7.1 Knowledge Management – Ziele, Begriffe und Grundlagen

> **Kurz erklärt**
>
> Knowledge Management sorgt dafür, dass wichtiges Wissen im IT-Betrieb gesammelt, geprüft, strukturiert, auffindbar und nutzbar gemacht wird.
>
> Ziel ist, dass Mitarbeitende, Service Desk, Fachgruppen und Benutzer nicht jedes Problem immer wieder neu lösen müssen.
>
> Gute Wissensnutzung verbessert Supportqualität, verkürzt Bearbeitungszeiten, reduziert Wiederholungsfehler und macht IT-Services stabiler.

Warum Knowledge Management wichtig ist

Im IT-Betrieb entsteht täglich Wissen.

Beispiele:

- bekannte Fehlermeldungen,
- Workarounds,
- Standardlösungen,
- Installationsanleitungen,
- Betriebsrunbooks,
- Eskalationswege,
- bekannte Fehler,
- Anleitungen für Benutzer,
- technische Prüfschritte,
- Lessons Learned aus Incidents,
- Ergebnisse aus Problem Management,
- Änderungen nach Releases,
- Sicherheits- und Betriebshinweise.

Wenn dieses Wissen nicht dokumentiert wird, entstehen typische Probleme:

- dieselben Fragen werden immer wieder gestellt,
- Service Desk analysiert bekannte Fehler jedes Mal neu,
- Lösungen hängen von einzelnen Personen ab,
- neue Mitarbeitende brauchen länger,
- Workarounds werden uneinheitlich angewendet,
- Benutzer erhalten widersprüchliche Antworten,

- Wissen geht bei Krankheit, Urlaub oder Wechsel verloren,
- und Incidents dauern länger als nötig.

Knowledge Management macht Wissen wiederverwendbar.

Ziel von Knowledge Management

Knowledge Management soll sicherstellen, dass richtiges Wissen zur richtigen Zeit für die richtige Zielgruppe verfügbar ist.

Ziele sind:

- Wissen auffindbar machen,
- Lösungen wiederverwendbar machen,
- Bearbeitungszeiten reduzieren,
- Fehler bei Standardfällen vermeiden,
- Self-Service ermöglichen,
- Service Desk entlasten,
- Qualität von Antworten verbessern,
- neue Mitarbeitende schneller einarbeiten,
- Lessons Learned sichern,
- und kontinuierliche Verbesserung unterstützen.

Knowledge Management ist kein reines Dokumentationsarchiv.

Es ist ein aktiver Bestandteil des Servicebetriebs.

Wissen ist nicht gleich Dokumentation

Dokumentation kann Wissen enthalten.

Aber nicht jede Dokumentation ist automatisch nützliches Wissen.

Dokumentation	Nutzbares Wissen
irgendwo abgelegt	auffindbar
sehr technisch oder unklar	zielgruppengerecht
veraltet	geprüft und aktuell
ohne Zusammenhang	mit Service, CI oder Prozess verknüpft
nur für Spezialisten verständlich	für passende Zielgruppe verständlich
nicht gepflegt	mit Owner und Review versehen

Beispiel:

Ein 80-seitiges Betriebshandbuch kann wichtig sein.

Für den Service Desk ist aber oft ein kurzer Artikel mit Symptomen, Prüfschritten, Workaround und Eskalationsweg hilfreicher.

Arten von Wissen

Im IT-Betrieb gibt es unterschiedliche Wissensarten.

Wissensart	Beispiel
Benutzerwissen	Anleitung zum Zurücksetzen des Passworts
Service-Desk-Wissen	Prüfschritte bei VPN-Problemen
Fachwissen	technische Analyse einer Datenbankstörung
Betriebswissen	Runbook für Neustart eines Dienstes
Known-Error-Wissen	bekannter Fehler mit Workaround
Change-Wissen	Hinweise nach einem Update
Sicherheitswissen	Umgang mit verdächtigen Mails
Servicewissen	Zuständigkeiten, Supportzeiten und Abhängigkeiten

Diese Wissensarten benötigen unterschiedliche Form, Sprache und Zugriffsbeschränkung.

