellt nutzbares Wissen bereit
        ↓
Supplier Management bewertet externe Beiträge
        ↓
Measurement and Reporting zeigt Ergebnisse und Trends
        ↓
Service Review bewertet Abweichungen und Risiken
        ↓
Continual Improvement verfolgt Verbesserungen
```

Die Practices arbeiten nicht nacheinander in einer festen Reihenfolge.

Sie tauschen fortlaufend Informationen aus.

---

**Service Level Management und Incident Management**

Incident Management liefert wichtige Informationen darüber, wie stabil und zuverlässig ein Service im Alltag funktioniert.

Relevante Informationen:

- Anzahl der Incidents je Service,
- Prioritäten,
- Auswirkung und Dringlichkeit,
- Reaktionszeiten,
- Wiederherstellungszeiten,
- Lösungszeiten,
- Major Incidents,
- Eskalationen,

- Wiedereröffnungen,
- Incidents nach Changes,
- betroffene Benutzer und Standorte,
- verwendete Workarounds,
- Benutzerfeedback.

Diese Daten helfen zu prüfen, ob vereinbarte Serviceziele erreicht werden.

---

**Was Service Level Management an Incident Management liefert**

Service Level Management unterstützt Incident Management durch:

- vereinbarte Servicezeiten,
- Supportzeiten,
- Prioritätsregeln,
- Reaktionsziele,
- Wiederherstellungsziele,
- Kommunikationsanforderungen,
- Eskalationsregeln,
- Kritikalität der Services,
- vereinbarte Statusintervalle,
- und Erwartungen der Kunden.

Dadurch weiß Incident Management, welche Incidents besonders schnell und intensiv bearbeitet werden müssen.

---

**Service Level und Incident-Priorität**

Die Priorität eines Incidents sollte nicht allein von der Lautstärke eines Benutzers oder der technischen Komplexität abhängen.

Zu berücksichtigen sind:

- betroffener Service,
- Servicekritikalität,
- Anzahl betroffener Benutzer,
- betroffener Geschäftsprozess,
- Dringlichkeit,
- vorhandener Workaround,
- Sicherheits- oder Datenschutzbezug,
- Servicezeit,
- vereinbarte Serviceziele.

Beispiel:

Ein technisch kleiner Fehler kann hohe Priorität besitzen, wenn ein geschäftskritischer Prozess vollständig blockiert wird.

---

**Reaktionszeit im Incident Management**

Service Level Management definiert möglicherweise ein Reaktionsziel.

Incident Management muss dafür nachvollziehbar erfassen:

- Zeitpunkt des Eingangs,
- Zeitpunkt der Priorisierung,
- Zeitpunkt der ersten qualifizierten Bearbeitung,
- Zeitpunkt der ersten Benutzerinformation,
- Supportzeit,
- Wartezeiten,
- Eskalationen.

Eine automatische Eingangsbestätigung sollte nicht mit einer qualifizierten Bearbeitung verwechselt werden.

---

**Wiederherstellung und dauerhafte Lösung unterscheiden**

Incident Management konzentriert sich auf die schnelle Wiederherstellung des Service.

Service Level Management kann deshalb ein Ziel für die Wiederherstellungszeit besitzen.

Beispiel:

Ein Workaround macht Benutzer nach zwei Stunden wieder arbeitsfähig.

Die dauerhafte Lösung erfolgt erst eine Woche später durch einen Change.

Dann gilt:

- Service wurde nach zwei Stunden wiederhergestellt.
- Das zugrunde liegende Problem blieb zunächst bestehen.
- Die endgültige Lösung erfolgte später.

Diese Zeitpunkte sollten getrennt gemessen werden.

---

**SLA-Verletzung durch Incident**

Wenn ein Incident zu einer SLA-Verletzung führt, sollte geprüft werden:

- Wurde der Incident richtig priorisiert?

- War der Service korrekt zugeordnet?
- Wurde rechtzeitig eskaliert?
- Waren Zuständigkeiten klar?
- War ein Workaround bekannt?
- War die Kommunikation ausreichend?
- Haben interne OLAs funktioniert?
- Hat ein Lieferant verzögert?
- War das Ziel realistisch?
- Ist ein Problem Record erforderlich?

Eine SLA-Verletzung ist häufig nicht nur ein technisches Ereignis.

Sie kann auch organisatorische Ursachen besitzen.

---

**Beispiel: verspätete Eskalation**

**Situation**

Ein geschäftskritischer Service fällt aus.

**SLA-Ziel**

Wiederherstellung innerhalb von vier Stunden.

**Problem**

Der Service Desk eskaliert erst nach zwei Stunden an das zuständige Fachteam.

**Folge**

Das Wiederherstellungsziel wird verfehlt.

**Analyse**

- CI-Zuordnung war unklar.
- Eskalationskriterien fehlten.
- OLA mit dem Fachteam war nicht eindeutig.
- Kein passender Knowledge-Artikel war vorhanden.

**Verbesserungen**

- Eskalationsregel überarbeiten,
- OLA präzisieren,
- Service- und CI-Zuordnung verbessern,
- Knowledge-Artikel erstellen,
- Service Desk schulen.

---

**Service Level Management und Problem Management**

Problem Management untersucht Ursachen wiederkehrender oder schwerwiegender Incidents.

