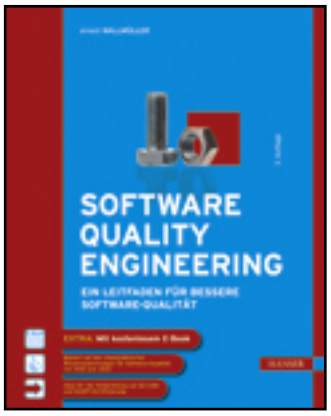
HANSER



Leseprobe

Ernest Wallmüller

Software Quality Engineering

Ein Leitfaden für bessere Software-Qualität

ISBN: 978-3-446-40405-2

Weitere Informationen oder Bestellungen unter

http://www.hanser.de/978-3-446-40405-2

sowie im Buchhandel.

4 Qualitätsmanagement auf Projektebene, im Betrieb und in der Wartung

4.1 QM auf Projektebene – Voraussetzungen und Grundlagen

Aus Sicht eines *Projektverantwortlichen* verstehen wir unter QM alle Tätigkeiten der Projektführung, welche die Projekt- und Qualitätsziele (Qualitätsplanung) sowie die Verantwortlichkeiten festlegen und diese durch Qualitätslenkung, Qualitätsprüfung/-überwachung und Verbesserungsmaßnahmen verwirklichen. Dazu muss nach dem Stand der Technik und der Wissenschaft eine unabhängige Qualitätsorganisation etabliert werden, die die Projektführung, konkret den Projektleiter und das Management, aber auch die Projektmitarbeiter unterstützt bzw. coacht und die Prozess- und Produktqualität sicherstellt (CMMI spricht von Prozess- und Produkt-Qualitätssicherung, PPQA).

QM aus *organisatorischer Sicht* bedeutet, eine entsprechende Qualitäts-(management)-Organisation mit entsprechenden Ressourcen aufzubauen und zu betreiben (siehe auch Kapitel 3). Dies ist oft ein Mangel in unreifen Organisationen, da die verantwortlichen Manager nicht begreifen (nicht budgetieren) wollen, dass eine Supportorganisation für Prozesse und Qualität heute zum Stand der Technik gehört bzw. in zahlreichen Normen (z. B. ISO 9001) und Best-Practice-Modellen (z. B. Enterprise SPICE/SUP.3 QA and QM) eindeutig gefordert wird. Bei Projekten ist es oft dringend notwendig, die Einhaltung der Spielregeln im Projekt regelmäßig zu hinterfragen (vergleichbar mit dem Fußball, wo ein Schiedsrichter und zwei Linienrichter für die Regelüberwachung eingesetzt werden – die Videoüberwachung wird derzeit in verschiedenen Gremien diskutiert, und es dürfte nur noch eine Frage der Zeit sein, wann sie eingeführt wird) bzw. den Zustand der Prozesse (den Zustand des Motors und den Ölstand) zu prüfen. Dass dafür ein Aufwand und Budget notwendig sind, dürfte wohl selbstverständlich sein in zivilisierten europäischen Unternehmungen.

Qualität im Projekt ist vielseitig und bezieht sich auf die Qualität der Projektprozesse (Projektabwicklungsprozess, Vorgehensmodell, Projekt-Lifecycle), auf die Qualität der eingesetzten Projektressourcen (Personal, Werkzeuge, Hilfsmittel, Infrastruktur, etc.) und auf die Qualität der entstehenden Zwischen- und Endprodukte. Darüber hinaus spielen die Umgebungsbedingungen für ein Projekt, die Vorlaufprozesse (Marketing, Sales, Portfoliomanagement), wie ein Projekt entstanden ist, bzw. die Prozesse am Ende oder nach Abschluss des Projekts für die entstehenden Produkte oder Services eine entscheidende Rolle. Insbesondere der klaglose

Übergang in den Betrieb (Operationsprozesse) bzw. in die Wartung (Maintenance-Prozesse) ist ein kritischer Erfolgsfaktor, dem gegenwärtig immer mehr Bedeutung und Aufmerksamkeit geschenkt wird.

Der japanische SQuBOK bildet diese qualitätsrelevanten Strukturen, Elemente bzw. Methoden für Projekte wesentlich genauer und besser ab als der amerikanische SQEBOK. Der japanische SQuBOK unterscheidet zwischen *SQM auf organisatorischer Ebene* (QM-System, Lifecycle-Management, Prozessbewertungen und -verbesserungen, Inspektionsmanagement, Auditmanagement, HR-/Bildungsmanagement, Governance- und Compliance-Management), dem *allgemeinen SQM auf Projektebene* (Entscheidungsmanagement, in CMMI: DAR, Einkaufs-/Lieferantenmanagement, in CMMI: SAM, Architekturmanagement, Risikomanagement und dem Projektmanagement) sowie dem *spezifischen SQM auf Projektebene* (Management des Qualitätsplans, Review-Management, Testmanagement und dem Management von Qualitätsanalysen und Assessments). Wir folgen für unser Buch dem japanischen Weg.

Auf Projektebene spielt der Qualitätsplan (Q-Plan, Synonyme: QS-Plan, QA Plan) eine zentrale Rolle. Er ist das Hilfsmittel, um das Entstehen der Qualität sicherzustellen und die Einhaltung der Prozesse bzw. das Entstehen der Arbeitsergebnisse nach den Regeln des Prozesses bzw. der Organisation zu sichern.

Der Qualitätsplan für eine Software ist in einigen Branchen Teil der Legitimation für den Einsatz der Software. Insofern ist er dort zwingender Teil der Qualitätssicherung des Unternehmens, das die Software erstellt und wartet. In anderen Branchen entstand das Bewusstsein für die Notwendigkeit eines Qualitätsplans für Software erst Ende der 90er-Jahre mit der Einführung von Best-Practice-Modellen.

Als Basis für den Qualitätsplan wird festgelegt, welche Risiken mit der Schaffung und dem Betrieb einer IT- bzw. Software-Lösung verbunden sind und wie verhindert wird, dass diese Risiken dem Projekt schaden (risikobasiertes Qualitätssichern), welche Qualitätsziele erreicht werden sollen, wie in der Herstellung verfahren und wie die entstehende oder vorhandene Qualität kontrolliert werden soll. Erst auf der Basis der abgeschlossenen Qualitätsplanung kann eine seriöse Validierung (inkl. Abnahme) des Systems durchgeführt werden.

Der Standard IEEE 730 für den Software Quality Assurance Plan (SQAP) beschreibt den Aufbau einer Qualitätssicherungsorganisation auf Projektebene. In einem SQAP werden die Entwicklungs-, Test- und Schulungsabläufe innerhalb eines Software-Projektes oder allgemein gültige Richtlinien einer Firma beschrieben, um die Software zu erstellen und in den Betrieb überzuführen.

Das Qualitätsmanagement auf Projektebene ist für das Coaching und den Support, die Entwicklung und Pflege eines Qualitätsplans verantwortlich.

4.1.1 Allgemeines, phasenübergreifendes Qualitätsmanagement auf Projektebene

Allgemeine phasen- bzw. iterationsübergreifende Aufgaben, die im Rahmen des QMs zu erfüllen sind, umfassen:

- Managen kritischer Entscheidungen (Beschlüsse, in CMMI: DAR)
- Beschaffungs- und Lieferantenmanagement (in CMMI: SAM)
- Architekturmanagement
- Risikomanagement
- Projekt-, Programm- und Portfoliomanagement

Managen kritischer Entscheidungen bedeutet, dass Entscheidungen durch die Verantwortlichen rechtzeitig, basierend auf Fakten und unter Berücksichtigung von Alternativen, bestmöglich ausgewählt, getroffen und auch umgesetzt werden. CMMI bietet dafür das Prozessgebiet DAR (Entscheidungsanalyse und Beschlussfassung). Anwendung findet das strukturierte und systematisch durchgeführte Entscheidungsmanagement (meistens methodisch auf Basis einer Nutzwertanalyse) beispielsweise bei der Auswahl von Lieferanten, bei komplexen Architekturentscheidungen und bei der Frage nach der Beendigung des Testens bzw. bei der Frage nach dem richtigen Zeitpunkt der Auslieferung von Releases.

QM für das Beschaffungsmanagement stellt Vorgaben, den Prozess und die Hilfsmittel für eine proaktive Beschaffung und den Einsatz von Produkten, Produktkomponenten, externen Ressourcen und Dienstleistungen bereit. Das QM hat dafür zu sorgen, dass die IT- bzw. Software-Lieferanten nach einem Vertrag arbeiten, regelmäßig über den Stand ihrer Arbeiten berichten, die Qualität ihrer Prozesse stimmt und ihre Arbeitsergebnisse den Anforderungen der Auftraggeber entsprechen. CMMI bietet dafür das Prozessgebiet SAM.

Architekturmanagement kümmert sich beispielsweise um einen „Architekturbaukasten", aus dem sich die Anwendungsentwickler bedienen sollen bzw. müssen. Eine Entscheidungsmatrix für Projekte dient dazu, die relevanten IT-Vorhaben direkt aus der Architekturstrategie abzuleiten. Und um die Architektur längerfristig konsistent zu halten, wird ein Prozess für die Ausrichtung der Architektur (Architektur-Alignment) festgelegt. Ein Regelwerk mit Architekturrichtlinien und -standards (Book of Standards) beschreibt sowohl die Architektur als auch die zu verwendenden Technikbausteine. Ein Review- und Genehmigungsprozess stellt sicher, dass die Projekte alle Architekturvorgaben auch tatsächlich berücksichtigen und umsetzen. Wesentliche Merkmale bei Architekturprüfungen sind die Wirksamkeit (geeignete Architektur ist verfügbar und wird genutzt mit minimalen Risiken und Kosten), Effizienz (IT-Infrastruktur und Applikationen unterstützen die Geschäftsziele kosten- und leistungseffizient) sowie Sicherheit (Architektur erfüllt alle relevanten Sicherheitsanforderungen).

Relevante Standards sind IEEE Std 1471-2000 (IEEE Recommended Practice for Architectural Description of Software-Intensive Systems) und dessen Erweiterung ISO/IEC 42010:2007, Systems and Software Engineering – Recommended Practice for Architectural Description of Software-Intensive Systems.

Erste Ansätze für eine Standardisierung von Prozessen für Architekturmanagement sind der Visual Architecting Process™ (VAP, www.bredemeyer.com), der arch42-Prozess (www.arc42.de, freies Portal für Software-Architekturen), der eigentlich nur sechs Architektur-Grundaufgaben umfasst, sowie der Best-Practice-Prozess Enterprise Architecture von Enterprise SPICE®.

Die Erfahrungen großer Firmen zeigen, dass die Qualität der IT- und Software-Architekturen für den Betrieb, die Wartung, aber auch für mögliche IT-Integrationen bei Fusionen von Firmen entscheidend und erfolgssichernd sind.

In den Bereichen *Projekt-*, *Programm-*, *Portfolio- und Risikomanagement* geht es darum, dass diese Prozesse mit ihren Aufgaben vor und zum Projektstart richtig aufgesetzt und den Regeln des Prozesses bzw. der Organisation entsprechend durchgeführt werden bzw. effizient ablaufen. Eine wichtige Hilfestellung ist die QM-Unterstützung des Projektleiters beim Tailoring seines Projektes, bei der Q- und Prozessprüfung und der Q- und Prozessüberwachung, sowie bei der Behandlung von Risiken durch Vermeidungs- und Notfallmaßnahmen. Die wesentlichen Best-Practice-Standards für die vier Managementdisziplinen sind Enterprise SPICE®, OPM3®, P3M3® sowie eingeschränkt CMMI® und PRINCE 2® (für Projekt- und Risikomanagement).

4.1.2 Projektphasenspezifisches Software-QM

Zu diesem Bereich des QMs gehören folgende Aufgaben:

- Management des Qualitätsplans
- Management von Prüfungen
- Testmanagement
- Qualitätsanalysen

Management des Qualitätsplans

Bei der Entwicklung des Qualitätsplans stehen unter anderem die Lieferobjekte mit Erfüllungs- und Validierungskriterien, Qualitätskontroll- und Qualitätssicherungsaktivitäten im Mittelpunkt. Der Qualitätsplan schafft die Voraussetzungen und die Sicherheit zur Fertigstellung der Lieferobjekte und sorgt für eine ausreichende Verifikation dieser Artefakte. Ebenso werden Maßnahmen aufgeführt, welche die Qualität der Lieferungen sichern.