Explizites und implizites Wissen

Explizites Wissen

ist dokumentiert und kann weitergegeben werden.

Beispiele:

- Knowledge-Artikel,
- Runbook,
- Checkliste,
- FAQ,
- Servicebeschreibung,
- Known Error,
- Installationsanleitung.

Implizites Wissen

ist in den Köpfen von Personen vorhanden.

Beispiele:

- Erfahrung mit einem instabilen System,
- bekannte Eigenheiten eines Lieferanten,
- typische Fehler nach Updates,
- inoffizielle Abhängigkeiten,
- bewährte Diagnosewege.

Knowledge Management versucht, wichtiges implizites Wissen in nutzbares explizites Wissen zu überführen.

Wissen muss zielgruppengerecht sein

Nicht jede Zielgruppe benötigt dieselben Informationen.

Zielgruppe	braucht typischerweise
Benutzer	einfache Anleitung, klare Schritte, keine internen Details
Service Desk	Symptome, Prüfschritte, Workaround, Eskalationsweg
Fachteam	technische Details, Logs, Konfigurationen, Ursachen
Management	Risiko, Auswirkungen, Status, Entscheidungspunkte
Security Team	Schutzmaßnahmen, Meldewege, Klassifizierung
Lieferant	technische Nachweise, Versionen, Fehlerbild

Ein Benutzerartikel sollte keine internen Admin-Schritte enthalten.

Ein interner Runbook-Artikel darf technischer und detaillierter sein.

Beispiele für Knowledge-Artikel

Mögliche Artikeltypen:

- „VPN verbindet nicht nach Ruhezustand“
- „Passwort über Self-Service zurücksetzen“
- „Druckwarteschlange sicher neu starten“
- „TLS-Zertifikat erneuern“
- „Known Error: Client-Version 5.8 verliert Tunnelzustand“
- „Runbook: Datenbankdienst kontrolliert neu starten“
- „Checkliste: Server nach Update prüfen“
- „FAQ: MFA bei neuem Smartphone einrichten“
- „Service Desk: Erstdiagnose bei DNS-Problemen“
- „Benutzerinformation: Wartungsfenster Mitarbeiterportal“

Wichtig ist, dass Titel und Inhalt Suchbegriffe aus der Praxis berücksichtigen.

Knowledge Base

Eine Knowledge Base ist ein strukturierter Ort für Wissen.

Sie kann enthalten:

- Benutzeranleitungen,
- interne Supportartikel,
- technische Runbooks,
- FAQ,
- Known Errors,
- Workarounds,
- Checklisten,
- Troubleshooting-Guides,
- Servicebeschreibungen,
- Release-Hinweise,
- Sicherheitsanweisungen.

Das Werkzeug ist nicht entscheidend.

Möglich sind:

- ITSM-System,
- Wiki,
- Self-Service-Portal,
- Dokumentationsplattform,
- Knowledge-Modul,
- internes Handbuch.

Wichtig sind Auffindbarkeit, Aktualität, Qualität und Pflegeverantwortung.

Knowledge Management und Self-Service

Knowledge Management unterstützt Self-Service.

Benutzer können einfache Anliegen selbst lösen, wenn Informationen verständlich bereitstehen.

Beispiele:

- Passwort zurücksetzen,
- MFA neu einrichten,
- WLAN verbinden,
- Standardsoftware beantragen,
- Drucker hinzufügen,
- Status einer Störung prüfen,
- häufige Fehlermeldung verstehen.

Self-Service funktioniert nur, wenn Artikel:

- leicht auffindbar,
- verständlich formuliert,
- aktuell,
- kurz genug,
- und praktisch getestet sind.

Ein schlechter Self-Service erzeugt zusätzliche Rückfragen.

Knowledge Management und Service Desk

Der Service Desk profitiert besonders von guter Wissensbasis.