Service Level Management hilft dabei, Problems zu priorisieren.

Relevante Kriterien:

- wiederholte SLA-Verletzungen,
- hohe Anzahl ähnlicher Incidents,
- lange Wiederherstellungszeiten,
- kritischer Service betroffen,
- hohe geschäftliche Auswirkung,
- häufig genutzter Workaround,
- starke Benutzerunzufriedenheit,
- Lieferantenabhängigkeit,
- Sicherheitsrisiko,
- steigender Trend.

Ein Problem kann hohe Priorität besitzen, auch wenn aktuell kein Incident offen ist.

---

**Was Problem Management an Service Level Management liefert**

Problem Management liefert:

- Ursache oder wahrscheinliche Ursache,
- Known Errors,
- Workarounds,
- betroffene Services,
- betroffene Versionen,
- technische und organisatorische Risiken,
- dauerhafte Lösungsvorschläge,
- geplante Changes,
- erwartete Dauer bis zur Lösung,
- verbleibendes Risiko.

Diese Informationen helfen, Servicequalität und zukünftige Risiken realistisch zu bewerten.

---

**Häufige Workarounds als Service-Level-Risiko**

Ein Workaround kann die Wiederherstellung beschleunigen.

Wenn er jedoch ständig genutzt wird, kann dies auf ein ungelöstes Problem hinweisen.

Zu prüfen ist:

- Wie oft wird der Workaround verwendet?
- Wie viel Arbeitszeit verursacht er?
- Wie stark beeinträchtigt er Benutzer?
- Welche Risiken bestehen?
- Ist er zuverlässig?
- Gibt es eine dauerhafte Lösung?
- Warum wurde sie noch nicht umgesetzt?
- Wird das SLA nur durch den Workaround formal eingehalten?

Ein erreichtes SLA bedeutet nicht automatisch, dass der Service gesund ist.

---

**Beispiel: Dienstneustart als Workaround**

**Situation**

Ein Anwendungsdienst fällt regelmäßig aus.

**Workaround**

Der Service Desk veranlasst einen Neustart.

**Kennzahl**

Die Wiederherstellungszeit bleibt innerhalb des SLA.

**Tatsächliches Problem**

- Service ist regelmäßig instabil.
- Benutzer werden häufig unterbrochen.
- Fachteam investiert wiederholt Zeit.
- Ursache ist weiterhin offen.

**Verbesserung**

Problem Record priorisieren und dauerhafte Lösung über Change Enablement umsetzen.

---

**Problem-Priorisierung nach Servicewirkung**

Problem Management sollte nicht nur technische Schwere betrachten.

Wichtige Fragen:

- Welches SLA wird gefährdet?
- Welche Serviceziele werden wiederholt verfehlt?
- Welche Benutzergruppen sind betroffen?
- Welche Geschäftsprozesse hängen davon ab?

- Gibt es einen sicheren Workaround?
- Wie häufig tritt der Fehler auf?
- Welche Kosten entstehen?
- Welche Risiken entstehen bei Nicht-Handeln?

Service Level Management liefert dafür die geschäftliche und servicebezogene Perspektive.

---

**Service Level Management und Change Enablement**

Changes können Servicequalität verbessern oder verschlechtern.

Beispiele:

- Fehlerbehebung,
- Sicherheitsupdate,
- Performance-Optimierung,
- Kapazitätserweiterung,
- neue Funktion,
- Plattformmigration,
- Firewall-Anpassung,
- Zertifikatserneuerung,
- Monitoring-Erweiterung.

Service Level Management hilft zu bewerten, welche Serviceziele durch einen Change beeinflusst werden.

---

**Was Service Level Management an Change Enablement liefert**

Für die Change-Bewertung sind wichtig:

- Kritikalität des betroffenen Service,
- vereinbarte Servicezeit,
- Wartungsfenster,
- Verfügbarkeitsziele,
- Performance-Ziele,
- betroffene Benutzergruppen,
- Wiederherstellungsanforderungen,
- Lieferantenabhängigkeiten,
- Kommunikationsanforderungen,
- mögliche SLA-Auswirkungen.

Dadurch kann ein Change nicht nur technisch, sondern auch servicebezogen bewertet werden.

---

**Was Change Enablement an Service Level Management liefert**

Change Enablement liefert:

- geplante Änderungen,
- betroffene Services und CIs,
- erwartete Ausfallzeiten,
- Risiken,
- Wartungsfenster,
- Rollback-Möglichkeiten,
- Testergebnisse,
- Failed Changes,
- Incidents nach Changes,
- erwarteten Nutzen,
- und Ergebnisse der Nachprüfung.

Diese Informationen gehören in Service Reviews und Service-Level-Berichte.

---

**Geplante Wartung und Service Levels**

Geplante Wartung muss klar geregelt sein.

Zu prüfen ist:

- Liegt die Wartung im vereinbarten Wartungsfenster?
- Wurde sie rechtzeitig angekündigt?
- Wird sie aus der Verfügbarkeitsberechnung ausgeschlossen?
- Welche Benutzer sind betroffen?
- Wurde das Wartungsfenster überschritten?
- Gab es ungeplante Nebenwirkungen?
- Sind abhängige Services betroffen?