Der Qualitätsplan beschreibt umfassend alle Aktivitäten, die zur Qualitätssicherung und -kontrolle im Projekt eingesetzt werden. Die terminlichen Aspekte zur Implementierung dieser Tätigkeit müssen identifiziert und in den Arbeitsplan (Terminplan) eingefügt werden, damit sichergestellt ist, dass diese auch tatsächlich durchgeführt werden.

Der Inhalt des Qualitätsplans nach IEEE Std. 730 umfasst:

- Zweck des Q-Managements für das Projekt mit einer Projektbeschreibung, den Qualitätszielen und Qualitätsrisiken, dem Lifecycle-Modell, Prioritäten, Umfang der QS-Aktivitäten
- Referenzierte Dokumente bzw. eine vollständige Liste aller Dokumente, welche im Q-Plan erwähnt werden, insbesondere zu verwendende Standards und Vorgaben aus dem Qualitätsmanagementsystem der übergeordneten Organisation
- Management mit Qualitätsorganisation und Matrix der Aufgaben, Kompetenzen und Verantwortung (AKV-Matrix) für QM im Projekt
- Minimale Produktdokumentation mit Identifikation, Ablage und Dokumentenstandards
- Festlegung von Normen, Verfahren, und Konventionen, insbesondere für Messungen

- Reviews, Audits und Tests (können bei größeren Projekten in einem eigenen Prüfplan definiert werden)
- Problem-/Fehler-Meldewesen, organisatorische Vereinbarungen über Korrekturmaßnahmen, Änderungsverfahren
- Entwicklungswerkzeuge, -methoden und -vorgehen
- Konfigurations- und Versionsverwaltung von Code, Dokumenten und sonstigen Artefakten
- Datenträgersteuerung
- Lieferantenmanagement
- Aufbewahrung, Pflege und Rückzug von Projektdokumenten
- Training (kann in einen eigenen Trainingsplan ausgelagert werden)
- Risikomanagement (kann bei Bedarf in einen eigenen Risikomanagementplan ausgelagert werden)

Im deutschsprachigen Arbeitsumfeld ist es auch üblich, den Qualitätsplan und weitere wichtige Pläne bzw. alle für das Projekt relevanten Informationen in einem „Projekthandbuch" zusammenzufassen. International wird hingegen nur von Projektmanagementplänen gesprochen (siehe IEEE 1058 Standard for Software Project Management Plans).

Management von Prüfungen

Erfolgt über den Prüfplan, der bei Großprojekten entweder ein eigener Plan oder Teil des Qualitätsplans ist. Aus der Auflistung aller Q-Tätigkeiten wird der Prüfplan herausgearbeitet. Der Zweck dieses Plans besteht darin, dass

- vertragliche Verpflichtungen erfüllt werden;
- alle im Voraus wissen, wann was wie durch wen geprüft wird;
- den Entscheidungsinstanzen aufgezeigt wird, wie, wann und welche Lieferobjekte geprüft werden (Prüfplan, siehe Bild 4.1, nächste Seite);
- die Kontrolle der Qualität der Lieferobjekte im Detail geplant und
- die Transparenz durch Prüfberichte geschaffen wird.

Der Prüfplan stellt die erforderliche Vorarbeit dar, um bei den Prüfungen nichts zu übersehen; ohne sorgfältige Prüfungsvorbereitung ist das Prüfergebnis evtl. gefährdet.

Testmanagement

Einen essenziellen Bereich innerhalb der Qualitätssicherung stellt das Testmanagement dar. Das Testmanagement ist für die Planung und Koordination aller Tätigkeiten sowie der an der Vorbereitung und Durchführung der Tests beteiligten Personen in der Testphase verantwortlich. Es sorgt für die rechtzeitige Planung und umfassende Organisation von Testmaßnahmen wie z. B. dem Aufbau und der Bereitstellung von Testfällen, der Durchführung von Einzeltests, Modultests, Integrationstests sowie Regressionstests, aber auch der Bereitstellung der Testorganisation und der Testinfrastruktur. Ebenso ist der Performancetest wesentlicher Bestandteil von Testszenarien (siehe dazu auch Kapitel 6).

Phase	Nr.	Lieferobjekte	Abschluss	Status		Kontroll-datum	Kontrolltechnik	Status	
Projekt-Impuls	AP 04	Projektantrag	25.01.2003	📄	✓	25.01.2003	Vernehmlassungs-verfahren	📄	👍
Initialisierungs-phase	AP 01	Projektplanung	15.02.2003	📄	✓	15.02.2003	Technischer Review	📄	👍
	AP 12	Anforderungskatalog	15.02.2003	📄	✓	15.02.2003	Walkthrough	📄	👍
	AP 11	Business Case	15.02.2003	📄		15.02.2003	Technischer Review	📄	
	AP 05	Projektauftrag	22.02.2003	📄	✓	22.02.2003	Vier-Augen-Prinzip	📄	
Konzeptions-phase	AP 24	Ist-Analyse	22.03.2003	📄		22.03.2003	Eigenkontrolle	📄	
	AP 27	Lösungsvarianten	12.04.2003	📄		12.04.2003	Vernehmlassungs-verfahren	📄	
	AP 28	Lösungsbewertung	19.04.2003	📄		19.04.2003	Vier-Augen-Prinzip	📄	

📄 = geplant ✓ = erstellt 👍 = geprüft

BILD 4.1 Beispiel Prüfplan

Ein gutes Testmanagement ist entscheidend für den Erfolg eines Software-Projekts. Das Testmanagement ist verantwortlich für das Erreichen der projektspezifischen Qualitätsziele und die Auswahl geeigneter Verfahren. Es beinhaltet die Konzeption, Überwachung sowie die Durchführung der Tests. Folgende Problemfelder im Testmanagement treten häufig auf:

- Komplexität der Software wird unterschätzt;
- die Vorbereitungsdauer auf die Tests ist ungenügend;
- fehlende oder ungenügende Testautomatisierung;
- ungenügender Einsatz von Testmethoden.

Der Nutzen eines optimierten Testmanagements besteht in der frühzeitigen Findung der Fehler (bevor sie der Kunde findet), sicheren Erreichung der Projektziele in einer vertraglich geforderten Nachweispflicht, in gut motivierten Testern und Entwicklern sowie eingesparten Entwicklungskosten.

Qualitätsanalysen

Qualitätsanalysen können auf das Produkt bzw. auf den Prozess der Erstellung, Wartung und des Betriebs von Software angewandt werden. Der prozessorientierte Aspekt wird gegenwärtig über Assessments, die beispielsweise auf den Best-Practice-Modellen CMMI, Automotive Spice oder ISO 15504 beruhen, abgedeckt. Mit ihnen werden auch Reifegrad-Bestimmungen von einzelnen Prozessen oder ganzen Organisationseinheiten durchgeführt (siehe dazu

[Wall06], [Kneu09]). Das Qualitätsmanagement kümmert sich dabei um die Organisation und Durchführung dieser Assessments (Assessment-Management).

Die produktorientierte Qualitätsanalyse verwendet Qualitätsattribute und Kriterien sowie Messansätze wie z. B. GQM, um Zusammenhänge oder Phänomene von Software-Produkten wie z. B. Architekturschwächen, Codemängel, Fehlerdichte oder Fehlerfindung näher zu untersuchen. Weitere Einsatzmöglichkeiten sind Code Comprehension und Reengineering von Software-Systemen. Es werden Techniken zur Analyse von Architekturen, des Designs von Komponenten und von Code eingesetzt. Bekannteste Technik der Codeanalyse ist die statische Software-Analyse. Sie wird durchgeführt, ohne die zu untersuchenden Programme wirklich auf dem Rechner auszuführen. In den meisten Fällen wird die Analyse an einer bestimmten Version des Quellcodes und in selteneren Fällen am Objektcode durchgeführt. Software-Werkzeuge dafür sind z. B. RATS, Yasca, CPD, FxCop, StyleCop, Checkstyle, FindBugs, PMD, Hammurabi, Sonar und Soot.

4.1.3 Ein Prozess für das Software-Qualitätsmanagement

Ein wirkungsvolles Qualitätsmanagement unterstützt ein Projekt in allen Phasen seiner Entwicklung. Beispielsweise wird durch Reviews in der Vorstudie, im Vorprojekt bzw. in der Elaborations- oder in den Konzeptphasen sichergestellt, dass alle Kunden- bzw. Produktanforderungen vollständig erfasst und konsistent beschrieben werden. Fehler werden so früh wie möglich, nicht erst während des Testens der Anwendung entdeckt.

Durch die Überwachung der Einhaltung von Richtlinien und Standards für das Projektvorgehen sowie die Initiierung von Maßnahmen zur Risikominimierung und Festlegung von Qualitätszielen werden die Qualität und damit die Kundenzufriedenheit mit den erstellten Projektergebnissen erhöht.

Diese Qualitätssicherungsmaßnahmen werden durch den Prozess Software-Qualitätsmanagement geplant, dokumentiert, gesteuert und überwacht. Sie werden aus den Anforderungen an die Projektergebnisse und den bereits zu Projektbeginn identifizierten Risiken abgeleitet.

Ein Teil der analytischen (prüfenden) Qualitätssicherungsmaßnahmen wird im Testprozess beschrieben. Diese umfassen alle dynamischen Prüfungen sowie statische Programmanalysen. Obwohl ein Teil der Qualitätssicherungsmaßnahmen auch eine überwachende, kontrollierende Funktion hat, stellen sie in erster Linie eine Unterstützung der Projektleitung zum Erreichen der Projektziele (in time, in budget, in quality) dar.

Nachfolgend geben wir einen Prozess für das Software-QM [Kneu09] an, der sich in der Praxis mehrfach bewährt hat. Die beteiligten Rollen sind der Projektmanager, Projektqualitätsmanager (PQM), der Qualitätsmanager für die Organisation, Qualitätsingenieur, Tester, Projektmitarbeiter (Requirements Engineer, Business Analyst, Designer, Programmierer) und das Management (Linie), das für die Projekte verantwortlich ist.

Eintrittskriterien für den Prozess sind Projekt-/Produktanforderungen sowie Qualitätsanforderungen (nichtfunktionale Anforderungen), QM-Policy der Organisation, organisatorische Standards und Prozesse, angemessene Ressourcen für das Projekt und die Vorgaben für die

durchzuführenden QM-Aufgaben wie z. B. Dokumentenmuster, Checklisten, Werkzeuge und Hilfsmittel.

Die Ergebnistypen des Prozesses Software-Qualitätsmanagement:

- der *Qualitätsplan,* der alle qualitätsrelevanten Festlegungen für ein Projekt enthält. Er wird möglichst bald nach Projektstart oder möglichst schon vor dem Projekt im Rahmen einer Vorstudie oder eines Vorprojekts, spätestens aber mit Beginn der Phase Grobkonzept erstellt und bei Bedarf im Laufe des Projektfortschritts weiter detailliert und angepasst;
- der *Prüfplan,* der eine verbindliche Auflistung aller Ergebnistypen, der durchzuführenden Prüfmaßnahmen nebst Ressourcen und Terminen enthält;
- der *Qualitätsstatusbericht* inklusive relevanter Metriken zur regelmäßigen Berichterstattung;
- der *Qualitätsabschlussbericht,* der das beendete Projekt unter verschiedenen Aspekten bewertet.

Für die Durchführung von Maßnahmen der Qualitäts- und Prüfplanung müssen in der Projektplanung Ressourcen berücksichtigt und auch bereitgestellt werden. Qualitäts-(sicherungs-)maßnahmen lassen sich in zwei Kategorien einteilen:

- *Konstruktive Qualitätssicherungsmaßnahmen* dienen der Erlangung der geeigneten Produkt- und Prozessqualität. Sie haben zum Ziel, die Qualitätsanforderungen zu erfüllen und sollen das Entstehen von Abweichungen, Fehlern und Qualitätsmängeln von vornherein verhindern. Hierzu werden geeignete System- und Software-Engineering-Verfahren, Prinzipien, Methoden, Formalismen und Werkzeuge vorgegeben.
- *Analytische Qualitätssicherungsmaßnahmen* sind jene, die der Erkennung und Lokalisierung von Abweichungen, Mängeln und Fehlern im weitesten Sinne dienen, d. h. alle Maßnahmen zur Prüfung, Bewertung und Beurteilung der Qualität.

Folgende Arbeitspakete (AP) sind für das Software-Qualitätsmanagement im Projektplan einzuplanen:

AP: Qualität vor Projekt sichern

Diese Aufgaben dienen der Festlegung, Durchführung und Auswertung von initialen Qualitätssicherungsmaßnahmen für die Ergebnisse der Vorstudie oder des Vorprojekts. Wesentliche Aktivität dabei ist:

- A1: Initiale Qualitätssicherung durchführen
  Meist werden hier Reviews der Vor- und Initialprojektdokumente durchgeführt. Es können aber auch Zielüberprüfungsworkshops bzw. Workshops zur Klärung der Bedürfnisse des Auftraggebers durchgeführt werden, um die spätere Anforderungsermittlung und Erhebung zu vereinfachen bzw. in geeigneter Weise zu initialisieren.