Nützlich sind:

- Standardantworten,
- Diagnosefragen,
- Prüfschritte,
- bekannte Symptome,
- Workarounds,
- Eskalationskriterien,
- Priorisierungshinweise,
- Kommunikationsvorlagen,
- Links zu Runbooks,
- Known Errors,
- Serviceinformationen.

Dadurch kann der Service Desk schneller und einheitlicher arbeiten.

Gleichzeitig liefert der Service Desk viele Hinweise, welche Artikel fehlen oder verbessert werden müssen.

Knowledge Management und Incident Management

Incident Management nutzt Wissen zur schnellen Wiederherstellung.

Beispiele:

- bekannter Workaround wird angewendet,
- Service Desk findet passenden Artikel,
- Incident wird mit Knowledge-Artikel verknüpft,
- Benutzer erhält geprüfte Anleitung,
- wiederkehrender Fehler wird schneller erkannt.

Incident Management liefert auch neues Wissen zurück.

Beispiele:

- neue Fehlermeldung,
- neuer Workaround,
- unklare Anleitung,
- falscher Artikel,
- fehlender Eskalationsweg,
- wiederkehrende Benutzerfrage.

Knowledge Management und Problem Management

Problem Management erzeugt wichtiges Wissen.

Beispiele:

- bekannte Ursachen,
- Known Errors,
- Workarounds,
- dauerhafte Lösungen,
- beitragende Faktoren,
- Lessons Learned,
- technische Prüfschritte,
- Risikohinweise.

Dieses Wissen sollte nicht nur im Problem Record bleiben.

Es muss so aufbereitet werden, dass Service Desk, Fachgruppen oder Benutzer es nutzen können.

Knowledge Management und Change Enablement

Changes verändern Services und damit auch Wissen.

Nach einem Change müssen möglicherweise aktualisiert werden:

- Benutzeranleitungen,
- Screenshots,
- Menüpfade,
- Runbooks,
- Known Errors,
- Workarounds,
- Servicebeschreibungen,
- Supportwege,
- Monitoringhinweise,

- Eskalationsregeln.

Ein Change ist nicht vollständig abgeschlossen, wenn die Technik geändert wurde, aber das Wissen noch den alten Stand zeigt.

Knowledge Management und Release Management

Releases bringen neue oder geänderte Funktionen.

Knowledge Management unterstützt durch:

- Release Notes,
- Benutzerinformationen,
- Service-Desk-Briefings,
- bekannte Einschränkungen,
- neue Workarounds,
- aktualisierte FAQ,
- Schulungsunterlagen,
- Hinweise auf geänderte Prozesse.

Nach einem Release steigen oft Rückfragen, wenn Wissen nicht rechtzeitig vorbereitet wurde.

Knowledge Management und Service Configuration Management

Configuration-Daten helfen, Wissen besser zuzuordnen.

Beispiele:

- Artikel gehört zu einem Service,
- Workaround gehört zu einem Known Error,
- Runbook gehört zu einem CI,
- Anleitung gehört zu einer Anwendungsversion,
- Sicherheitsartikel gehört zu einer bestimmten Benutzergruppe.

Ohne Service- oder CI-Bezug sind Knowledge-Artikel schwerer auffindbar und schwerer aktuell zu halten.

Lebenszyklus von Wissen

Wissen sollte einen Lebenszyklus besitzen.

```
Wissen erfassen
   ↓
Inhalt prüfen
   ↓
Artikel veröffentlichen
   ↓
Artikel nutzen
   ↓
Feedback sammeln
   ↓
Artikel aktualisieren
   ↓
Artikel archivieren oder entfernen
```

Wissen darf nicht dauerhaft ungeprüft online bleiben.

Veraltetes Wissen kann falsche Handlungen verursachen.

Wissensbedarf erkennen

Wissensbedarf entsteht zum Beispiel durch:

- häufige Incidents,
- wiederkehrende Benutzerfragen,
- neue Services,
- neue Releases,
- Major Incidents,
- bekannte Fehler,
- fehlende Runbooks,
- neue Mitarbeitende,
- Sicherheitsvorfälle,
- unklare Eskalationen,
- wiederholte Fehlbedienung,
- Service-Desk-Rückfragen.