- Wurde der Service Desk informiert?

Nicht jede angekündigte Unterbrechung darf automatisch aus der Messung ausgeschlossen werden.

Die Regeln müssen vorher vereinbart sein.

---

**Change Freeze und kritische Geschäftszeiten**

Service Level Management hilft zu erkennen, wann Changes besonders riskant sind.

Beispiele:

- Monatsabschluss,
- Jahresabschluss,
- Verkaufsaktion,

- Produktionshochlauf,
- Prüfungszeitraum,
- wichtige Veranstaltung,
- gesetzliche Meldefrist,
- hohe saisonale Nutzung.

In solchen Zeiträumen kann ein Change Freeze oder eine besonders strenge Bewertung sinnvoll sein.

Emergency Changes können trotzdem notwendig sein.

Sie benötigen dann eine bewusste Risikoentscheidung.

---

**Changes zur Verbesserung von Service Levels**

Wenn Serviceziele regelmäßig verfehlt werden, können Changes notwendig sein.

Beispiele:

- zusätzliche Kapazität bereitstellen,
- Hochverfügbarkeit einführen,
- Monitoring erweitern,
- fehlerhafte Version aktualisieren,
- Automatisierung einführen,
- Eskalationsworkflow verbessern,
- Knowledge Base aktualisieren,
- redundante Verbindung einrichten,
- Backup- und Restore-Verfahren verbessern.

Service Level Management beschreibt den Verbesserungsbedarf.

Change Enablement steuert die kontrollierte Umsetzung.

---

**Beispiel: Performance-Ziel wird verfehlt**

**Beobachtung**

Das Mitarbeiterportal ist verfügbar, reagiert aber zu langsam.

**Service Review**

Das vereinbarte Performance-Ziel wird regelmäßig verfehlt.

**Analyse**

- Datenbank ist ausgelastet.

- Nutzerzahl ist gestiegen.
- Kapazitätsplanung wurde nicht angepasst.
- Monitoring zeigt nur Verfügbarkeit.

**Changes**

- Datenbankressourcen erweitern,
- Abfragen optimieren,
- Performance-Monitoring ergänzen,
- Lasttest einführen.

**Erfolgskriterium**

Die wichtigsten Benutzeraktionen liegen wieder innerhalb des vereinbarten Zielwerts.

---

**Service Level Management und Service Configuration Management**

Service Configuration Management zeigt, wie ein Service technisch und organisatorisch aufgebaut ist.

Wichtige Informationen:

- Configuration Items,
- Serviceabhängigkeiten,
- Versionen,
- Standorte,
- Owner,
- Supportgruppen,
- Lieferanten,
- Zertifikate,
- Schnittstellen,
- Monitoring,
- Backup,
- Kritikalität.

Diese Informationen sind notwendig, um Service Levels realistisch festzulegen und Risiken zu verstehen.

---

**Warum Serviceabhängigkeiten für SLAs wichtig sind**

Ein Service kann nur so zuverlässig sein wie seine kritischen Abhängigkeiten.

Beispiel:

Das Mitarbeiterportal hängt ab von:

- Webserver,
- Datenbank,
- Identity Provider,
- Netzwerk,
- DNS,
- Zertifikat,
- Cloud-Plattform,
- externem Lieferanten.

Ein hohes Verfügbarkeitsziel ist unrealistisch, wenn eine kritische Abhängigkeit deutlich schwächere Ziele oder keine Redundanz besitzt.

---

**Configuration-Daten für SLA-Planung**

Zu prüfen ist:

- Welche CIs unterstützen den Service?
- Welche CIs sind Single Points of Failure?
- Welche Komponenten besitzen keinen Owner?
- Welche Lieferanten sind beteiligt?
- Welche Versionen sind im Einsatz?
- Welche Zertifikate oder Lizenzen können ablaufen?
- Welche Systeme besitzen keine ausreichende Überwachung?
- Welche Abhängigkeiten sind nicht redundant?
- Welche CIs haben besonders viele Incidents?

Diese Informationen helfen, Serviceziele realistisch zu gestalten.

---

**Service Level Management verbessert Configuration Management**

Service Reviews können Datenqualitätsprobleme sichtbar machen.

Beispiele:

- Incident wurde falschem Service zugeordnet.
- Kritisches CI besitzt keinen Owner.
- Lieferantenbeziehung fehlt.
- Service Map ist unvollständig.
- Zertifikat ist nicht als CI erfasst.
- veraltete Anwendungsversion ist dokumentiert.
- Monitoring ist nicht mit dem Service verknüpft.

Solche Erkenntnisse sollten in die Pflege der Configuration-Daten einfließen.

---

**Beispiel: unbekannte Abhängigkeit**

**Situation**

Ein interner Service verfehlt sein Verfügbarkeitsziel.

**Analyse**

Der Service hängt von einer externen API ab.

Diese Beziehung war in der CMDB nicht dokumentiert.

**Folgen**

- Lieferantenabhängigkeit wurde im SLA nicht berücksichtigt.
- Monitoring prüfte die API nicht.
- Eskalationsweg war unklar.

**Verbesserungen**

- API als relevante Abhängigkeit dokumentieren,
- Lieferant zuordnen,
- Monitoring ergänzen,
- SLA und Underpinning Contract prüfen.