AP: Produkt- und Prozessqualität planen

Hier werden die im Projektverlauf durchzuführenden Qualitätsmaßnahmen festgelegt. Die Ergebnisse finden sich im Qualitätsplan wieder. Wesentliche Aktivitäten hierbei:

- A2: Qualitätsziele definieren

- A3: Qualitätsmaßnahmen für Produkt- und Prozessqualität planen und dokumentieren
- A4: Ressourcen für Qualitätsmaßnahmen planen, organisieren und bereitstellen

Für die Produktqualität ist es hilfreich, Arbeitsmittel in Form eines Attributsbaums nach ISO 9126 oder ISO 25000 und zugeordnete Bewertungshilfsmittel bereitzustellen, um die Qualitätsziele griffig formulieren zu können. Für die Prozessqualität sollte festgelegt werden, welches Qualitätsniveau, welche Prozessreife bzw. welcher Prozessreifegrad vorgegeben sind. Für A3 und die Produktsicherung ist es sinnvoll, die Projekt- bzw. Produktrisiken zu kennen. Dazu kann auch eine Projektumfeldanalyse dienen, die die relevanten Stakeholder mit ihren potenziellen Gefahren identifiziert. Für die Prozesssicherung wird eine ausreichende Anzahl von Miniassessments geplant. A4 ist einerseits mit der Projektleitung und dem Management und andererseits mit dem Qualitätsmanager der Organisation bzw. dem Leiter der Qualitätsingenieure abzustimmen.

AP: Produkt- und Prozessqualität sichern

Hier werden die geplanten Qualitätsmaßnahmen durchgeführt, gesteuert und überwacht. Wesentliche Aktivitäten:

- A5: Qualitätsmaßnahmen steuern und überwachen
- A6: Qualitätsmaßnahmen wie z. B. Reviews der projektbezogenen Ergebnisse oder Miniassessments zur Sicherstellung der Prozesseinhaltung planen und durchführen

Der Qualitätsingenieur stellt zusammen mit dem PQM sicher, dass beispielsweise die vereinbarten Reviews mit entsprechender Sorgfalt durchgeführt und die aufgetretenen Mängel auch tatsächlich behoben werden.

AP: Qualitätssicherung analysieren und auswerten

Hier werden die durchgeführten Qualitätsmaßnahmen analysiert und ausgewertet sowie der Qualitätsreport zur Verfügung gestellt. Wesentliche Aktivitäten:

- A7: Projektergebnisse aus Qualitätssicht analysieren und beurteilen
- A8: Qualitätsstatus des Produkts, der Prozesse und des Projekts bewerten, messen und berichten

AP: Qualitätsplanung aktualisieren

Auf Grundlage der Auswertungen werden der Qualitätssicherungsplan sowie der Prüfplan fortgeschrieben. Wesentliche Aktivität:

- A9: Qualitätsmaßnahmen korrigieren bzw. anpassen und ergänzen

AP: Projektabschlussbericht erstellen

Hier erfolgt rückblickend eine Beurteilung des Projekts unter qualitativen Gesichtspunkten. Wesentliche Aktivität:

- A10: Projektabschlussbericht unter Qualitätsaspekten erstellen

Der Prozess kann je nach Typ des Produkts (Hardware-, Elektronik- und/oder Software-System), Art der Systementwicklung (Vorgehensmodell) bzw. nach Reifegradstufe der Entwick-

lungsorganisation angepasst werden. Die Erfahrungen in der Praxis zeigen, dass mit einem Prozess für QM die Arbeit der Qualitätssicherung auf Projektebene transparenter und planbarer wird. Ebenso wird dem Management bewusst gemacht, dass ein Basisaufwand für QM-Tätigkeiten im Rahmen der Aufwandsschätzung und Projektplanung einzukalkulieren ist.

4.2 QM im Betrieb von Systemen bzw. Applikationen bzw. im Service Management

Unter Betrieb verstehen wir die Lokation, an der die Software bzw. Applikationen betrieben werden. Früher war dies nur das Rechenzentrum (Operationcenter), heute können die Server oder Server-Farmen überall stehen und über globale Netzwerke die Services den Endkunden am Arbeitsplatz oder zu Hause angeboten werden. In diesem Bereich vollzieht sich ein gewaltiger Wandel. Waren früher die eingesetzten Technologien (Hardware, Netzwerke, Betriebssysteme etc.) für den Betrieb der Applikationen und der Produkte entscheidend bzw. wettbewerbsbestimmend, so hat sich der Fokus auf die Qualität bzw. das Kosten-/Nutzenverhältnis des Erbringens von Services verlagert. Bei Letzteren verschiebt sich die Art der Services von reinen IT-Services zu Business-Services.

Unternehmen sind heutzutage oft in starkem Maße von der Verfügbarkeit und der Zuverlässigkeit ihrer Geschäfts-/IT-Services abhängig. Es wird erwartet, dass die IT nicht nur das Kerngeschäft des Unternehmens unterstützt, sondern auch einen Beitrag zum Wettbewerbsvorteil leistet. Die IT kann beispielsweise den Einsatz neuer Technologien ermöglichen, wodurch dem Unternehmen neue Marktchancen eröffnet werden. Letztendlich soll sie helfen, die Zielsetzung der gesamten Unternehmung zu verwirklichen. Sie muss daher in der Lage sein, flexibel auf den sich im Laufe der Zeit ändernden Bedarf an Geschäfts- und IT-Services zu reagieren.

Was ist ein Service (eine Dienstleistung)?

Eine Dienstleistung nach Wikipedia im Sinne der Volkswirtschaftslehre ist ein wirtschaftliches Gut, bei dem im Unterschied zur Ware nicht die materielle Produktion oder der materielle Wert eines Endproduktes im Vordergrund stehen, sondern eine von einer natürlichen Person oder einer juristischen Person zu einem Zeitpunkt oder in einem Zeitrahmen erbrachte Leistung zur Befriedigung von Bedürfnissen. Es wird zwischen personenbezogenen (z. B. Arzt oder Koch), sachbezogenen (Banken, Versicherungen) und originären Dienstleistungen (z. B. Lieferung nichtmaterieller Güter, Abschleppdienst, IT-Serviceprovider) unterschieden. Der Erbringer einer solchen Leistung wird als Dienstleister bezeichnet.

Ein Service in der CMMI-Terminologie ist ein nicht berührbares, dinghaftes, nicht speicherbares Produkt.

Services werden gleichzeitig produziert und konsumiert. Services etablieren eine Beziehung, die durch Service-Vereinbarungen geregelt ist. Services werden durch ein Service-System erbracht oder geliefert. Ein Service-System (z. B. ein Kundendienstsystem) ist eine Konfiguration von technologischen und organisatorischen Elementen bzw. Verbrauchsgütern, um Dienstleistungen zu liefern, die die Bedürfnisse und Anforderungen von Kunden zufriedenstellen.

Services haben im Gegensatz zu Produkten einen anderen Geschäftsrhythmus. Ein Produkt wird während eines Projekts entwickelt (Entwicklungsdauer relativ lange) und dann geliefert. Ein Service wird entwickelt (einmalig, relativ kurz) und dann geliefert bzw. erbracht (relativ lange).

Was beeinflusst die Qualität einer Dienstleistung?

Services kommen in Interaktion zwischen Erbringer und Abnehmer (Kunde) zustande. Die Dienstleistung, z. B. Frisur erstellen, wird gleichzeitig erbracht und konsumiert. Die Qualität eines Service, z. B. Frisur gestalten, kann nicht im Voraus beurteilt werden. Eine Bewertung ist erst nach der Erbringung möglich. Der Friseur zeigt anhand eines Spiegels dem Kunden, welche Leistung er erbracht hat. Zudem ist die Qualität eines Service stark von dem Verhältnis zwischen Anbieter und Kunde abhängig. In der Regel suchen wir uns einen Friseur aus, der unseren Anforderungen bzw. Erwartungen entspricht. Anders als bei einem Herstellungsprozess können Kunden und Anbieter während der Erbringung noch Einfluss auf den Service nehmen, beispielsweise kann die Haarlänge beim Haarschnitt noch verkürzt werden. Wie ein Kunde den Service empfindet und was der Dienstleister zu liefern glaubt, hängt in erheblichem Maße von den persönlichen (und damit häufig subjektiven) Eindrücken sowie den bestehenden Erwartungen ab. Dies ist gerade bei Frisuren und Friseuren eindrücklich festzustellen. Bei komplexen Dienstleistungen wie beispielsweise der weltweiten Einführung eines neuen Produkts oder einer Dienstleistung (die Einführung ist hier der Service) ist daher die Klärung und Kommunikation von Bedürfnissen bzw. Anforderungen ein erfolgskritischer Faktor.

Bei der Erbringung von Services steht die Wahrnehmung des Kunden bzw. die Transparenz im Mittelpunkt. Um die Qualität eines Service zu ermitteln, wird sich der Kunde in den meisten Fällen folgende Fragen stellen [Ande10]:

- Entspricht der Service inhaltlich meinen Erwartungen?
- Kann ich beim nächsten Mal mit einem ähnlichen Service rechnen?
- Wird der Service zu vertretbaren Kosten erbracht?

Ob der Service den Erwartungen entspricht, hängt nicht allein von der Servicequalität des Anbieters ab, sondern vor allem davon, wie genau der Service zuvor im Dialog mit dem Kunden vereinbart wurde. Eine ständige Kommunikation mit dem Kunden ist erforderlich, um die Services immer wieder den Vorstellungen des Kunden angleichen zu können. Deshalb muss dafür gesorgt werden, dass Kunde und Anbieter das gleiche Verständnis über die zu erbringenden Services haben. Beispielsweise kann in einem Restaurant der Gast anhand der Speisekarte den Umfang des Angebots erkennen und danach auswählen; der Kellner steht für Fragen und Änderungswünsche zur Verfügung. Außerdem informiert der Kellner über aktuelle Tagesangebote oder macht Menüvorschläge für Gäste. Während sich der Gast im Restaurant aufhält, erkundigt sich ein guter Kellner öfters danach, ob er mit der bisherigen Leistung

zufrieden war und ob er weitere Wünsche hat. Der Kellner ist aktiv damit beschäftigt, das Angebot und die Nachfrage aufeinander abzustimmen. Seine Erfahrungen mit den Gästen fließen in die Präsentation des Angebotes ein – und ist im Interesse eines entsprechenden Trinkgeldes bemüht, den Service permanent zu verbessern.

Um eine gute Qualität liefern zu können, muss der Anbieter ständig neu ermitteln, wie der Service aufgenommen wurde und was der Kunde in Zukunft erwartet. Was für den einen selbstverständlich ist, wird von manch anderem als außergewöhnlich aufgefasst. Was zunächst außergewöhnlich ist, wird für einen Kunden schnell zum Normalfall, da er sich daran gewöhnt hat. Es ist also für den Anbieter wichtig, die Kundenempfindungen hinsichtlich der Qualität zu beobachten. Nur dann kann er entscheiden, ob zum Beispiel sein Service angepasst, der Kunde besser informiert oder der Preis korrigiert werden muss. Daher sind regelmäßige, aber auch zufällig geplante Kundenbefragungen und ein ständiger Dialog mit Kunden bzw. Kundenvertretern besonders wichtig.

Was umfasst QM in serviceorientierten Organisationen?

Grundsätzlich gelten dieselben Aussagen wie bereits in Kapitel 2. Auch der Deming-Zyklus mit Plan-Do-Check/Study-Act eignet sich hervorragend, um Serviceleistungen zu erbringen und zu verbessern. Die Rolle der Kommunikation und deren Planung scheint uns aber besonders wichtig zu sein, da eine gute Servicequalität nachweislich auch zu einer starken Kundenbindung führt. Andermatten [Ande10| und insbesondere ITIL V3.0 empfehlen, die Kommunikation zu analysieren und anhand eines Kommunikationskonzepts zu verbessern. Ausgangspunkt sollte ein klares Bild (eine Vision) der Kommunikation sein, das es zu erreichen gilt. Identifikation und Analyse der Stakeholder im Kommunikationsprozess zur Serviceerbringung, Erstellung sowie Pflege eines Kommunikationsplans mit Zielen, Methoden und regelmäßigen Kommunikationsveranstaltungen sowie dessen Umsetzung und Überwachung sind wichtige Elemente der Qualitätsgestaltung.