Eine gute Frage ist:

> Welche Information hätte den letzten Incident schneller lösbar gemacht?

Wissen erfassen

Wissen kann erfasst werden aus:

- gelösten Incidents,
- Problem Records,
- Known Errors,
- Change Records,
- Release Notes,
- Betriebsdokumentation,
- Expertenwissen,
- Benutzerfeedback,
- Schulungen,
- Lieferanteninformationen,
- Monitoring-Analysen,
- Major-Incident-Reviews,
- Lessons Learned.

Wichtig ist, Wissen nicht nur zu sammeln, sondern zu strukturieren und nutzbar zu machen.

Wissen prüfen

Vor Veröffentlichung sollte geprüft werden:

- Ist der Inhalt fachlich korrekt?
- Ist die Zielgruppe klar?
- Ist der Artikel verständlich?
- Sind Schritte vollständig?
- Sind Risiken genannt?
- Sind Screenshots aktuell?
- Gibt es Sicherheits- oder Datenschutzprobleme?
- Ist der Artikel mit Service oder CI verknüpft?
- Ist ein Owner eingetragen?
- Gibt es ein Review-Datum?

Ungeprüfte Artikel können mehr Schaden als Nutzen verursachen.

Wissen veröffentlichen

Bei Veröffentlichung ist wichtig:

- passender Titel,
- klare Suchbegriffe,
- sinnvolle Kategorie,
- Zielgruppe,
- Zugriffsschutz,
- Version oder Gültigkeit,

- Owner,
- Veröffentlichungsdatum,
- Review-Datum,
- Verknüpfung zu Service, CI, Known Error oder Change.

Ein Artikel muss so veröffentlicht werden, dass er im richtigen Moment gefunden wird.

Wissen nutzen

Wissen sollte aktiv in Prozesse eingebunden sein.

Beispiele:

- Service Desk sucht beim Ticketstart nach Artikeln,
- Ticketformular schlägt passende Artikel vor,
- Benutzer erhalten Self-Service-Vorschläge,
- Known Errors sind mit Incidents verknüpft,
- Change-Abschluss verlangt Knowledge-Prüfung,
- Problem-Abschluss verlangt Workaround- oder Known-Error-Dokumentation.

Ungenutztes Wissen veraltet schneller.

Feedback sammeln

Feedback hilft, Knowledge-Artikel zu verbessern.

Mögliche Signale:

- Artikel wird häufig genutzt,
- Artikel wird selten gefunden,
- Benutzer bewerten Artikel schlecht,
- Service Desk ergänzt immer wieder dieselbe Erklärung,
- Artikel löst das Problem nicht,
- Suchbegriffe führen zu keinem Treffer,
- Artikel ist nach Release veraltet,
- Schrittfolge ist unklar.

Feedback sollte einfach möglich sein.

Wissen aktualisieren

Artikel müssen aktualisiert werden, wenn sich etwas ändert.

Auslöser:

- Change,
- Release,
- neuer Known Error,
- gelöster Known Error,
- neue Version,
- neuer Menüpfad,
- geänderte Zuständigkeit,
- geänderter Workaround,
- Sicherheitswarnung,
- Benutzerfeedback,
- neue Screenshots,
- geänderter Service.

Ein Artikel ohne Pflegeverantwortung veraltet meist schnell.

Wissen archivieren

Nicht mehr gültige Artikel sollten archiviert oder entfernt werden.

Beispiele:

- alter Workaround gilt nicht mehr,
- Service wurde abgeschaltet,
- Version ist nicht mehr im Einsatz,
- Known Error wurde behoben,
- Anleitung bezieht sich auf alte Oberfläche,
- Sicherheitsanweisung wurde ersetzt.

Archivierung kann sinnvoll sein, wenn spätere Nachvollziehbarkeit wichtig ist.

Veraltete Artikel sollten aber nicht mehr als aktuelle Lösung erscheinen.