---

**Service Level Management und Knowledge Management**

Knowledge Management beeinflusst Reaktions- und Wiederherstellungszeiten.

Gutes Wissen kann:

- Erstdiagnose beschleunigen,
- bekannte Fehler schneller erkennen,
- Workarounds bereitstellen,
- Eskalationen verbessern,
- Benutzerkommunikation vereinheitlichen,
- Self-Service ermöglichen,
- und wiederkehrende Tickets reduzieren.

Knowledge Management trägt damit direkt zur Servicequalität bei.

---

**Was Service Level Management an Knowledge Management liefert**

Service Level Management zeigt, bei welchen Services und Themen Wissen besonders wichtig ist.

Beispiele:

- Service mit vielen Incidents,
- häufige SLA-Verletzungen,
- lange Wiederherstellungszeiten,
- viele Eskalationen,
- hohe Zahl wiederkehrender Requests,
- häufig genutzte Workarounds,
- schlechte Benutzerbewertungen,
- unklare Statuskommunikation.

Daraus können neue oder verbesserte Knowledge-Artikel entstehen.

---

**Was Knowledge Management an Service Level Management liefert**

Knowledge Management liefert:

- Nutzung von Knowledge-Artikeln,
- Suchanfragen ohne Treffer,
- häufig genutzte Workarounds,
- Artikelbewertungen,
- Knowledge-Lücken,
- Self-Service-Erfolgsdaten,
- Runbooks,
- Known Errors,
- Benutzerfeedback.

Diese Informationen helfen zu verstehen, warum bestimmte Serviceziele erreicht oder verfehlt werden.

---

**Beispiel: lange Wiederherstellungszeit**

**Beobachtung**

DNS-Incidents werden regelmäßig zu spät gelöst.

**Analyse**

- Service Desk besitzt keine einheitlichen Prüfschritte.
- Eskalationen enthalten unvollständige Informationen.
- Fachteam wiederholt immer dieselbe Diagnose.

**Verbesserungen**

- Service-Desk-Artikel erstellen,
- Diagnose-Checkliste ergänzen,
- Eskalationsanforderungen definieren,

- Knowledge-Nutzung im Ticket messen.

**Erwartete Wirkung**

Kürzere Erstdiagnose und schnellere Wiederherstellung.

---

**Service Level Management und Supplier Management**

Viele Services hängen von externen Lieferanten ab.

Beispiele:

- Cloud-Anbieter,
- Internetprovider,
- Softwarehersteller,
- Hardwarewartung,
- Rechenzentrum,
- SaaS-Anbieter,
- Telekommunikationsanbieter.

Supplier Management sorgt dafür, dass Lieferantenleistungen die vereinbarten Serviceziele unterstützen.

---

**Was Service Level Management an Supplier Management liefert**

Supplier Management benötigt:

- relevante SLA-Ziele,
- kritische Services,
- Supportzeiten,
- Wiederherstellungsziele,
- Verfügbarkeitsziele,
- Sicherheitsanforderungen,
- Eskalationsanforderungen,
- Reportingbedarf,
- Wartungsfenster,
- und Risiken.

Daraus können geeignete Lieferantenverträge und Leistungsziele abgeleitet werden.

---

**Was Supplier Management an Service Level Management liefert**

Supplier Management liefert:

- Vertragsziele,

- tatsächliche Lieferantenleistung,
- Supportzeiten,
- Eskalationswege,
- Vertragsverletzungen,
- geplante Wartungen,
- Herstellerprobleme,
- Lieferantenrisiken,
- End-of-Support-Termine,
- und Verbesserungsmaßnahmen.

Diese Informationen gehören in Service Reviews.

---

**Lieferantenvertrag und internes SLA abstimmen**

Ein internes SLA darf nicht stärker sein als die tatsächlich verfügbare Unterstützung, sofern keine eigenen Schutzmaßnahmen bestehen.

Beispiel:

**Internes Ziel**

Wiederherstellung innerhalb von vier Stunden.

**Lieferantenvertrag**

Hersteller reagiert innerhalb eines Arbeitstages.

Mögliche Lösungen:

- Vertrag verbessern,
- internes Fachwissen erweitern,
- Redundanz aufbauen,
- Workaround entwickeln,
- internes SLA anpassen,
- Risiko dokumentieren und akzeptieren.

---

**Service Level Management und Monitoring and Event Management**

Monitoring liefert technische und servicebezogene Messdaten.

Beispiele:

- Verfügbarkeit,
- Antwortzeiten,
- Transaktionsfehler,
- Kapazität,

- Schnittstellenzustand,
- Zertifikatsablauf,
- Backupstatus,
- Datenbankzustand,
- Anmeldefunktion.

Service Level Management definiert, welche Serviceeigenschaften gemessen werden müssen.

---

**Monitoring aus Serviceperspektive**

Unzureichend:

> Webserver antwortet auf Ping.

Besser:

- Startseite ist erreichbar.
- Anmeldung funktioniert.
- zentrale Benutzeraktion ist erfolgreich.
- abhängige Schnittstelle antwortet.
- Antwortzeit liegt im Zielbereich.