Eine wesentliche Rolle spielt das Prozessmanagement und insbesondere die Messung und Steuerung von Prozessen zur Serviceerbringung. Kritischer Erfolgsfaktor ist die richtige Auswahl der Metriken, auch Key-Performance-Indikatoren genannt, sowie die Erstellung eines Service-Messplans und ein kontinuierlicher Mess- und Analyseprozess für Services und Serviceprozesse.

Auch die Reifesteigerung von Service-Organisationen ist in den letzten Jahren ein Thema geworden (beispielweise durch ISO 20000, CMMI-SVC oder den Einsatz des EFQM-Modells). Bei der Ermittlung des Reifegrads der Organisation reicht es jedoch nicht aus, lediglich die Anbieter des Service zu betrachten. Die Reifegradebene des Kunden spielt ebenfalls eine wichtige Rolle. Wenn innerhalb der Beziehung zwischen Anbieter und Kunde große Unterschiede hinsichtlich des Reifegrads bestehen, kann dies zu Problemen in der Kommunikation sowie zu Abstimmungsschwierigkeiten in Bezug auf die gegenseitige Erwartung führen. Um diesen Problemen aus dem Wege zu gehen, empfiehlt es sich daher, die Zusammenarbeit auf einer gemeinsamen Ebene zu beginnen und die Kommunikation gegebenenfalls an eine niedrigere Ebene anzupassen, oder beide Partner (Kunde und Lieferant) beabsichtigen, ein gemeinsames Reifeniveau, z. B. CMMI Maturity Level 3, zu erreichen.

Was sind nun typische Probleme der Praxis im Service Management nach einer Studie der IDC aus dem Jahre 2010 [IDC10]?

- Ca. 80 % aller Störungen werden durch ungeplante und nicht autorisierte Änderungen verursacht (Change Management).
- IT-Organisationen wenden ca. 20 % ihrer Kapazität für die Behebung von Störungen und Problemen auf.
- Mehr als 51 % der Unternehmen gaben an, dass Störungen aufgrund schlechter Performance steigen.
- 74 % der IT-Operationsbereiche wollen prozessorientiert agieren, aber nur ca. 14 % sind bereits prozessorientiert aufgestellt (Prozessmanagement, Prozessfokus).

Anbieter von IT-Services können es sich eigentlich nicht erlauben, ihren Fokus nur auf die Technik und ihre eigene interne Organisation zu beschränken. Sie müssen sich darüber hinaus auch mit der Qualität ihrer Services und der Reife/Qualität ihrer Prozesse auseinandersetzen und ein gutes Kundenverhältnis regelmäßig pflegen.

Wir gehen davon aus, dass der Service-Erbringer bemüht ist, eine relativ konstante Qualität kostengünstig zu liefern, indem er die Produktionsbedingungen sowie den eigentlichen Produktionsprozess standardisiert, gründlich überwacht und steuert. Für den Abnehmer der Services bzw. des Produktes gibt es oft leider keinerlei Transparenz. Allerdings bleibt meistens noch die Möglichkeit, die Entscheidung für den einen oder anderen Service-Erbringer von einer Reihe von Messkriterien, Standards oder auch Prüfsiegeln abhängig zu machen. Bei Analyse und Betrachtungen des Lebenszyklus eines Produktes sind daher die Phasen Herstellung, Vertrieb und Gebrauch/Service-Nutzung völlig getrennt voneinander zu betrachten.

Eines der am weitesten verbreiteten Best-Practice-Modelle im Zusammenhang mit IT-Service-Management ist ITIL. ITIL (IT Infrastructure Library, www.itil-officialsite.com, www.itil.org) hat zum Ziel, die Kerngeschäftsprozesse durch entsprechend ausgerichtete IT-Services zu unterstützen. Im Gegensatz zum Anwender, der die Services der IT zur Erfüllung seiner Arbeit nutzt, ist der Kunde derjenige, der die Services in Zusammenarbeit mit dem Service Level Management definiert und auch letztendlich bezahlt. Für diese beiden Gruppen von IT-Kontakten gibt es bei ITIL dedizierte Schnittstellen. Für den Kunden ist dies das Service Level Management. Für die Anwender steht der Service Desk als Single Point of Contact zur Verfügung, eine der zentralen Schaltstellen, um die Service-Nutzer zu unterstützen bzw. ihnen zu helfen.

Ziel der Prozesse des IT-Service-Managements ist es letztendlich, nachhaltig zur Qualität und Wirtschaftlichkeit von IT-Services beizutragen und dadurch auch die Leistungserbringung im Geschäft sowie den Gesamterfolg der Organisation sicherzustellen.

ITIL als bekanntester und wohl populärster Ansatz für das IT-Service-Management legt sich nicht auf die Art der Organisation fest, sondern beschreibt den Zusammenhang der Aktivitäten innerhalb der für jede Organisation erkennbaren Prozesse. Damit wird ein Gerüst für den Austausch von Erfahrungen zwischen verschiedenen Organisationen zur Verfügung gestellt. Zudem liefert dieser Ansatz einen Rahmen, um aus Erfahrungen innerhalb dynamischer Organisationen zu lernen und dadurch auch systematisch an Verbesserungen zu arbeiten.

ITIL ist eine herstellerunabhängige Sammlung von Best Practices, mit denen es IT-Organisationen über einen prozessorientierten skalierbaren Ansatz ermöglicht wird, Qualitäts- und Effizienzsteigerungen innerhalb ihrer IT-Prozesse zu erzielen und somit ihren Kunden einen

gleichbleibenden IT-Service zu liefern. Seit den 90er-Jahren hat sich ITIL zu einem internationalen De-facto-Standard entwickelt. Im Frühsommer 2007 erschien die ITIL-Version 3.0. Die Kernpublikationen von ITIL V3 (Service Lifecycle) bestehen aus 5 Büchern:

- Service Strategy (Service-Strategie)
- Service Design (Modelle für den produktiven Betrieb)
- Service Transition (Service-Implementierung bzw. Serviceüberführung)
- Service Operation (operativer Betrieb von Services)
- Continual Service Improvement (kontinuierliche Verbesserung von Services)

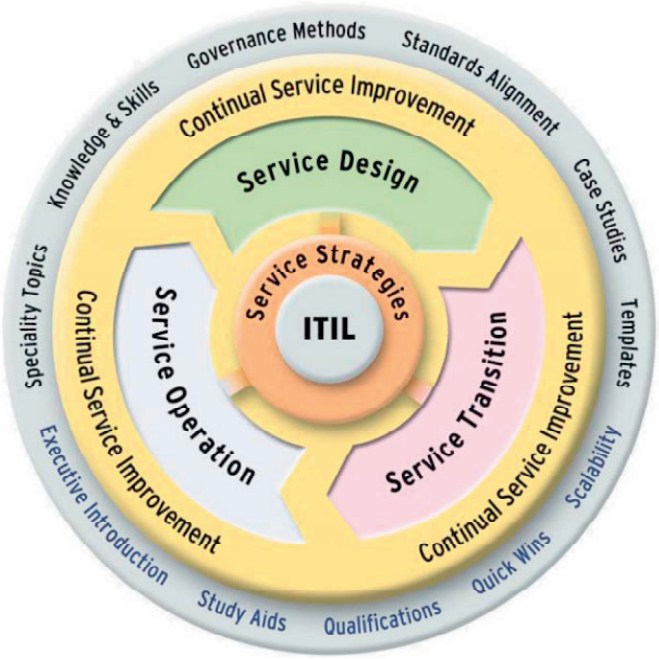


BILD 4.2
Überblick ITIL V3
(Copyright: Glenfis AG)

Service-Strategie (SST)

Mit SST wird eine Service-Strategie entwickelt und umgesetzt. Dazu braucht es das Demand Management, das IT Finance Management und das Service Portfolio Management. Das Ziel beim Service Portfolio Management besteht darin, eine Strategie für die Bereitstellung von Services für Kunden zu entwerfen; hierzu zählt auch ein Konzept für die Weiterentwicklung der Serviceangebote und Kompetenzen des Service-Providers. Das Ziel beim finanziellen Management von Services besteht im Schaffen der Voraussetzungen für die Budgetierung sowie die Kostenrechnung und Leistungsverrechnung des Service-Providers.

Service Design (SD)

SD plant und entwickelt neue oder modifiziert vorhandene Services oder Service-Management-Prozesse. Service Design besteht aus Service Catalogue Management (SCM), Service

Level Management (SLM), Risikomanagement, Capacity Management, Availability Management, IT Service Continuity Management (ITSCM), IT Security Management, Compliance Management, IT Architecture Management und Supplier Management.

Das Ziel von Service Catalogue Management besteht im Sicherstellen, dass ein Servicekatalog entwickelt und gepflegt wird, der präzise Informationen zu allen in Betrieb befindlichen und geplanten Services enthält. Das Service Catalogue Management versorgt alle weiteren Service-Management-Prozesse mit wesentlichen Informationen. Dabei handelt es sich um Angaben zu den Details der Services, ihrem aktuellen Status sowie zu ihren wechselseitigen Abhängigkeiten.

Das Ziel von Service Level Management (SLM) ist das Abschließen von Service-Level-Vereinbarungen mit den Kunden und Entwerfen von Services, die den vereinbarten Service-Level-Zielen entsprechen. Darüber hinaus ist das Service Level Management auch dafür verantwortlich sicherzustellen, dass alle Operational Level Agreements (Vereinbarungen auf Betriebsebene) und Underpinning Contracts (Verträge mit Drittparteien) zielführend sind. Das Service Level Management überwacht die Service Levels und berichtet laufend über deren Einhaltung.

Das Ziel des Risikomanagements besteht im Identifizieren, Bewerten und Überwachen von Risiken. Dazu gehört es, den Wert von Assets für das Unternehmen zu analysieren, mögliche Bedrohungen für diese Assets zu identifizieren und die jeweilige Gefährdung der Assets einzuschätzen.

Ziel des Capacity Managements ist es, sicherzustellen, dass die Kapazität der IT-Services und der IT-Infrastruktur ausreicht, um die vereinbarten Service-Level-Ziele wirtschaftlich zu erbringen. Das Capacity Management berücksichtigt alle Ressourcen, die erforderlich sind, um den IT-Service zu erbringen, und plant dabei die kurz-, mittel- und langfristigen Anforderungen von Geschäftsseite ein.

Ziel des Availability Managements ist das Definieren, Analysieren, Planen, Messen und Verbessern aller Faktoren, die für die Verfügbarkeit von IT-Services wesentlich sind. Das Availability Management ist verantwortlich dafür, dass die gesamte IT-Infrastruktur und alle Prozesse, Werkzeuge, Rollen usw. zum Erreichen der vereinbarten Verfügbarkeitsziele geeignet sind.

Das IT Service Continuity Management (ITSCM) managt Risiken, die gravierende Auswirkungen auf die IT-Services haben können. Es stellt sicher, dass der IT-Service-Provider auch im Falle außergewöhnlicher Ereignisse die in den Service Levels vereinbarten Minimalanforderungen bereitstellen kann; dies geschieht durch risikomindernde Maßnahmen und durch eine gezielte Wiederherstellungsplanung für die IT-Services. Das ITSCM sollte in einer Weise gestaltet sein, dass es das Business Continuity Management (BCM) unterstützt.

Das IT Security Management stellt sicher, dass alle Güter, Informationen, Daten und IT-Services eines Unternehmens jederzeit hinsichtlich ihrer Vertraulichkeit, Integrität und Verfügbarkeit geschützt sind. Es ist normalerweise in ein unternehmensweites Security Management eingebunden, das einen breiteren Wirkungsbereich als der IT-Service-Provider hat.

Das Compliance Management stellt sicher, dass die IT-Services, Prozesse und Systeme mit den Unternehmensrichtlinien und gesetzlichen Bestimmungen konform gehen.

Das IT Architecture Management entwickelt und pflegt einen Vorgehensplan für die mittel- und langfristige Weiterentwicklung der Technologielandschaft, der die Servicestrategie einbezieht und innovative Technologien berücksichtigt.

Das Supplier Management stellt sicher, dass alle Verträge mit Lieferanten die geschäftsseitigen Notwendigkeiten unterstützen und dafür sorgen, dass alle Supplier ihre vertraglichen Pflichten erfüllen.