Qualitätsmerkmale guter Knowledge-Artikel

Ein guter Artikel ist:

- fachlich korrekt,
- aktuell,
- klar strukturiert,
- verständlich,
- zielgruppengerecht,
- auffindbar,
- mit Service oder CI verknüpft,
- mit Owner versehen,
- risikoarm anwendbar,

- getestet,
- und regelmäßig überprüft.

Ein guter Artikel beantwortet nicht alles.

Er beantwortet genau das, was die Zielgruppe in dieser Situation braucht.

Struktur eines Supportartikels

Ein Service-Desk-Artikel kann enthalten:

- Titel,
- Kurzbeschreibung,
- Symptome,
- betroffener Service,
- Zielgruppe,
- Voraussetzungen,
- Prüfschritte,
- Lösung oder Workaround,
- Abbruchkriterien,
- Eskalationsweg,
- Risiken,
- verwandte Artikel,
- Owner,
- Review-Datum.

Nicht jeder Artikel benötigt alle Felder.

Die Struktur muss zum Zweck passen.

Struktur eines Benutzerartikels

Ein Benutzerartikel sollte besonders verständlich sein.

Mögliche Struktur:

- Was ist das Problem?
- Wen betrifft es?
- Was kann ich selbst tun?
- Schritt-für-Schritt-Anleitung,
- erwartetes Ergebnis,
- wann muss ich ein Ticket erstellen?
- welche Informationen soll ich im Ticket angeben?
- Link zu weiterem Support.

Benutzerartikel sollten keine unnötigen internen Fachbegriffe enthalten.

Struktur eines Runbooks

Ein Runbook ist eine betriebliche Schritt-für-Schritt-Anleitung.

Mögliche Inhalte:

- Zweck,
- betroffener Service,
- Voraussetzungen,
- benötigte Berechtigungen,
- Sicherheits- oder Risiko-Hinweise,
- genaue Schritte,
- Prüfpunkte,
- erwartete Ergebnisse,
- Abbruchkriterien,
- Rollback,
- Eskalation,
- Dokumentation nach Durchführung.

Runbooks sind besonders wichtig für wiederkehrende technische Betriebsaufgaben.

Titel und Suchbegriffe

Der Titel sollte so gewählt werden, dass Benutzer oder Service Desk den Artikel finden.

Ungeeignet:

> Fehler 0x80231-A Spezialfall

Besser:

> VPN verbindet nach Ruhezustand nicht mehr

Ungeeignet:

> Authentifizierungsproblem Anwendung X

Besser:

> „Anmeldung im Mitarbeiterportal schlägt fehl

Ein Artikel kann zusätzlich Synonyme oder typische Fehlermeldungen enthalten.

Artikel kurz und praktisch halten

Zu lange Artikel werden oft nicht gelesen.

Besser sind:

- klare Überschriften,
- kurze Schritte,
- konkrete Beispiele,
- Tabellen bei Varianten,
- Hinweise auf Risiken,
- Links zu tieferer Dokumentation.

Ein Artikel sollte nicht versuchen, ein ganzes Handbuch zu ersetzen.

Für tiefe technische Details kann auf Runbooks oder Fachteamdokumentation verwiesen werden.

Wissensfreigabe

Nicht jeder Artikel darf ohne Prüfung veröffentlicht werden.

Freigabe kann erforderlich sein bei:

- Sicherheitsanweisungen,
- Workarounds mit Risiko,
- Benutzerkommunikation,
- Anleitungen mit Administratorrechten,
- Datenschutzbezug,
- Änderungen an kritischen Services,
- rechtlichen oder vertraglichen Aussagen.

Die Freigabe muss aber zum Risiko passen.

Zu viel Freigabeaufwand verhindert schnelle Wissensnutzung.

Zugriffsschutz

Knowledge-Artikel können unterschiedliche Sichtbarkeit haben.

Beispiele:

Sichtbarkeit	Beispiel
öffentlich für Benutzer	MFA einrichten
intern für Service Desk	Prüfschritte bei VPN-Störung
nur Fachteam	Datenbank-Restart-Runbook
vertraulich	Sicherheitsvorfall-Prozess
Management	Risiko- und Statusübersicht

Zu viele Einschränkungen verhindern Nutzen.