Ein Service kann technisch „grün“ erscheinen, obwohl Benutzer nicht arbeiten können.

---

**Monitoring und Messmethode abstimmen**

Zu klären ist:

- Welche Messpunkte entsprechen dem SLA?
- Wird während der richtigen Servicezeit gemessen?
- Werden geplante Wartungen gekennzeichnet?
- Werden Teilausfälle erkannt?
- Werden alle relevanten Standorte berücksichtigt?
- Werden technische und synthetische Tests kombiniert?
- Werden Alarme einem Service und Owner zugeordnet?
- Sind Messdaten für Reporting verfügbar?

Monitoring muss zu den vereinbarten Zielen passen.

---

**Service Level Management und Measurement and Reporting**

Measurement and Reporting bereitet Daten so auf, dass Servicequalität bewertet werden kann.

Service Level Management liefert dafür:

- Zielwerte,
- Servicezeiten,
- Messdefinitionen,
- relevante Kennzahlen,
- Ausnahmen,
- Zielgruppen,
- Reportingfrequenz,
- und Reviewanforderungen.

Measurement and Reporting liefert:

- Ist-Werte,
- Trends,
- Abweichungen,
- Datenqualitätsinformationen,
- Dashboards,
- Reports,
- und Vergleichswerte.

---

**Reporting braucht Kontext aus anderen Practices**

Eine Zahl allein erklärt nicht, warum ein Ziel verfehlt wurde.

Beispiel:

Verfügbarkeit beträgt 98,9 Prozent statt 99,5 Prozent.

Für eine sinnvolle Bewertung werden zusätzliche Informationen benötigt:

- Incident Management: Welche Ausfälle traten auf?
- Problem Management: Welche Ursachen sind bekannt?
- Change Enablement: Gab es einen Change davor?
- Configuration Management: Welche Abhängigkeit war betroffen?
- Supplier Management: War ein Lieferant beteiligt?
- Knowledge Management: Gab es einen Workaround?
- Continual Improvement: Welche Maßnahmen laufen?

Reporting verbindet diese Informationen.

---

**Service Level Management und Continual Improvement**

Service Level Management erkennt Abweichungen und Verbesserungspotenziale.

Continual Improvement sorgt dafür, dass daraus konkrete Maßnahmen entstehen.

Mögliche Verbesserungen:

- SLA-Ziel anpassen,
- Monitoring erweitern,
- Problem Record priorisieren,
- Workaround dokumentieren,
- Change durchführen,
- OLA verbessern,
- Lieferantenvertrag anpassen,
- Configuration-Daten pflegen,
- Self-Service verbessern,
- Kapazität erweitern,
- Benutzerkommunikation verbessern.

---

**Verbesserungen aus Service Reviews**

Ein möglicher Ablauf:

```
Serviceziel wird verfehlt
        ↓
Abweichung wird im Report sichtbar
        ↓
Service Review bewertet Ursache und Auswirkung
        ↓
Improvement wird erfasst
        ↓
Owner, Termin und Erfolgskriterium werden festgelegt
        ↓
Problem, Change, Knowledge oder Supplier Management wird eingebunden
        ↓
Maßnahme wird umgesetzt
        ↓
Wirkung wird gemessen
        ↓
Ergebnis wird im nächsten Service Review geprüft
```

So wird aus einer Kennzahl eine konkrete Verbesserung.

---

**Service-Level-Ziele überprüfen**

Nicht jede wiederholte Zielverfehlung bedeutet, dass das Ziel gesenkt werden sollte.

Zu prüfen ist:

- Ist das Ziel weiterhin fachlich notwendig?
- Ist die Messmethode korrekt?
- Sind Daten zuverlässig?
- Sind Ressourcen ausreichend?
- Haben sich Anforderungen verändert?
- Ist die technische Architektur geeignet?
- Unterstützen Lieferanten das Ziel?
- Gibt es unverhältnismäßige Kosten?
- Welche Risiken entstehen bei Anpassung?

Ziele sollten realistisch, aber nicht bequem sein.

---

### Neue Service-Level-Ziele aus Verbesserungen

Neue Erkenntnisse können zusätzliche Ziele erforderlich machen.

Beispiele:

- Performance-Ziel ergänzen,
- Statuskommunikation bei Major Incidents definieren,
- Self-Service-Erfolgsziel aufnehmen,
- Restore-Test-Ziel festlegen,
- Lieferantenreaktion messen,
- Knowledge-Aktualisierung nach Changes prüfen,
- Benutzererfahrung berücksichtigen.

Service Levels sollten sich mit dem Service weiterentwickeln.

---

### Rollen und Verantwortlichkeiten im Zusammenspiel

| Rolle oder Practice | Beitrag |
|---|---|
| **Service Owner** | verantwortet Servicequalität und Prioritäten |
| **Service Level Management** | definiert, misst und überprüft Serviceziele |
| **Service Desk** | liefert Benutzerfeedback und Incident-Daten |
| **Incident Management** | stellt Services wieder her |
| **Problem Management** | untersucht Ursachen und Wiederholungen |
| **Change Enablement** | setzt Änderungen kontrolliert um |
| **Configuration Management** | liefert Services, CIs und Abhängigkeiten |

| Rolle oder Practice | Beitrag |
|---|---|
| **Knowledge Management** | stellt Lösungen und Workarounds bereit |
| **Supplier Management** | steuert externe Leistungen |
| **Measurement and Reporting** | bereitet Daten und Trends auf |
| **Continual Improvement** | verfolgt Verbesserungen |

Die Verantwortlichkeiten müssen klar sein.

