10.3 Alarme, Eskalationen und Automatisierung

> **Kurz erklärt**
>
> Nicht jedes Event erfordert menschliches Eingreifen.
>
> Erst wenn ein Ereignis eine definierte Bedeutung besitzt, wird daraus ein Alarm.
>
> Je nach Schwere können anschließend automatische Aktionen, Eskalationen oder die Erstellung eines Incident-Tickets erfolgen.
>
> Ziel ist es, Probleme möglichst schnell und möglichst automatisch zu erkennen und zu behandeln.

Vom Event zum Alarm

Ein Monitoring-System erzeugt kontinuierlich Events.

Nur ein Teil dieser Events wird zu Alarmen.

Ein typischer Ablauf:

Die Bewertung erfolgt anhand definierter Regeln.

Was ist ein Alarm?

Ein Alarm ist eine Benachrichtigung über ein relevantes Ereignis.

Er informiert darüber, dass eine Reaktion erforderlich oder zumindest sinnvoll sein kann.

Ein Alarm bedeutet jedoch nicht automatisch, dass bereits ein Incident vorliegt.

Beispiele für Alarme

Typische Alarme:

- Festplatte fast voll
- Backup fehlgeschlagen
- RAID degradiert
- Server nicht erreichbar
- hohe CPU-Auslastung
- Zertifikat läuft bald ab
- Temperatur überschritten
- Netzwerkverbindung unterbrochen
- Container abgestürzt
- Datenbank antwortet nicht

Je nach Kritikalität unterscheiden sich die weiteren Maßnahmen.

Alarme priorisieren

Nicht jeder Alarm besitzt dieselbe Bedeutung.

Mögliche Kriterien:

- betroffener Service,
- Business Impact,
- Anzahl betroffener Benutzer,
- Kritikalität,
- Tageszeit,
- Sicherheitsrelevanz,
- vorhandene Redundanz,
- Wiederholungen.

Dadurch kann dieselbe technische Meldung unterschiedlich behandelt werden.

Alarmstufen

Viele Organisationen verwenden mehrere Alarmstufen.

Beispiel:

Stufe	Bedeutung
Information	nur protokollieren
Warnung	beobachten
Hoch	Administrator informieren
Kritisch	sofort reagieren
Notfall	Eskalation und Incident

Die genaue Anzahl der Stufen ist organisationsabhängig.

Alarmkanäle

Alarme können über verschiedene Wege zugestellt werden.

Beispiele:

- Dashboard
- E-Mail
- SMS
- Push-Nachricht
- Microsoft Teams
- Slack
- Telefonanruf
- Bereitschaftssystem
- Pager
- ITSM-System

Der Kanal richtet sich nach der Kritikalität.

Alarmrouting

Nicht jeder Alarm geht an dieselbe Person.

Beispiele:

Alarm	Empfänger
Netzwerk	Netzwerkteam
Backup	Backup-Team
Datenbank	Datenbankadministration
Cloud	Cloud-Team

Alarm	Empfänger
Active Directory	Windows-Team
Linux	Linux-Team
Container	Plattform-Team

Dadurch gelangen Informationen direkt an die zuständigen Personen.

Automatische Reaktionen

Viele Alarme können automatisiert verarbeitet werden.

Beispiele:

- Dienst neu starten
- Container neu starten
- VM verschieben
- Backup erneut starten
- Speicher bereinigen
- DNS umschalten
- Last verteilen
- Ticket erstellen
- Benutzer informieren

Nicht jede Situation erfordert sofort einen Administrator.

Selbstheilung (Self-Healing)