Zu wenig Einschränkung kann Sicherheitsrisiken erzeugen.

Knowledge Owner

Ein Knowledge Owner ist verantwortlich für Inhalt und Aktualität eines Artikels.

Aufgaben:

- fachliche Korrektheit prüfen,
- Review durchführen,
- Änderungen einarbeiten,
- veraltete Inhalte entfernen,
- Feedback bewerten,
- Gültigkeit bestätigen,
- verwandte Artikel prüfen.

Ohne Owner veralten Artikel häufig.

Review-Datum

Ein Review-Datum zeigt, wann ein Artikel überprüft werden muss.

Besonders wichtig bei:

- sicherheitsrelevanten Artikeln,
- Workarounds,
- Known Errors,
- Benutzeranleitungen mit Screenshots,
- Runbooks,
- Artikeln zu kritischen Services,

- Lieferantenabhängigkeiten.

Ein Artikel kann auch vor dem Review-Datum aktualisiert werden, wenn ein Change oder Release den Inhalt verändert.

Wissensqualität messen

Mögliche Kennzahlen:

Kennzahl	mögliche Aussage
häufig genutzte Artikel	wichtiges Wissen
schlechte Bewertungen	Artikel unklar oder falsch
Suchanfragen ohne Treffer	Wissenslücke
Artikel ohne Owner	Pflegeproblem
Artikel ohne Review	Aktualität unsicher
Tickets trotz Artikel	Artikel nicht auffindbar oder nicht hilfreich
Workarounds ohne Artikel	Wissen nicht verfügbar
alte Artikel mit hoher Nutzung	Risiko veralteter Informationen

Kennzahlen müssen mit Feedback und Kontext bewertet werden.

Knowledge Management und KCS

Viele Organisationen orientieren sich an Ansätzen wie **Knowledge-Centered Service (KCS)**.

Grundidee:

- Wissen entsteht während der Arbeit,
- Lösungen werden beim Bearbeiten von Tickets erfasst,
- Artikel werden durch Nutzung verbessert,
- Wissen wird nicht erst nachträglich als Zusatzaufgabe gepflegt.

Wichtig:

KCS ist ein Praxisansatz und nicht identisch mit ITIL.

Er kann Knowledge Management sinnvoll ergänzen.

Wissen während der Arbeit erfassen

Gute Praxis ist:

- Wenn ein Incident gelöst wird, prüfe, ob Wissen dokumentiert werden sollte.
- Wenn ein Artikel fehlt, erstelle einen Entwurf.
- Wenn ein Artikel falsch ist, markiere oder korrigiere ihn.
- Wenn ein Workaround genutzt wird, verknüpfe ihn.
- Wenn ein Known Error entsteht, veröffentliche passende Informationen.
- Wenn ein Release kommt, aktualisiere relevante Artikel.

Wissen sollte dort entstehen, wo die Arbeit stattfindet.

Typische Fehler

Fehler 1

Knowledge Base wird als Ablage für alte Dokumente genutzt.

Fehler 2

Artikel haben keinen Owner.

Fehler 3

Artikel werden veröffentlicht, aber nie überprüft.

Fehler 4

Benutzerartikel sind zu technisch.

Fehler 5

Interne Runbooks sind für Benutzer sichtbar.

Fehler 6

Workarounds stehen nur in Tickets.

Fehler 7

Known Errors werden nicht mit Artikeln verknüpft.

Fehler 8

Service Desk nutzt vorhandenes Wissen nicht.

Fehler 9

Suchbegriffe passen nicht zur Sprache der Benutzer.

Fehler 10

Nach Changes und Releases werden Artikel nicht aktualisiert.

Fehler 11

Feedback wird gesammelt, aber nicht bearbeitet.

Fehler 12

Veraltete Artikel bleiben als aktuelle Lösung sichtbar.