Servicequalität ist jedoch eine gemeinsame Aufgabe.

---

### Informationsfluss zwischen den Practices

```
Service Level Management
    ↓ definiert Ziele
Incident Management
    ↓ liefert Störungsdaten
Problem Management
    ↓ liefert Ursachen und Known Errors
Change Enablement
    ↓ liefert Änderungen und Ergebnisse
Configuration Management
    ↓ liefert Abhängigkeiten und Kritikalität
Knowledge Management
    ↓ liefert Workarounds und Nutzungswissen
Supplier Management
    ↓ liefert externe Leistungsdaten
Measurement and Reporting
    ↓ erstellt Berichte und Trends
Service Review
    ↓ beschließt Verbesserungen
Continual Improvement
    ↓ verfolgt Umsetzung und Wirkung
```

Dieser Informationsfluss muss nicht in einem einzigen Werkzeug stattfinden.

Wichtig sind verlässliche Verknüpfungen und Verantwortlichkeiten.

---

### Praxisbeispiel: VPN-Service

**Service Level Management**

Definiert Verfügbarkeit, Supportzeit und Wiederherstellungsziel.

**Incident Management**

Liefert Daten zu VPN-Abbrüchen und Wiederherstellungszeiten.

**Problem Management**

Erkennt fehlerhafte Client-Version als gemeinsame Ursache.

**Knowledge Management**

Stellt Workaround und Prüfschritte bereit.

**Change Enablement**

Steuert Pilot und Rollout einer neuen Client-Version.

**Service Configuration Management**

Zeigt betroffene Clientversionen, Gateway und MFA-Abhängigkeit.

**Supplier Management**

Prüft Herstellerunterstützung.

**Continual Improvement**

Misst, ob die VPN-Incidents nach dem Rollout sinken.

---

**Praxisbeispiel: Zertifikatsausfall**

**Incident**

Anmeldung am Mitarbeiterportal fällt aus.

**Problem**

Zertifikat war abgelaufen.

**Configuration Management**

Zertifikat war nicht als CI mit Owner und Ablaufdatum erfasst.

**Service Level Management**

Verfügbarkeitsziel wurde verfehlt.

**Change Enablement**

Zertifikat wird erneuert und Monitoring eingeführt.

**Knowledge Management**

Runbook und Prüfschritte werden erstellt.

**Continual Improvement**

Alle produktiven Zertifikate werden erfasst und mindestens 30 Tage vor Ablauf überwacht.

---

**Praxisbeispiel: Lieferantenabhängige Störung**

**Situation**

Ein SaaS-Service ist nicht erreichbar.

**Incident Management**

Koordiniert Benutzerkommunikation und Eskalation.

**Supplier Management**

Eskaliert an den Anbieter.

**Service Level Management**

Prüft Auswirkung auf SLA und Benutzer.

**Configuration Management**

Zeigt betroffene Geschäftsprozesse und Schnittstellen.

**Problem Management**

Prüft wiederkehrende Providerstörungen.

**Continual Improvement**

Bewertet zusätzlichen Workaround, Redundanz oder Vertragsänderung.

---

**Praxisbeispiel: Service verfügbar, aber langsam**

**Monitoring**

Server und Anwendung erscheinen verfügbar.

**Benutzerfeedback**

Benutzer melden lange Ladezeiten.

**Service Level Management**

Erkennt, dass nur Verfügbarkeit, aber keine Benutzerperformance gemessen wird.

**Configuration Management**

Zeigt Datenbank und externe Schnittstelle als Abhängigkeiten.

**Problem Management**

Untersucht Performanceursache.

**Change Enablement**

Steuert Kapazitäts- und Konfigurationsänderung.

**Measurement and Reporting**

Ergänzt Antwortzeit und synthetische Benutzertransaktion.

---

**Praxisbeispiel: viele Standardtickets**

**Beobachtung**

Viele Tickets betreffen Passwort und MFA.

**Service Level Management**

Erkennt hohe Bearbeitungsmenge und schlechte Benutzererfahrung.

**Knowledge Management**

Überarbeitet Benutzerartikel und Suchbegriffe.

**Service Request Management**

Vereinfacht Formulare und Automatisierung.

**Service Desk**

Liefert häufige Fragen und Rückmeldungen.

**Continual Improvement**

Misst Ticketentwicklung, Self-Service-Nutzung und Artikelbewertung.

---

**Typische Fehler im Zusammenspiel**

**Fehler 1**

Service Levels werden festgelegt, ohne technische Abhängigkeiten zu prüfen.

---

**Fehler 2**

Incident-Daten werden gemessen, aber nicht servicebezogen ausgewertet.

---

**Fehler 3**

Wiederholte SLA-Verletzungen führen nicht zu Problem Management.

---

**Fehler 4**

Changes werden durchgeführt, ohne Service-Level-Auswirkungen zu prüfen.