Service Transition (ST)

ST managt die Überführung eines neuen oder geänderten Service oder eines neuen oder geänderten Service-Management-Prozesses in die Produktion. ST umfasst:

- Change Management
- Projektmanagement (Transition Planning)
- Service Asset & Configuration Management
- Release & Deployment Management
- Service Validation & Testing
- Evaluation
- Knowledge Management

Das Change Management steuert den Lebenszyklus aller Changes. Das vorrangige Ziel des Change Management besteht darin, nutzbringende Changes zu ermöglichen und dabei negative Auswirkungen auf die IT-Services zu vermeiden.

Das Projektmanagement (Transition Planning and Support) plant und koordiniert Ressourcen zum Ausrollen eines Major Release innerhalb des prognostizierten Kosten-, Zeit- und Qualitätsrahmens.

Das Service Asset and Configuration Management stellt Informationen zu Configuration Items (Konfigurationselementen) bereit, die zur Erbringung von IT-Services erforderlich sind, einschließlich ihrer Beziehungen untereinander.

Das Release and Deployment Management plant, legt fest und kontrolliert, wie ein Release getestet und in die Live-Umgebung ausgerollt wird. Das primäre Ziel des Release and Deployment Managements besteht darin sicherzustellen, dass die Integrität der Live-Umgebung geschützt wird und nur zuvor geprüfte Komponenten ausgerollt werden.

Mit Service-Validierung und -Test wird sichergestellt, dass ausgerollte Releases und die daraus resultierenden Services qualitätsgeprüft sind sowie bewertet wird, ob der IT-Betrieb in der Lage ist, den neuen Service angemessen zu unterstützen.

Mit der Anwendungsentwicklung und dem Customizing von Anwendungen und Systemen wird die erforderliche Funktionalität für die IT-Services bereitgestellt. Dieser Prozess umfasst sowohl die Entwicklung und Wartung kundenspezifischer Anwendungen als auch die Anpassung von Standardsoftware sowie deren abschließende Bewertung (Evaluation).

Das Knowledge Management erfasst, analysiert, speichert und stellt Wissen und Informationen innerhalb der IT-Organisation bereit. Der primäre Zweck des Knowledge Managements

besteht darin, Wissen effizient verfügbar zu machen, sodass es nicht mehr notwendig ist, einmal erworbenes Wissen aufwendig wiederzuerlangen.

Service Operation (SO)

Das IT Operations Management überwacht und steuert die IT-Services und die IT-Infrastruktur. Der Prozess IT Operations Management führt die laufenden Routinetätigkeiten aus, die mit dem Betrieb von Infrastrukturkomponenten und Anwendungen verbunden sind. Allgemeine SO-Tätigkeiten, die zu erledigen sind, umfassen den IT-Betrieb (Console, Job Scheduling, Backup- und Wiederherstellungs-Aktivitäten, Print- und Output-Management sowie regelmäßig anfallende Wartungs- und Administrationsarbeiten), Mainframe Support, Server Management and Support, Desktop Support, Middleware Management, Internet/Web Management, Application Management und IT-Security-Aufgaben.

Das IT Facilities Management verwaltet die physikalischen Anlagen und Einrichtungen, in denen die IT-Infrastruktur untergebracht ist. Facilities Management umfasst alle Aspekte der Verwaltung der Anlagen und Einrichtungen, z. B. Stromversorgung und Kühlung, Zugangskontrolle und Umfeldüberwachung.

Funktional und von der Aufbauorganisation gehören zu Service Operation der Service Desk, das Technical Management, das IT Operations Management und das Application Management.

Zu SO gehören folgende Prozesse:

- Event Management
- Incident Management
- Request Fulfilment
- Problem Management
- Access Management

Das Event Management filtert und kategorisiert Events bzw. leitet geeignete Maßnahmen ein.

Das Incident Management verwaltet alle Zwischenfälle über ihren gesamten Lebenszyklus. Das primäre Ziel des Incident Management besteht darin, einen IT-Service für den Anwender so schnell wie möglich wiederherzustellen.

Das Request Fulfilment bearbeitet Service-Anfragen, wobei es sich in den meisten Fällen um geringfügige Changes (Standard Changes, z. B. Anforderung zur Passwort-Änderung) oder Anfragen nach Informationen handelt.

Das Problem Management verwaltet alle Probleme über ihren gesamten Lebenszyklus. Die primären Ziele des Problem-Managements bestehen darin, dem Auftreten von (sich wiederholenden) Zwischenfällen vorzubeugen und die Auswirkungen von Incidents, die nicht verhindert werden können, minimal zu halten. Das proaktive Problem Management analysiert Incident Records und setzt Daten, die in anderen IT-Service-Management-Prozessen erhoben worden sind, dazu ein, Trends oder maßgebliche Probleme zu bestimmen.

Das Access Management bewilligt autorisierten Anwendern das Recht, einen Service zu nutzen, und unterbindet gleichzeitig den Zugriff für unautorisierte Anwender. Der Access-Ma-

nagement-Prozess führt im Wesentlichen Vorgaben aus, die im IT Security Management definiert worden sind. Access Management wird hin und wieder auch als Rights Management (Rechte-Management) oder als Identity Management (Identitäts-Management) bezeichnet.

Kontinuierliche Service-Verbesserung (KSV)

Mit der KSV werden Aktivitäten zur Verbesserung der IT-Services identifiziert und implementiert, die die Business-Prozesse unterstützen. Weiters werden Verbesserungen an den Service-Management-Prozessen identifiziert und umgesetzt. Die Verbesserungsaktivitäten unterstützen den Lifecycle-Ansatz durch alle Phasen (Service Strategy, Service Design, Service Transition und Service Operation) und sollten die drei Aspekte Prozesseffektivität, Prozesseffizienz und Kosteneffektivität einbeziehen.

Typische Aufgaben, die im Rahmen der KSV zu erledigen sind, umfassen:

- Review, Analyse und Erarbeitung von Empfehlungen zur Verbesserung in jeder Phase des Lifecycle
- Review und Analyse der erreichten Service Levels
- Identifikation und Durchführung von Verbesserungsmaßnahmen zur Steigerung der Servicequalität und der Effizienz sowie Effektivität der ITSM-Prozesse
- Verbesserung der Kosteneffektivität in der Service-Erbringung
- Identifikation und Durchführung von Verbesserungsmaßnahmen für die ITSM-Prozesse und die sie unterstützenden Tools
- Sicherstellung, dass geeignete Qualitätssicherungsmethoden für die kontinuierliche Verbesserung genutzt werden

Zur KSV gehören die Service-Evaluierung, die Prozess-Evaluierung, die Definition und Überwachung von Verbesserungsinitiativen.

Die Service-Evaluierung prüft regelmäßig die Servicequalität. Dies schließt das Identifizieren von Bereichen ein, in denen die vereinbarten Service Levels nicht erreicht werden. Außerdem wird regelmäßig das Gespräch mit dem Kunden gesucht, um sicherzustellen, dass die vereinbarten Service Levels noch den Erfordernissen des Kunden entsprechen.

Mit der Prozess-Evaluierung werden die Prozesse des Service-Providers regelmäßig geprüft und bewertet. Dies schließt das Identifizieren von Bereichen ein, in denen die Kennzahlenziele nicht erreicht werden. Außerdem werden regelmäßige Benchmarkings, Audits, Maturity Assessments und Service Reviews durchgeführt.

Mit der Definition von Verbesserungsinitiativen werden konkrete Initiativen zur Verbesserung von Prozessen und Services definiert, ausgehend von den Ergebnissen der Service- und Prozess-Evaluierung. Die sich ergebenden Initiativen sind entweder interne Projekte, die der Service-Provider in eigener Verantwortung verfolgt, oder Initiativen, die eine Zusammenarbeit mit dem Kunden erfordern.

Schließlich wird mit der Überwachung von Verbesserungsinitiativen festgestellt, ob diese wie geplant fortschreiten und ggf. korrigierende Maßnahmen einzuleiten sind.

Service Management gemäß ITIL besteht aus einer Menge von spezialisierten organisatorischen Fertigkeiten (Capabilities), um für den Kunden einen Wert in Form von Services zu er-

bringen. Die Fertigkeiten haben die Form von Funktionen und Prozessen, um Services über den gesamten Lebenszyklus, bestehend aus Service Strategy, Service Design, Service Transition, Service Operation und kontinuierlicher Verbesserung, zu managen.

Service Management nach ITIL ist vielfach noch nicht unternehmensweit implementiert. In den meisten Fällen sind es kleinere Organisationseinheiten, welche Service Management nach ITIL umgesetzt haben. Viele Organisationen haben in den Anfängen ITIL falsch interpretiert; beispielsweise wurde für jeden Prozess eine hierarchische Position geschaffen, um diesen zu managen. Die Philosophie von ITIL besteht unabhängig von der Größe der Organisation im Ausrichten der IT-Dienstleistungen auf die Geschäftsbedürfnisse – nicht mehr – aber auch nicht weniger. ITIL V3 setzt sich nur langsam in der Praxis durch. Viele der Implementierungen sind auf der Basis von ITIL V2.0.

Ein interessantes Konkurrenzmodell zu ITIL V3 ist CMMI-SVC V1.3 des SEI. CMMI for Services ist eine Konstellation (Variante) des CMMI-Baukastens, die folgende Vorteile hat:

- Es benutzt die im CMMI vorhandenen 16 Kernprozesse (dies bedeutet: ca. 77 % der CMMI-DEV-Prozesse & Hilfsmitteln werden wieder verwendet, d. h. Investitionsschutz).

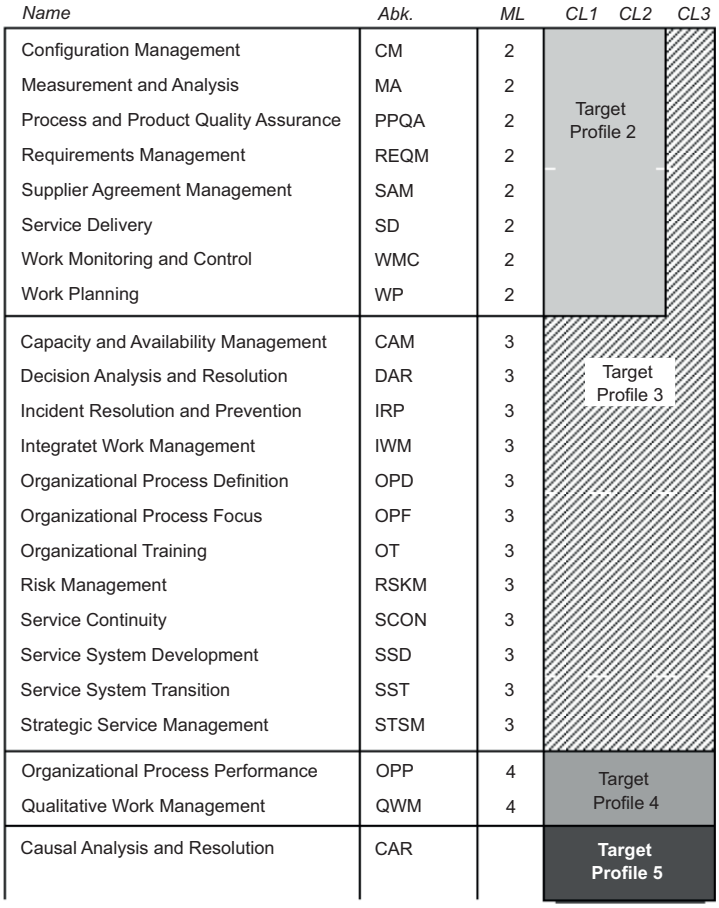


BILD 4.3
CMMI-SVC V1.3

- Es erweitert die Kernprozesse um serviceorientierte Prozesse und Praktiken.
- Es benutzt zur Bewertung und Verbesserung der Reife der Prozesse die Assessmentmethode SCAMPI und schafft dadurch eine international anerkannte Vergleichbarkeit bzw. ein Benchmarking von Service-Organisationen.
- Es benutzt das leicht verständliche und anschauliche Konzept eines Service-Systems.
- Es ist nicht auf IT-Services beschränkt und erlaubt auch beliebige Business Services zu implementieren.

ISO/IEC 20000 ist der internationale zertifizierungsfähige Standard für IT-Service-Management. Diese Norm beschreibt einen integrierten Satz von Managementprozessen für die Lieferung von Dienstleistungen als ganzheitlichen Ansatz zwischen internen und externen Organisationen im Rahmen des IT-Service-Managements. ISO 20000 ist ausgerichtet an den Prozessbeschreibungen von ITIL V2.0 und ergänzt diese.