Praxisbeispiel: VPN-Fehler

Situation

Benutzer verlieren nach dem Ruhezustand die VPN-Verbindung.

Problem

Der Service Desk erklärt den Workaround jedes Mal neu.

Knowledge Management

Ein Artikel wird erstellt mit:

- Symptomen,
- betroffener Clientversion,
- Prüfschritten,
- Workaround,
- Eskalationsweg,
- Known-Error-Verknüpfung.

Nutzen

Neue Incidents werden schneller gelöst.

Der Service Desk arbeitet einheitlicher.

Praxisbeispiel: MFA-Einrichtung

Situation

Viele Benutzer erstellen Tickets zur MFA-Einrichtung.

Analyse

Der Self-Service-Artikel ist schwer auffindbar und zu technisch.

Verbesserung

- neuer Titel mit Benutzerbegriffen,
- klare Schritt-für-Schritt-Anleitung,
- Screenshots,
- Abschnitt „Wann muss ich ein Ticket erstellen?“,
- Link direkt im Serviceportal.

Nutzen

Weniger Tickets und bessere Benutzererfahrung.

Praxisbeispiel: Change macht Artikel veraltet

Situation

Nach einem Release ändern sich Menüpfade im Mitarbeiterportal.

Problem

Der alte Knowledge-Artikel zeigt falsche Schritte.

Folge

Benutzer erstellen Tickets, obwohl die Funktion vorhanden ist.

Verbesserung

Change-Abschluss enthält künftig die Prüfung betroffener Knowledge-Artikel.

Praxisbeispiel: Runbook fehlt

Situation

Ein Dienst muss regelmäßig kontrolliert neu gestartet werden.

Problem

Nur ein erfahrener Administrator kennt die genaue Reihenfolge.

Risiko

Bei Abwesenheit wird der Dienst falsch neu gestartet.

Knowledge Management

Ein Runbook wird erstellt mit:

- Voraussetzungen,
- Schritten,
- Prüfungen,
- Abbruchkriterien,
- Rollback,
- Eskalationsweg.

Nutzen

Die Aufgabe wird sicherer und wiederholbarer.

Checkliste Knowledge-Artikel erstellen

- [ ] Zielgruppe festgelegt
- [ ] Problem oder Thema klar beschrieben
- [ ] betroffener Service genannt
- [ ] Symptome oder Anlass beschrieben
- [ ] Voraussetzungen genannt
- [ ] Schritte verständlich formuliert
- [ ] erwartetes Ergebnis genannt
- [ ] Risiken oder Einschränkungen beschrieben
- [ ] Eskalationsweg angegeben
- [ ] verwandte Artikel verknüpft
- [ ] Owner eingetragen
- [ ] Review-Datum gesetzt

Checkliste Benutzerartikel

- [ ] einfache Sprache

- [ ] keine unnötigen internen Begriffe
- [ ] klare Schrittfolge
- [ ] Screenshots aktuell, falls verwendet
- [ ] Suchbegriffe der Benutzer berücksichtigt
- [ ] Voraussetzungen genannt
- [ ] erwartetes Ergebnis beschrieben
- [ ] Ticketweg erklärt
- [ ] keine vertraulichen internen Informationen enthalten

Checkliste interner Supportartikel

- [ ] bekannte Symptome beschrieben
- [ ] betroffene Services oder CIs verknüpft
- [ ] Prüfschritte enthalten
- [ ] Workaround beschrieben
- [ ] Abbruchkriterien genannt
- [ ] Eskalationskriterien definiert
- [ ] Known Error verknüpft, falls vorhanden
- [ ] Risiken beschrieben
- [ ] Owner und Review-Datum vorhanden

Checkliste Knowledge-Pflege

- [ ] Artikel mit hohem Nutzungsgrad geprüft
- [ ] schlecht bewertete Artikel überprüft
- [ ] Suchanfragen ohne Treffer ausgewertet
- [ ] Artikel ohne Owner bereinigt
- [ ] Artikel ohne Review-Datum geprüft
- [ ] veraltete Artikel archiviert
- [ ] Artikel nach Changes aktualisiert
- [ ] Artikel nach Releases aktualisiert
- [ ] Feedback bearbeitet
- [ ] doppelte Artikel zusammengeführt

Bedeutung für Fachinformatiker für Systemintegration

Fachinformatiker besitzen viel praktisches Betriebswissen.