---

**Fehler 5**

Geplante Wartung wird pauschal aus allen Messungen ausgeschlossen.

---

**Fehler 6**

CMDB-Daten sind veraltet und führen zu falscher Risikobewertung.

---

**Fehler 7**

Lieferantenverträge unterstützen interne Serviceziele nicht.

---

**Fehler 8**

Monitoring misst einzelne Server, aber keine nutzbare Servicefunktion.

---

**Fehler 9**

Workarounds halten SLA-Ziele formal ein, obwohl der Service dauerhaft instabil ist.

**Fehler 10**

Knowledge-Lücken verlängern Wiederherstellungszeiten.

---

**Fehler 11**

Reports zeigen Abweichungen, aber keine Verantwortlichen oder Maßnahmen.

---

**Fehler 12**

Verbesserungen werden umgesetzt, aber ihre Wirkung wird nicht gemessen.

---

**Checkliste Zusammenspiel mit Incident Management**

- [ ] Incident ist dem richtigen Service zugeordnet
- [ ] Priorität berücksichtigt Auswirkung und Dringlichkeit
- [ ] Servicekritikalität ist bekannt
- [ ] Reaktionszeit wird korrekt gemessen
- [ ] Wiederherstellungszeit wird korrekt gemessen
- [ ] Workaround wird dokumentiert
- [ ] SLA-Verletzung wird analysiert
- [ ] Eskalationsweg ist bekannt
- [ ] OLA-Beiträge sind abgestimmt
- [ ] Major Incidents werden im Service Review betrachtet
- [ ] Benutzerfeedback wird berücksichtigt

---

**Checkliste Zusammenspiel mit Problem Management**

- [ ] wiederkehrende SLA-Verletzungen werden untersucht
- [ ] Problems sind mit Services verknüpft
- [ ] häufige Workarounds werden ausgewertet
- [ ] Known Errors sind dokumentiert
- [ ] geschäftliche Auswirkung ist bekannt
- [ ] Problem-Priorität berücksichtigt Servicekritikalität
- [ ] dauerhafte Lösung ist bewertet
- [ ] geplanter Change ist verknüpft

- [ ] Rest Risiko ist dokumentiert
- [ ] Service Review kennt den Problemstatus

---

**Checkliste Zusammenspiel mit Change Enablement**

- [ ] betroffener Service ist im Change angegeben
- [ ] Servicezeit und Wartungsfenster sind geprüft
- [ ] mögliche SLA-Auswirkung ist bewertet
- [ ] betroffene Benutzergruppen sind bekannt
- [ ] abhängige Services sind geprüft
- [ ] Service Desk wurde informiert
- [ ] Kommunikation wurde vorbereitet
- [ ] Erfolgskriterien beziehen sich auf Servicequalität
- [ ] Incidents nach dem Change werden geprüft
- [ ] Failed Changes werden im Service Review betrachtet
- [ ] geplante Wartung wird korrekt berichtet

---

**Checkliste Zusammenspiel mit Service Configuration Management**

- [ ] Service besitzt dokumentierte CIs
- [ ] kritische Abhängigkeiten sind sichtbar
- [ ] Single Points of Failure sind bekannt
- [ ] Owner und Supportgruppen sind aktuell
- [ ] Lieferantenbeziehungen sind dokumentiert
- [ ] Zertifikate und Schnittstellen sind berücksichtigt
- [ ] Kritikalität ist nachvollziehbar
- [ ] Configuration-Daten werden bei SLA-Planung genutzt
- [ ] Datenfehler aus Reviews werden korrigiert
- [ ] Änderungen aktualisieren die CMDB

---

**Checkliste Zusammenspiel mit Knowledge Management**

- [ ] häufige Incidents besitzen passende Artikel
- [ ] Known Errors sind auffindbar
- [ ] Workarounds sind dokumentiert

- [ ] Eskalationskriterien sind beschrieben
- [ ] Service Desk nutzt Knowledge aktiv
- [ ] Knowledge-Lücken werden ausgewertet
- [ ] Artikel nach Changes aktualisiert
- [ ] Benutzerwissen ist verständlich
- [ ] häufige Suchanfragen werden analysiert
- [ ] Knowledge-Nutzung wird im Service Review betrachtet

---

**Checkliste Zusammenspiel mit Supplier Management**

- [ ] kritische Lieferanten sind bekannt
- [ ] Lieferantenverträge unterstützen SLA-Ziele
- [ ] Supportzeiten sind kompatibel
- [ ] Eskalationswege sind dokumentiert
- [ ] Wartungsfenster sind abgestimmt
- [ ] Lieferantenleistung wird gemessen
- [ ] Vertragsverletzungen werden analysiert
- [ ] Lieferantenrisiken sind dokumentiert
- [ ] Alternativen oder Workarounds sind geprüft
- [ ] Supplier Reviews fließen in Service Reviews ein

---

**Checkliste Service Review über alle Practices**

- [ ] SLA-Ziele und Ist-Werte geprüft
- [ ] Incident-Trends betrachtet
- [ ] Major Incidents bewertet
- [ ] offene Problems und Known Errors geprüft
- [ ] Changes und Releases bewertet
- [ ] Configuration-Risiken betrachtet
- [ ] Knowledge-Lücken berücksichtigt
- [ ] Lieferantenleistung bewertet
- [ ] Benutzerfeedback einbezogen
- [ ] kommende Risiken und Anforderungen betrachtet
- [ ] Verbesserungen priorisiert
- [ ] Owner, Termin und Erfolgskriterium festgelegt

**Bedeutung für Fachinformatiker für Systemintegration**

Fachinformatiker liefern wichtige technische Informationen für das Zusammenspiel der Practices.