Die Anforderungen der ISO/IEC, deren Erfüllung bei einer Zertifizierung nachgewiesen werden müssen, umfassen:

- Anforderungen an das Managementsystem
- Planung und Implementierung des Service Managements
- Planen und Implementieren neuer oder geänderter Services
- Service Level Management
- Service Reporting
- Availability & Service Continuity Management
- Finanzplanung und Kostenrechnung für IT-Services
- Capacity Management
- Information Security Management
- Business Relationship Management
- Supplier Management
- Incident Management
- Problem Management
- Configuration Management
- Change Management
- Release Management

ISO 20000 besteht aus 4 Teilen:

- Teil 1 ISO 20000 – „Service Management: Specification“
  Der erste Teil des Standards (ISO 20000-1) enthält die formelle Spezifikation des Standards. Es sind Vorgaben dokumentiert, die eine Organisation einhalten, sicherstellen und nachweisen muss, um eine Zertifizierung zu erhalten. ISO 20000-1 enthält die „Musskriterien“ des Standards.
- Teil 2 ISO 20000 – „Service Management: Code of Practice“
  Innerhalb des zweiten Teils der ISO 20000 werden die Anforderungen des ersten Teils um

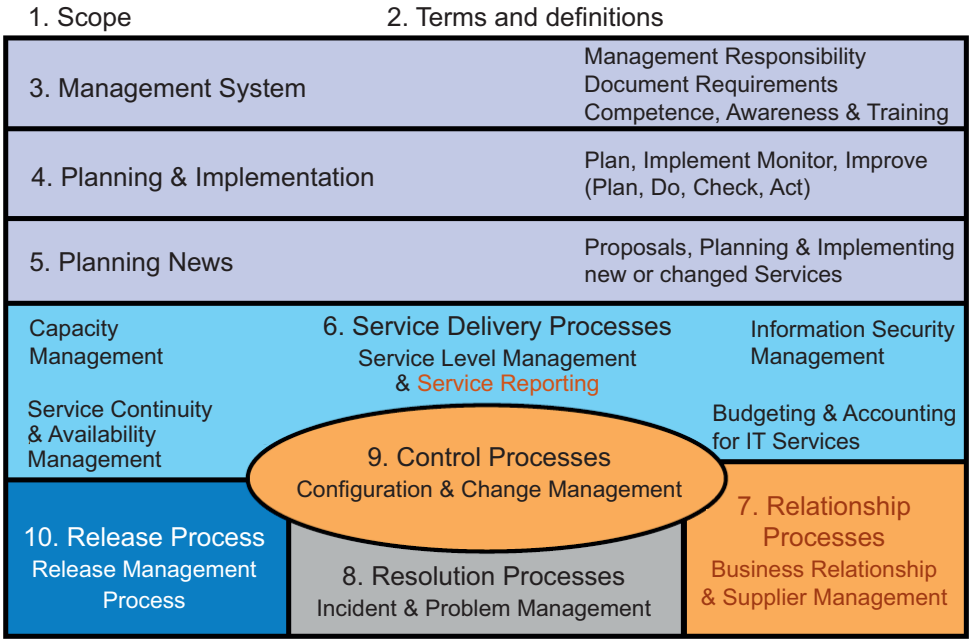


BILD 4.4 ISO/IEC 20000 – Struktur der Service-Management-Prozesse

Erläuterungen der Best Practices ergänzt. ISO 20000-2 bietet Leitlinien und Empfehlungen für IT-Service-Management-Prozesse im Rahmen des formellen Standards.

- Teil 3 BIP 0005 – „IT Service Management – A Managers Guide“
  Als Ergänzung zu ISO 20000 enthält BIP 0005 des BSI (British Standards Institute) eine Managementbeschreibung zur Zielsetzung und zu den Inhalten des IT-Service-Managements auf der Basis von ISO 20000 und ITIL.
- Teil 4 BIP 0015 – „IT Service Management Self-Assessment Workbook“
  Mithilfe von BIP 0015 kann eine Selbstbewertung der bestehenden Prozesse in Relation zu den Vorgaben und den Best Practices der Norm ISO 20000 vorgenommen werden. Hierzu sind zu jedem Prozess entsprechende Fragestellungen beschrieben.

Zertifizierung einer IT-Service-Organisation

Die erfolgreiche Umsetzung von IT-Service-Management in einer IT-Organisation kann durch eine Zertifizierung nach ISO 20000 abgeschlossen werden. Damit besteht die derzeit einzige Möglichkeit, eine erfolgreiche Implementierung des IT-Service-Managements anhand eines internationalen Standards objektiv zu messen und zu auditieren. Eine ISO-20000-Zertifizierung kann auf einen Kunden, einen IT-Service oder einen Standort einer IT-Organisation begrenzt werden.

Durchgeführt werden die Zertifizierungsaudits durch unabhängige Prüforganisationen – die RCBs (Registered Certification Bodies).

Im Verlauf eines Audits werden verschiedene Voraussetzungen überprüft. Es sind dies:

A) Die IT-Organisationen müssen verschiedene Dokumente und Reports vorweisen können bezüglich der Effektivität, Planung, Handhabung und Steuerung von Services:

- Richtlinien und Pläne
- Service Level Agreements
- Prozesse und Prozeduren
- Aktualität der Reports
- Prozeduren und Verantwortlichkeiten für das Management der Dokumentationen

B) Alle Rollen und Verantwortlichkeiten des Service Managements müssen definiert und aktuell bezüglich der geforderten Kompetenzen sein. Mitarbeiterkompetenzen und Trainingsanforderungen müssen überprüft werden. Das Topmanagement muss sicherstellen, dass jeder seine Rolle und Aktivitäten sowie seine Bedeutung bei der Erreichung der Zielsetzung des Service Managements verstanden hat.

Viele Service-Organisationen scheuen den Aufwand einer Implementierung nach ISO 20000 und deren Zertifizierung, da sie nur eine Teilmenge von ITIL-Prozessen umsetzen wollen. Ein weiterer Nachteil scheint das Fehlen eines Reifegrad-Konzepts zu sein, das einen kontinuierlichen Weg der Verbesserung und des Fortschritts für Service-Organisationen und deren Kunden aufzeigen könnte.

Der Übergang von der Entwicklung in den Betrieb von Systemen

Probleme beim Übergang von der Entwicklung in den Betrieb von Systemen sind heute in der Praxis häufig vorzufinden:

- Die Anforderungen des Betriebs werden bei der Produkt-/Applikationsentwicklung zu wenig oder gar nicht berücksichtigt.
- Bei Ende des Applikations-/Software-Projekts wird das Release- und Ausbreitungsmanagement zu wenig berücksichtigt („Die fertige Applikation wird über den Zaun geschmissen“).
- Service-Level-Vereinbarungen (OLA für interne Dienstleistungen, SLA für externe Dienstleistungen) für ein funktionierendes Zwischenfalls-, Problem- und Änderungsmanagement sind ungenügend oder fehlen.
- Die Koordination, wer welche Rolle beim Applikationsmanagement übernimmt (IT-Entwicklung, IT-Betrieb), ist mangelhaft oder gar nicht vorhanden.

Ein Lösungsansatz für die oben genannten Probleme besteht darin, die Schnittstellen zwischen IT-Entwicklung und IT-Betrieb zu klären und zu vereinbaren. Dabei können die Best-Practice-Modelle wie CMMI-DEV, CMMI-SVC und ITIL V3 ausgezeichnete Unterstützung sein.

Qualität und Kundenzufriedenheit durch IT-Service-Management

von Josef Dold

Klassische IT-Organisationen stehen vor immensen Herausforderungen. Sie müssen sich als professionelle Dienstleister mit hoher Kundenorientierung am Markt bewähren. Zur Bewältigung dieser Aufgabe stehen eine Reihe von Regeln und Instrumenten als Hilfsmittel zur Verfügung.

„Eine einfache Erfolgsregel ist, dem Kunden immer mehr zu geben, als er erwartet."
James Boswell (1740–1795), schottischer Biograph

Dem Kunden mehr geben, bedeutet hier, die vielfältigen Erwartungen des Kunden zu übertreffen (siehe auch **Kano-Modell**). Dabei muss der Dienstleister die Wirtschaftlichkeit seines Handelns sicherstellen, damit er als Unternehmung bestehen kann. Was bedeutet diese Einleitung für das IT-Service-Management und das Qualitätsverständnis? Zunächst eine Definition der Begriffe:

IT-Services nach ITIL® ...
... unterstützen die Geschäftsprozesse des Kunden. **Beispiel:** „Internet-Banking" ist ein Business-Service (Produkt) der Bank X mit einer Reihe von Geschäftsprozessen; z. B. „Giro", „Sparen", „Anlegen". IT-Services, z. B. „managed Web-Server" unterstützen diese Geschäftsprozesse. Die IT-Services müssen dabei dem Kunden einen Nutzen (Wertschöpfung, Mehrwert) liefern, indem die erwarteten Ergebnisse produziert werden, ohne dass der Kunde die spezifischen Risiken und Kosten trägt. Das Maß der Geschäftsprozessunterstützung durch IT-Services ist dabei der Wertbeitrag zum Unternehmenserfolg.

IT-Service-Management ...
... heißt, die Gesamtheit der Fähigkeiten so zu koordinieren, dass IT-Services für den Kunden einen kosteneffizienten, stetigen und steigenden Mehrwert darstellen. Der Nutzenaspekt muss im Vordergrund stehen, und es geht immer darum, die IT-Services zu verbessern und Kundenzufriedenheit zu steigern.

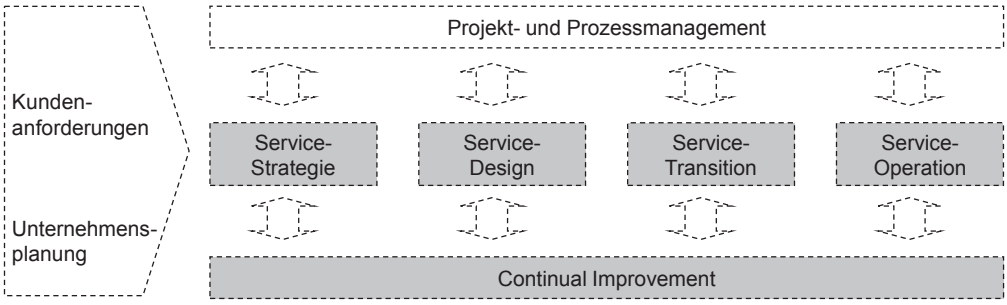


BILD 4.5 Erweiterte Grundaufgaben im IT-Service-Management nach ITIL® V3

Qualität im IT-Service-Management (ITSM)
Qualität im ITSM kann durch die Einbeziehung von Normen erreicht werden. Das ITSM braucht zusätzliche Regeln, Standards und Qualitätsziele, z. B. für eine unternehmenswei-

te, einheitliche Prozessdokumentation. IT-Service-Management korreliert bestenfalls mit einem unternehmensweiten QM-System, in dem die Normen ISO/IEC 20000 und ISO/IEC 27001 umgesetzt sind. Wichtig ist, dass alle Aktivitäten stets auf den Kunden ausgerichtet sind. ITSM hat das primäre Ziel, Kundenzufriedenheit unter Kosteneffizienz herzustellen und Mehrwert zu schaffen; daran orientiert sich das Qualitätsverständnis im ITSM. Dies bedeutet konkret: Qualität eines IT-Service = Erfüllungsgrad der Erwartungen eines Kunden. Bezüglich der Wahrnehmung des Kunden stehen hier folgende Perspektiven und Fragen im Mittelpunkt:

- Entspricht der IT-Service inhaltlich den **Anforderungen** und **Erwartungen**?
- Ist in der Zukunft eine gleichbleibende oder bessere **Service-Leistung** zu erwarten?
- Wird der IT-Service zu vertretbaren **Kosten** erbracht?