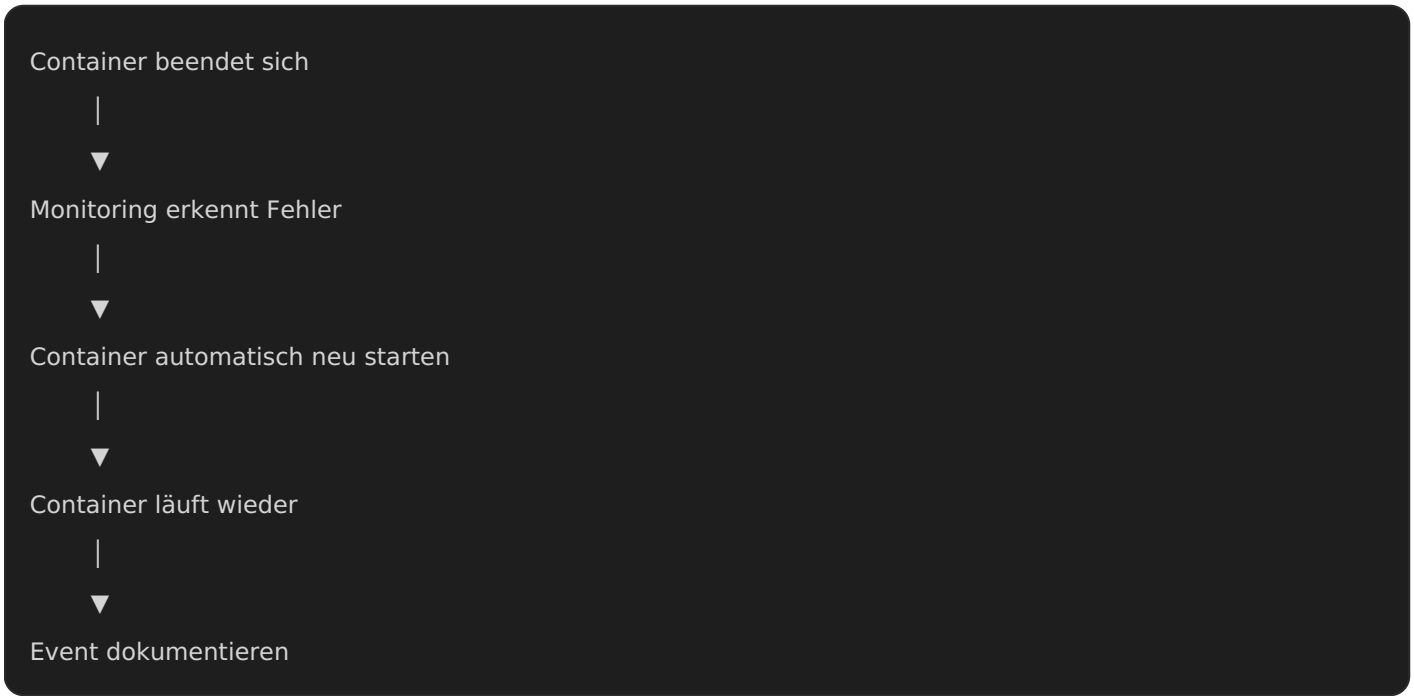
Self-Healing beschreibt automatische Korrekturmaßnahmen.

Beispiele:

- Dienst startet automatisch neu.
- Container wird neu erstellt.
- zweite Netzwerkverbindung wird aktiviert.
- Cluster übernimmt den Betrieb.
- Load Balancer entfernt fehlerhaften Server.
- Speicherplatz wird automatisch erweitert.

Dadurch entstehen viele Incidents gar nicht erst.

Beispiel



Der Benutzer bemerkt den Fehler möglicherweise überhaupt nicht.

Automatisierte Incident-Erstellung

Nicht jeder Alarm erzeugt automatisch einen Incident.

Sinnvoll ist dies beispielsweise bei:

- Serverausfall
- RAID-Fehler
- Datenbank nicht erreichbar
- Backup fehlgeschlagen
- Zertifikat abgelaufen
- Cloud-Service ausgefallen

Das ITSM-System erstellt automatisch ein Incident-Ticket.

Eskalationen

Kann ein Alarm nicht automatisch gelöst werden,

beginnt häufig eine Eskalation.

Beispiele:

- Bereitschaft informieren,
- Spezialisten hinzuziehen,
- Incident Manager benachrichtigen,

- Major Incident aktivieren.

Welche Eskalation erfolgt,

hängt von Schwere und Auswirkungen ab.

Zeitabhängige Eskalationen

Nicht jeder Alarm wird sofort eskaliert.

Beispiel:

CPU-Auslastung:

- über 90 % für 30 Sekunden → ignorieren
- über 90 % für 10 Minuten → Warnung
- über 95 % für 30 Minuten → Incident

Zeitfilter verhindern unnötige Reaktionen.

Mehrstufige Eskalationen

Ein möglicher Ablauf:

Nicht jeder Alarm erreicht automatisch die höchste Eskalationsstufe.

Alarmunterdrückung

Während geplanter Wartungen können Alarme unterdrückt werden.

Beispiel:

Ein Server wird bewusst neu gestartet.

Das Monitoring erkennt:

Server nicht erreichbar.

Da ein genehmigter Change aktiv ist,

wird kein Incident erzeugt.

Abhängigkeiten berücksichtigen

Ein ausgefallener Router verursacht häufig viele weitere Alarme.

Beispiele:

- Webserver offline
- NAS offline
- Drucker offline
- VoIP offline
- WLAN offline

Durch Abhängigkeitsregeln wird möglichst nur der eigentliche Fehler hervorgehoben.

Alarmkorrelation

Mehrere Alarme können zusammengehören.

Beispiel:

- Datenbank langsam
- Anwendung langsam
- Webserver meldet Fehler

Die eigentliche Ursache:

Storage reagiert verzögert.

Alarmkorrelation unterstützt Administratoren bei der Ursachenanalyse.

Alarmfluten vermeiden

Zu viele Warnungen führen häufig dazu,

dass wichtige Alarme übersehen werden.

Typische Maßnahmen:

- Filter,
- Korrelation,
- Deduplizierung,
- sinnvolle Grenzwerte,
- Wartungsfenster,
- Zeitfilter.

Dadurch steigt die Qualität der Alarmierung.

Dashboards

Dashboards stellen Alarme übersichtlich dar.