Im Arbeitsalltag bedeutet das:

- Incidents dem richtigen Service und CI zuordnen,
- Wiederherstellungszeiten sauber dokumentieren,
- wiederkehrende Fehler an Problem Management melden,
- Changes auf Serviceauswirkungen prüfen,
- technische Abhängigkeiten nachvollziehbar pflegen,
- Monitoring aus Serviceperspektive ausrichten,
- Knowledge-Artikel und Runbooks aktualisieren,
- Lieferantenabhängigkeiten sichtbar machen,
- und Verbesserungen technisch umsetzen und überprüfen.

Technische Arbeit wird dadurch nicht nur als einzelne Tätigkeit betrachtet.

Sie wird mit Servicequalität, Benutzerwirkung und geschäftlichem Nutzen verbunden.

---

**Zusammenfassung**

> Serviceziele und Erwartungen festlegen
> ↓
> Incident Management liefert Störungs- und Wiederherstellungsdaten
> ↓
> Problem Management untersucht wiederkehrende Ursachen
> ↓
> Change Enablement setzt Verbesserungen kontrolliert um
> ↓
> Service Configuration Management zeigt Abhängigkeiten und Risiken
> ↓
> Knowledge Management beschleunigt Diagnose und Wiederherstellung
> ↓
> Supplier Management steuert externe Beiträge
> ↓
> Monitoring und Reporting messen Servicequalität
> ↓
> Service Review bewertet Abweichungen und Risiken
> ↓
> Continual Improvement verfolgt Maßnahmen und Wirkung

---

**Merksätze**

> Service Level Management kann Servicequalität nicht allein erzeugen.

> Incident Management zeigt, wie sich Störungen auf Serviceziele auswirken.

> Problem Management verhindert, dass dieselben SLA-Verletzungen ständig wiederkehren.

> Change Enablement setzt Serviceverbesserungen kontrolliert um.

> Configuration Management zeigt, wovon ein Service tatsächlich abhängig ist.

> Knowledge Management verkürzt Diagnose- und Wiederherstellungszeiten.

> Interne Serviceziele müssen durch Teams und Lieferanten unterstützt werden.

> Monitoring muss nutzbare Servicefunktionen und nicht nur einzelne Systeme messen.

> Ein erreichtes SLA bedeutet nicht automatisch, dass ein Service dauerhaft gesund ist.

> Verbesserungen sind erst abgeschlossen, wenn ihre Wirkung gemessen wurde.

**Verwandte Seiten**

- 8.1 Service Level Management – Ziele, Begriffe und Grundlagen
- 8.2 SLA, SLR, OLA und Underpinning Contracts
- 8.3 Kennzahlen, Messmethoden und Reporting
- 8.4 Service Reviews und Continual Improvement
- Incident Management
- Problem Management
- Change Enablement
- Service Configuration Management
- Knowledge Management
- Supplier Management
- Monitoring and Event Management
- Measurement and Reporting
- Continual Improvement

---

**Quellen und Versionsstand**

**Offizielle Grundlagen**

- PeopleCert – ITIL Practice Guide: Service Level Management
- PeopleCert – ITIL Practice Guide: Incident Management
- PeopleCert – ITIL Practice Guide: Problem Management
- PeopleCert – ITIL Practice Guide: Change Enablement
- PeopleCert – ITIL Practice Guide: Service Configuration Management
- PeopleCert – ITIL Practice Guide: Knowledge Management
- PeopleCert – ITIL Practice Guide: Supplier Management
- PeopleCert – ITIL Practice Guide: Monitoring and Event Management
- PeopleCert – ITIL Practice Guide: Measurement and Reporting
- PeopleCert – ITIL Practice Guide: Continual Improvement
- ITIL Foundation – Version 5
- ITIL Service – Version 5

**Einordnung**

Die dargestellten:

- Informationsflüsse,
- Rollenbeiträge,
- Schnittstellen,
- Checklisten,
- Ablaufdarstellungen,
- und Praxisbeispiele

sind herstellerneutrale Praxisempfehlungen.

ITIL schreibt keine universelle:

- Schnittstellenmatrix,
- SLA-Prozessintegration,
- Incident- oder Problem-Priorisierung,
- Change-Bewertungsvorlage,
- Service-Review-Struktur,
- Reporting-Form,
- oder Toolintegration

für alle Organisationen vor.

Die konkrete Umsetzung muss an:

- Services,
- Kunden,
- Benutzergruppen,
- Risiken,
- Service Levels,
- Supportmodell,
- Organisationsstruktur,
- Lieferanten,
- Toollandschaft,
- Datenqualität,
- und verfügbare Fähigkeiten

angepasst werden.

**Behandelter Framework-Stand:** ITIL Version 5
**Zusätzlich berücksichtigt:** aktuelle ITIL-4-Practice-Guidance
**Fachlicher Stand:** August 2026

---