Die Gesamtqualität des IT-Service-Managements und der sich aus Prozessen bildenden Servicekette von der **Service Strategy** über das **Service Improvement** bis hin zur Wertschöpfung beim Kunden bilden den Umfang dieser Betrachtung. Es ist zu beachten, dass die Gesamtqualität des IT-Service-Managements keine einzelne physikalische Größe darstellt und somit schwer zu messen ist. Relativ gut messbar ist der Erfüllungsgrad einzelner Forderungen, z. B. eine bestimmte Lösungszeit bei Störungen (Incidents) und Instrumente beim Dienstleister, z. B. die Prozessqualität zur Produktion von IT-Services mittels Reifegradmodellen und der Norm ISO/IEC 20000. Es ist zwingend notwendig, mehrere Perspektiven zu betrachten, um eine Qualitätsbeurteilung von IT-Services vornehmen zu können. Es gilt:

> Gesamtqualität = Kundenzufriedenheit = Σ der Erfüllungsgrade einzelner Qualitätsperspektiven

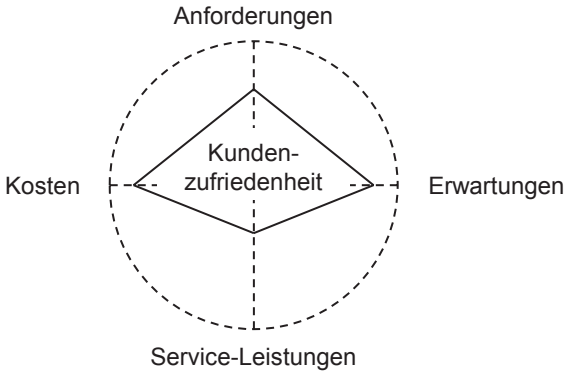


BILD 4.6 Perspektiven der Kundenzufriedenheit

Perspektive „Anforderungen“

Hier werden die konkreten Leistungsmerkmale (siehe Kano-Modell) der IT-Services gefordert. IT-Services müssen für den Kunden einen Mehrwert generieren. Sie definieren sich daher über den Nutzen für den Kunden und nicht über zur Verfügung gestellte Produkte oder Kapazitäten. Dabei können IT-Services nicht auf Lager produziert werden, sondern entstehen im Zeitpunkt des Bedarfs („just in time“) und müssen bei jedem Bedarf erneut erzeugt werden. Dies bedeutet, dass von einem IT-Service nie die auszuliefernde Version,

sondern immer nur ein Muster geplant, entwickelt, getestet und bereitgestellt werden kann. Aus dieser Darstellung wird klar, dass für IT-Services zwei Kategorien von Anforderungen betrachtet werden müssen:

- Was leistet der IT-Service („fit for purpose“)?
- Wie sicher ist die Lieferung („fit for use“)?

Beide Anforderungen an IT-Services müssen bei jeder Produktion im IT-Betrieb die Anforderungen des Kunden erfüllen. Diese Tatsachen verdeutlichen die enorme Wichtigkeit, die inhaltlichen Anforderungen (Leistungsmerkmale) der Kunden wirklich gut zu kennen, zu verstehen und umzusetzen. Sind die Inhalte nicht wie erwartet umgesetzt, sind meistens auch die Anforderungen an die Lieferung nicht zu erfüllen, besonders dann, wenn die Lieferungen in hoher Frequenz, z. B. 24 x 7 x 365, erfolgen müssen. Zur Unterstützung von Lösungen beschreibt ITIL® im Buch **Service Strategy** Aktivitäten (**„develop the offerings“)** und Prozesse **(Service Portfolio Management)**. Sie helfen bei der Aufgabe, das Dienstleistungsangebot vorausschauend den Bedürfnissen des Kunden anzupassen. Das Portfolio Management spiegelt dessen Bedarf und die damit verbundene, notwendige Reaktion des Dienstleisters wider. Den Mehrwert und somit letztlich die Qualität eines IT-Services kann dann immer nur der Kunde beurteilen.

Perspektive „Erwartungen“

Erwartungen gehen über die Anforderungen an Leistungsmerkmale (siehe Kano-Modell in diesem Abschnitt) hinaus. Wichtig ist, dass sie miteinander im Einklang stehen, d. h., die Erwartungen spiegeln auch die spezifischen Anforderungen. Ob der IT-Service den Erwartungen des Kunden umfänglich entspricht, hängt vor allem davon ab, wie genau zuvor die Abstimmung mit dem Kunden durchgeführt wurde. Dafür ist es notwendig, dass Kunde und Dienstleister das gleiche Verständnis von dem zu liefernden IT-Service haben. Hilfe für diese Abstimmung liefert ITIL® mit dem **Service Design**, speziell dem **Service Catalogue Management**. Das Vorgehen der Abstimmung erfolgt nach dem Prozess **Service Level Management**, die Ergebnisse werden in **Service Level Agreements** dokumentiert. Ein weiteres Hilfsmittel ist das **Kano-Modell** (Dr. Noriaki Kano, Professor an der Universität Tokio, 1978), das sich mit dem Einfluss von Leistungsmerkmalen auf die Kundenzufriedenheit befasst. Die Anwendung des Kano-Modells geschieht in der Planungsphase und dient dazu, mittels **Customer-Value-Analyse** optimale IT-Services zu entwickeln. Kano unterscheidet dabei 5 Qualitätsebenen:

- Basismerkmale – Kunde spricht nicht darüber, hat implizite Erwartung
- Leistungsmerkmale – Erwartung an die Kompetenz des Dienstleisters
- Begeisterungsmerkmale – Erwartung an Alleinstellungsmerkmale und Wettbewerbsvorteile
- Unerhebliche Merkmale – interessiert den Kunden gar nicht
- Rückweisungsmerkmale – Kunde möchte diese Merkmale nicht; ggf. spricht er nicht darüber

Es ist wichtig, diese Merkmale für jeden IT-Service und jeden Kunden zu kennen. Nur so kann der Dienstleister „mehr geben“ als erwartet, indem er in allen Phasen des Service-Lifecycle die richtigen Merkmale kennt und entsprechende Aktivitäten durchführt.

Perspektive „Service-Leistung“

Als „Service-Leistung“ werden hier die Planung, Implementierung und der Betrieb von IT-Services betrachtet. Denn die beste Service- und Kundenstrategie ist nicht realisierbar, wenn die IT-Services nicht nachhaltig mit hoher Qualität entwickelt und produziert werden. Zur Unterstützung stellt ITIL® dafür die Bücher **Service Design** (IT-Services planen, Zulieferungen verhandeln und Leistungen absichern), **Service Transition** (IT-Services gesichert verwalten und dem IT-Betrieb zuverlässig zur Verfügung stellen) und **Service Operation** (IT-Services den Kunden zum richtigen Zeitpunkt am richtigen Ort zur Erfüllung der Anforderungen bereitstellen) zur Verfügung. Das Framework beschreibt Referenzprozesse, an denen die Produktionsprozesse für IT-Services ausgerichtet werden können. Im Folgenden die wichtigsten Stellhebel, um IT-Services qualitativ hochwertig bereitzustellen:

Standardisierung erhöhen

Da Mehrwert zu schaffen für den Kunden kein Zufall sein darf, ist der Kunde Teil der Leistungserstellung und gleichzeitig Servicenehmer. Je stärker der Kunde aber eingebunden ist, desto schwieriger wird es, einen hohen allgemeinen Standardisierungsgrad umzusetzen. Die Individualität der IT-Services wird größer; oft werden spezielle Kundenwünsche erfüllt. Die Sicherstellung klassischer Qualitätsmerkmale wie Verfügbarkeit, Zuverlässigkeit und Wartbarkeit wird mangels Standardisierung aufwendiger und schwieriger. Der Kunde muss diese individuelle Leistung bezahlen. Kundenindividuelle IT-Services können nicht zum Preis einer standardisierten Massenproduktion geliefert werden. Versucht man es trotzdem, sinkt entweder die Qualität, oder der Dienstleister gerät in wirtschaftliche Schwierigkeiten.

Qualität der Zulieferer (Lieferanten) steuern

IT-Dienstleister brauchen eine Service-Strategie, die in der Zukunft eine höhere Leistung – quantitativ und qualitativ – zu geringeren Kosten ermöglicht. Zulieferungen über Dritte können dazu einen wesentlichen Beitrag leisten, wenn es gelingt, die Produktivität zu steigern und mit schlanken Ansätzen zielorientiert mit Lieferanten zusammenzuarbeiten. IT-Dienstleister müssen sich Gedanken über ihr grundlegendes Geschäftsmodell und ihr Basisportfolio an IT-Services machen. Wenn sie wettbewerbsfähig bleiben wollen, sollten sie sich auf das Anbieten von IT-Services beschränken, die in das Gebiet ihrer Kernkompetenzen fallen. Andere Services lassen sich besser von Dritten zukaufen. Bei der Umsetzung dieser Lösungsansätze kann der ITIL®-Referenzprozess **Supplier Management** helfen. Zentrale Aufgabe ist dabei die Definition von Rollen und Verantwortungen, abgestimmt auf die Servicevereinbarungen (SLA) mit den Kunden, sowohl auf Seiten des IT-Dienstleisters als auch auf Seiten der Zulieferer.

Sicherheit und Risiken

Weil die Geschäftsprozesse in immer größere Abhängigkeit von der IT stehen, ist eine Planung für den IT-Grundschutz und den Katastrophenfall unerlässlich. Die **Norm ISO/IEC 27001** spezifiziert die Anforderungen für Herstellung, Einführung, Betrieb, Überwachung, Wartung und Verbesserung eines Managementsystems. Sie wurde entworfen, um die Auswahl geeigneter Sicherheitsmechanismen zum Schutz sämtlicher Werte (Assets) in der Wertschöpfungskette sicherzustellen. Weitere Hilfe bietet ITIL® mit dem Prozess **IT Service Continuity Management.** Er beschreibt, was ein Unternehmen tun sollte, um im Katastrophenfall die wesentlichen IT-Services planvoll wiederherzustellen. Jedoch ist mittels Risikoanalysen zu verdeutlichen, wie sich die zu erwartenden Kosten für Maßnahmen und

die daraus resultierenden Vorteile verhalten. Risiken müssen angemessen, d. h. anhand ihrer Auswirkungen und Eintrittswahrscheinlichkeit betrachtet werden. Erst wenn bekannt ist, worin das Risiko für den Kunden und den IT-Dienstleister besteht, sollte in Maßnahmen investiert werden. Andernfalls drohen gewaltige Kosten, die der Kunde nicht akzeptiert, und Unzufriedenheit auf Seiten des Kunden.

Service-Leistung steigern
„Dem Kunden immer mehr geben, als er erwartet!" Wo sind die Stellhebel, um dies zu erreichen? Die Basis- und Leistungsmerkmale (siehe Kano-Modell) eines IT-Service reichen nicht aus. Der Kunde erwartet diese normalerweise auch vertraglich festgelegten Leistungen. Die Potenziale liegen bei den Leistungen mit Begeisterungsmerkmalen. Potenzielle Störungen frühzeitig entdecken und Maßnahmen vornehmen, bevor die Störungen auftreten, wäre eine solche Leistung, besonders in Verbindung mit einer Kostensenkung, denn Störungen verhindern ist kostengünstiger als Störungen beheben. Eine Reihe von Instrumenten stehen diesbezüglich zur Verfügung. ITIL® hilft hier mit den Prozessen **Availability Management** und **Continual Service Improvement**. Das **Availability Management** beschreibt das Vorgehen für die Erbringung qualitativ hochwertiger IT-Services und die kosteneffektive Bereitstellung des in den **Service Level Agreements** festgelegten Verfügbarkeitsniveaus. Dabei gilt es, die richtige Balance zwischen den aufzuwendenden Kosten und dem für das Business sinnvollen Verfügbarkeitsniveau zu erreichen.

Der Prozess **Continual Service Improvement** hat einen übergeordneten Charakter. Er beschreibt Arbeitsschritte zur Verbesserung der IT-Services und des Service Managements zur Maximierung der Wertschöpfung für den Kunden. Ziel ist eine kontinuierliche Anpassung und Neuausrichtung der IT-Services an die aktuellen und künftigen Kundenanforderungen. Dies geschieht durch die Analyse der erreichten Service Levels in Verbindung mit den KPIs (Key-Perfomance-Indikatoren) – was voraussetzt, dass jeder Prozess mit solchen KPIs ausgestattet ist. Des Weiteren werden Empfehlungen für Verbesserungen der Servicequalität für jede Phase des Service-Lifecycle erarbeitet. Der diesem Verfahren zugrunde liegende Modellansatz (PDCA = Plan, Do, Check, Act) ist auch als **Deming Cycle** bekannt (W. Edwards Deming (1903–1993), US-Wissenschaftler, führender Kopf der Qualitätsbewegung).

Weitere Instrumente wie die Norm **ISO/IEC 20000, CMMI for IT-Services oder Six Sigma** helfen bei der Ermittlung von Schwachstellen und Qualitätsbeurteilungen von Prozessen. Transparenz und Messbarkeit einzelner Prozessmerkmale werden dabei verbessert und somit die Steuer- und Standardisierbarkeit optimiert. Eine Zertifizierung nach ISO/IEC 20000 hat zudem den Vorteil, dass ein nachgewiesenes Qualitätsniveau über alle IT-Service-Managementprozesse belegbar und für vertrauensbildende Kundenmaßnahmen geeignet ist. Der ISO-Standard unterstützt nachhaltig das Ziel, IT-Qualität zu steigern bei gleichzeitiger Kostensenkung.