Dieses Wissen ist besonders wertvoll, wenn es dokumentiert und nutzbar gemacht wird.

Im Arbeitsalltag bedeutet das:

- Lösungen aus Incidents als Wissen sichern,
- wiederkehrende Fehler als Artikel dokumentieren,
- technische Runbooks schreiben,
- Workarounds sauber beschreiben,
- bekannte Risiken nennen,
- Service Desk mit Prüfschritten unterstützen,
- nach Changes Dokumentation aktualisieren,
- und veraltete Artikel melden oder korrigieren.

Gute Wissensarbeit spart nicht nur Zeit.

Sie macht IT-Betrieb zuverlässiger und weniger abhängig von Einzelpersonen.

Zusammenfassung

> Wissensbedarf erkennen
> ↓
> Wissen während der Arbeit erfassen
> ↓
> Zielgruppe und Zweck festlegen
> ↓
> Artikel strukturiert erstellen
> ↓
> fachlich prüfen
> ↓
> veröffentlichen und auffindbar machen
> ↓
> in Incident, Problem, Change und Service Desk nutzen
> ↓
> Feedback auswerten
> ↓
> Artikel aktualisieren oder archivieren
> ↓
> Wissensqualität kontinuierlich verbessern

Merksätze

>

> Wissen ist nur wertvoll, wenn es gefunden, verstanden und genutzt wird.

> Eine Knowledge Base ist kein Ablageort für beliebige Dokumente.

> Benutzer, Service Desk und Fachteam benötigen unterschiedliche Detailtiefe.

> Workarounds gehören nicht nur ins Ticket, sondern in nutzbare Knowledge-Artikel.

> Nach Changes und Releases muss Wissen aktualisiert werden.

> Veraltetes Wissen kann neue Incidents verursachen.

> Gute Wissensarbeit reduziert Abhängigkeit von Einzelpersonen.

Verwandte Seiten

- 7.2 Knowledge-Artikel, Runbooks und FAQ
- 7.3 Known Errors, Workarounds und Wissensnutzung im Service Desk
- 7.4 Self-Service und Benutzerwissen
- 7.5 Knowledge Management im Zusammenspiel mit Incident, Problem und Change
- Incident Management
- Problem Management
- Change Enablement
- Release Management
- Service Configuration Management
- Continual Improvement

Quellen und Versionsstand

Offizielle Grundlagen

- PeopleCert – ITIL Practice Guide: Knowledge Management
- PeopleCert – ITIL Practice Guide: Incident Management
- PeopleCert – ITIL Practice Guide: Problem Management
- PeopleCert – ITIL Practice Guide: Change Enablement
- PeopleCert – ITIL Practice Guide: Release Management
- ITIL Foundation – Version 5

Ergänzende Praxiseinordnung

- Knowledge-Centered Service als verbreiteter Praxisansatz für Wissensarbeit im Support

Einordnung

Die dargestellten:

- Artikeltypen,
- Checklisten,
- Rollenhinweise,
- Lebenszyklusmodelle,
- Qualitätskriterien,
- und Praxisbeispiele

sind herstellerneutrale Praxisempfehlungen.

ITIL schreibt keine universelle:

- Knowledge-Artikel-Vorlage,
- Review-Frequenz,
- Knowledge-Base-Struktur,
- Rollenverteilung,
- KCS-Nutzung,
- oder Toolauswahl

für alle Organisationen vor.

Die konkrete Umsetzung muss an:

- Services,
- Zielgruppen,
- Supportmodell,
- Sicherheitsanforderungen,
- Knowledge-Werkzeuge,
- Datenqualität,
- Organisation,
- Change-Modell,

- und verfügbare Fähigkeiten

angepasst werden.

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