Typische Informationen:

- aktuelle Alarme,
- kritische Services,
- Verfügbarkeit,
- offene Incidents,
- Trends,
- Systemstatus,
- Karten,
- Serviceübersichten.

Sie ermöglichen einen schnellen Überblick.

Benachrichtigung außerhalb der Arbeitszeit

Kritische Systeme werden häufig rund um die Uhr überwacht.

Außerhalb der Geschäftszeiten können Alarme beispielsweise an:

- Rufbereitschaft,
- Bereitschaftsdienst,
- externen Dienstleister

gesendet werden.

Dadurch bleibt die Reaktionsfähigkeit erhalten.

Praxisbeispiel

Ein RAID-Verbund meldet:

Eine Festplatte ausgefallen.

Automatisch geschieht:

- Alarm erzeugen,
- Incident erstellen,
- Storage-Team informieren,
- Dashboard aktualisieren,
- Bereitschaft benachrichtigen,
- Dokumentation ergänzen.

Die eigentliche Reparatur erfolgt anschließend durch den zuständigen Administrator.

Typische Fehler

Fehler 1

Jeder Alarm erzeugt sofort einen Incident.

Fehler 2

Keine automatische Fehlerbehandlung.

Fehler 3

Keine Alarmkorrelation.

Fehler 4

Zu viele Benachrichtigungen.

Fehler 5

Keine Wartungsfenster berücksichtigt.

Fehler 6

Falsche Empfänger erhalten Alarme.

Fehler 7

Self-Healing wird nicht genutzt.

Fehler 8

Bereitschaft wird wegen unwichtiger Warnungen gestört.

Fehler 9

Alarmregeln werden nach Änderungen nicht angepasst.

Fehler 10

Fehlgeschlagene Automatisierungen bleiben unbemerkt.

Checkliste Alarmmanagement

- [ ] Event korrekt bewertet
- [ ] Alarmstufe festgelegt
- [ ] richtiger Empfänger definiert
- [ ] Eskalation vorhanden
- [ ] Wartungsfenster berücksichtigt
- [ ] Alarm dokumentiert
- [ ] Korrelation eingerichtet
- [ ] Deduplizierung aktiv
- [ ] Dashboard aktualisiert
- [ ] regelmäßige Überprüfung geplant

Checkliste Automatisierung

- [ ] Self-Healing möglich
- [ ] Workflow getestet
- [ ] Rollback vorhanden
- [ ] Fehler protokolliert

- [ ] Monitoring aktiv
- [ ] Incident-Erstellung geprüft
- [ ] Benachrichtigungen getestet
- [ ] Verantwortlichkeiten dokumentiert

Bedeutung für Fachinformatiker für Systemintegration

Fachinformatiker richten häufig Alarmierungen und automatische Reaktionen ein.

Typische Aufgaben:

- Alarmregeln definieren,
- Eskalationen konfigurieren,
- Dashboards pflegen,
- Automatisierungen entwickeln,
- Self-Healing testen,
- Benachrichtigungen überprüfen,
- Alarmfluten reduzieren,
- Monitoring kontinuierlich verbessern.

Ein gut abgestimmtes Alarmmanagement sorgt dafür, dass kritische Ereignisse schnell erkannt und möglichst automatisch behandelt werden.

Zusammenfassung

> Monitoring erkennt Event
>
> ↓
>
> Event bewerten
>
> ↓
>
> Alarm erzeugen
>
> ↓
>
> Priorisieren
>
> ↓
>
> Automatische Reaktion oder Eskalation

> ↓
>
> Incident (falls erforderlich)
>
> ↓
>
> Dokumentation

Merksätze

> Nicht jeder Alarm führt zu einem Incident.

> Gute Alarmregeln vermeiden unnötige Benachrichtigungen.

> Self-Healing kann viele Incidents vollständig verhindern.

> Alarmkorrelation unterstützt die Ursachenanalyse.

> Die richtige Information muss zur richtigen Zeit die richtige Person erreichen.

Verwandte Seiten

- 10.1 Ziele und Grundlagen des Monitoring and Event Management
- 10.2 Event-Typen, Filter und Priorisierung
- 10.4 Monitoring-Werkzeuge, Dashboards und Kennzahlen
- 10.5 Zusammenspiel mit Incident, Problem, Change und Service Configuration Management
- Incident Management
- Problem Management
- Service Configuration Management
- Continual Improvement

Quellen und Versionsstand

Offizielle Grundlagen

- PeopleCert – ITIL Practice Guide: Monitoring and Event Management
- ITIL Foundation – Version 5

Einordnung

Die dargestellten Alarmierungsstrategien, Eskalationsmodelle, Self-Healing-Beispiele und Automatisierungen sind herstellerneutrale Praxisempfehlungen. ITIL definiert die Grundprinzipien des Monitoring and Event Management, schreibt jedoch keine konkreten Alarmstufen, Eskalationswege oder Automatisierungsmechanismen vor. Diese werden organisationsabhängig anhand der Servicekritikalität und der eingesetzten Monitoring- und ITSM-Werkzeuge umgesetzt.

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