Perspektive „Kosten"
Ein Kunde im IT-Service-Management bezieht kein Einmalprodukt, sondern eine immer wiederkehrende Leistung (IT-Services) über einen längeren Zeitraum. Kunde und Dienstleister vereinbaren ggf. eine langjährige Partnerschaft. Der Kunde möchte geringe Kosten – nachvollziehbare Kosten bzw. Kostentransparenz, damit er vergleichen kann. Der Prozess **Service Level Management** nach ITIL® hilft, diese Service-Beziehung zu regeln, indem

Service Level Agreements (SLA) präzise abgestimmt und festgeschrieben werden. Zwingender Inhalt dieser Vereinbarung ist ein Anreizsystem (Bonus-Malus-Regelung). Die Anreize müssen so gestaltet sein, dass sie zu Qualitätsverbesserungen und Kostensenkungen für den Kunden führen. Ferner ist ein verständliches, auf die Zielgruppe (z. B. Anwender- oder Management-Report) abgestimmtes Reporting notwendig; dies schafft Transparenz und Vertrauen. Der ITIL®-Prozess **Financial Management** sorgt dafür, dass ein entsprechendes Zahlenmaterial bereitgestellt wird. Er schafft finanzielle Transparenz hinsichtlich der Kosten bzw. des Budgets und liefert Kennzahlen zur Unterstützung von Unternehmensentscheidungen.

Einen wesentlichen Beitrag zur Kostenoptimierung kann der ITIL®-Prozess **Demand Management** liefern. Er sorgt dafür, dass Kapazität nach den Bedürfnissen bereitgestellt wird, d. h. Kapazität zum richtigen Zeitpunkt am richtigen Ort ohne Mehrkosten verfügbar ist und Kosten infolge überschüssiger, nicht genutzter Kapazität verhindert werden. Verschiedene Techniken im Demand Management, wie Off-Peak Pricing, Volume, Discounts oder gestaffelte Service Levels, können den Bedarf im Rahmen von spezifischen Modellen beeinflussen. Kapazitäten und Ressourcen, die einem Service zur Verfügung gestellt werden, sollten an Bedarfsprognosen ausgerichtet sein.

Fazit und Trends

Produktivität steigern, Wertbeitrag (Kundennutzen) erhöhen, Kosten senken und Kundenzufriedenheit optimieren! Dies sind existenzielle Herausforderungen, auf die IT-Organisationen künftig mit Lösungen antworten müssen. IT-Service-Management ist dabei ein unverzichtbarer Bestandteil solcher Lösungen. Um die genannten Herausforderungen zu bewältigen, ist vor allem eine gute Service-Strategie notwendig; es muss z. B. klar sein, ob „Massenfertigung“ oder „Maßschneiderei“ betrieben wird. Ein unkontrollierter Mix führt zu Unzufriedenheit, da Kunden unterschiedliche Erwartungshaltungen haben. Man denke an den gewaltigen Unterschied beim Kauf eines Anzuges von der Stange im Mega-Store gegenüber dem maßgeschneiderten Anzug beim Schneider; dies gilt auch für IT-Services.

IT-Services mit einem hohen Wertbeitrag und geringen Kosten lassen sich nur dann wirtschaftlich produzieren, wenn ein hoher Grad an Standardisierung, Modularisierung und Automatisierung gegeben ist; dies bedeutet Massenfertigung bzw. Industrialisierung.

Erwartet der Kunde individuelle IT-Services mit einem hohen Wertbeitrag bedeutet dies „Maßschneiderei“. Er muss bereit sein, die höheren Kosten dafür zu tragen.

Sind dem Kunden diese Sachverhalte bewusst und transparent, dankt er es dem Dienstleister mit hoher Kundenzufriedenheit, denn seine Erwartungen werden erfüllt.

Literatur

ITIL V3

Bd. 1: Sharon Taylor, Majid Iqbal, Michael Nieves: Service Strategy.
ISBN 978-0-11-331045-6
Bd. 2: Sharon Taylor, Vernon Lloyd, Colin Rudd: Service Design. ISBN 978-0-11-331047-0
Bd. 3: Sharon Taylor, Shirley Lacy, Ivor MacFarlane: Service Transition.
ISBN 978-0-11-331048-7
Bd. 4: Sharon Taylor, David Cannon, David Wheeldon: Service Operation.
ISBN 978-0-11-331046-3

Bd. 5: Sharon Taylor, Gary Case, George Spalding: Continual Service Improvement. ISBN 978-0-11-331049-4

Alan Calder: Information Security based on ISO 27001/ISO 27002, A Management Guide, Van Haren Publishing; Second edition, 2009

Nadin Ebel: ITIL® V3 – Basis Zertifizierung – Grundlagenwissen und Zertifizierungsvorbereitung für die ITIL Foundation-Prüfung, Addison-Wesley, 2008

Elmar Sauerwein: Das Kano-Modell der Kundenzufriedenheit; Deutscher Universitäts-Verlag, 2000

Josef Dold, www.itecdo.com, ist Experte für Prozess- und IT-Service-Management. In zahlreichen Projekten, Vorträgen und Beratungsmandaten hat er mit seinen visionären, aber immer auch praktischen Lösungsansätzen Mehrwert geschaffen. Seine Leidenschaft gilt der Entwicklung einer nachhaltigen IT-Ökonomie.

4.3 Qualitätsmanagement in der Wartung und Pflege von Produkten und Systemen

Wartungsaktivitäten überwiegen heute in vielen Firmen die Entwicklungsaktivitäten. Somit wird die Realisierung neuer Anwendungen und Systeme oft zurückgestellt („Application Backlog"). Gleichzeitig ist dies ein Zeichen, dass nach 50 Jahren „gelebter" Informatik die Verbreitung von Informationssystemen und Software-Produkten ein Ausmaß erreicht hat, bei dem die Pflege und Erweiterung bestehender Systeme (z. B. über neue Releases) im Vordergrund stehen bzw. in der Regel 80 bis 95 % des Budgets einer Organisation ausmachen (gemäß den Erfahrungen des Autors in den letzten 25 Jahren).

Das Wort „Wartung" bedeutet in der Umgangssprache die Wiederherstellung eines befriedigenden Zustands einer Maschine oder eines Geräts. Meist wird Wartung durch materielle Abnützung oder durch Alterung verursacht. Bei Software-Systemen gibt es keine materielle Abnützung oder Alterung, die zu einer Wartung führt. Software-Wartung bedeutet die Verbesserung des Originalzustands. Die Anforderungen an jemanden, der Software-Wartung betreibt, sind hoch. Einerseits muss er ein ausreichendes Fachwissen über die Applikationen bzw. das Software-Produkt besitzen, um die Software bzw. die Anforderungen an diese zu verstehen. Andererseits muss er genügend Software-Engineering-Kenntnisse besitzen, um die Anforderungen realisieren zu können. In der Software-Technik bezeichnet der Begriff Software-Wartung „die Veränderung eines Software-Produkts nach dessen Auslieferung, um Fehler zu beheben, Performance oder andere Attribute zu verbessern oder Anpassungen an die veränderte Umgebung vorzunehmen" (Definition gemäß IEEE 610.12-1990).

Im weiteren Sinne darf man auch Dienstleistungen und Maßnahmen, die die von der Norm beschriebenen Veränderungen begleiten oder unterstützen, zur Software-Wartung rechnen. Die Software-Wartung dient in der Regel dazu, die Verwendbarkeit und Betriebssicherheit von

Software zu erhalten. Software-Wartung ist ein unterschätzter Erfolgsfaktor für die im Markt wahrgenommene Qualität der Produkte.

Die Situation in der Wartung bereits eingeführter Software lässt sich folgendermaßen umschreiben:

- Fertige Software läuft seit den 60er- bzw. 70er-Jahren des vorigen Jahrhunderts.
- Benutzer wollen regelmäßig viele Erweiterungen ihres Systems.
- Programme enthalten oft viele versteckte Fehler.
- Änderungen müssen oft sehr schnell durchgeführt werden.
- Wissen über die Software (Aufbau und Funktionalität) ist nur implizit, wenn überhaupt vorhanden.
- Software wird nicht mehr (komplett) verstanden.
- Unzulänglichkeiten in der Software können nicht einfach behoben werden, die Benutzer müssen sich teilweise mit ihnen arrangieren.

Man unterscheidet zwischen korrektiver, perfektionierender und adaptiver Wartung:

- *Korrektive Wartung* bedeutet die Beseitigung von Fehlern.
- *Perfektionierende Wartung* bedeutet die Verbesserung von Attributen wie etwa der Performance oder der Wartbarkeit. Darunter fällt insbesondere die Bereinigung des Entwurfs oder der Implementierung durch Reengineering (Software), Refactoring usw.
- *Adaptive Wartung* bedeutet die Anpassung der Software an veränderte technische Bedingungen der Umgebung (vgl. IEEE 610.12-1990 und ISO/IEC 12207).

Bei der Beseitigung von Fehlern differenzieren manche Quellen noch zwischen korrektiver und präventiver Wartung, wobei unter *präventiver Wartung* dann die Behebung von solchen Fehlern verstanden wird, die bekannt, aber beim Anwender noch nicht in Erscheinung getreten sind (vgl. IEEE 610.12-1990).

Unter konstruktive Wartung fallen Änderungen, die einen bestimmten Aspekt einer Software verbessern (perfektionieren), ohne die Funktionalität der Software zu verändern. Hierzu gehört z. B. der Einbau eines schnelleren Sortieralgorithmus. Oftmals werden auch Erweiterungen zur konstruktiven Wartung gezählt. Wir wollen Erweiterungen als selbstständige Arbeiten außerhalb der Wartung betrachten. Die adaptive Wartung steht für Änderungen, die eine Software an eine neue Umgebung anpassen. Die Anschaffung neuer Hardware oder eines neuen Betriebssystems ändert die Umgebung einer Software. Die Korrektur bisher nicht entdeckter Fehler, d. h. wenn eine Software ihre Spezifikation nicht erfüllt, fällt unter die korrektive Wartung. Unter präventiver Wartung werden Modifikationen einer Software verstanden, um spätere Wartungsarbeiten zu erleichtern. Diese Wartungsart spielt in der Praxis de facto keine Rolle.

Erweiterungen nehmen einen relativ großen Teil der Änderungsarbeit an eingeführter Software ein. Ein Grund hierfür ist die häufig vorzufindende enge Ressourcenplanung (Zeit, Geld, Personal). Vielfach können nur Minimalsysteme ausgeliefert werden, die nachträglich noch erheblich erweitert werden. Ein weiterer Grund ist, dass allein das Vorhandensein einer Software beim Benutzer wieder neue Wünsche hervorruft. Hinter Erweiterungen verbirgt sich

also häufig eine umfangreiche Menge von Neuentwicklungsarbeit, die unter dem Mantel der Wartung versteckt wird. Durch eine geeignete Release- oder Entwicklungsstufenplanung kann diese versteckte Neuentwicklungsarbeit vermieden werden.

Die Praxis zeigt, dass die Wartung umfangreiche Ressourcen einer Informatik-Organisation in Anspruch nimmt und ernsthafte Probleme verursacht. Die Gründe:

- Für Alt-Software ist keine Dokumentation vorhanden. Der Entwicklungsprozess ist in der Regel nicht mehr nachvollziehbar. Durchgeführte Änderungen wurden nicht dokumentiert.
- Existierende Spezifikationen sind nicht mit dem Programmcode konsistent.
- Alt-Software wurde nicht nach Prinzipien des Software Engineering („structured programming“) entwickelt.
- Die ursprünglichen Entwickler sind nicht mehr greifbar.
- Für die Wartung und Erweiterung von Software sind gute Leute erforderlich. Die guten Leute sind aber meistens mit Neuentwicklungen beschäftigt.
- Alt-Software ist in einer veralteten Programmiersprache geschrieben.
- Änderungen reduzieren die Stabilität des Produkts.
- Änderungen wirken sich in Form eines Domino-Effekts (ripple effect) auf verschiedene Produktteile aus und beeinträchtigen das Produktverhalten.
- Die Produktentwicklung wurde ohne Berücksichtigung der späteren Wartung geplant und durchgeführt.
- Mangelnde Planung, insbesondere das Fehlen eines Release-Plans bei der Produktentwicklung, führen zu schwer wartbarer Software.
- Fehlende Verfolgbarkeit der Anforderungen und Entwurfsentscheidungen auf Code-Ebene erschweren die Modifikation der Software.

Die Wartung wird auch durch die Größe eines Produkts beeinflusst. Das Beispiel SAP R/3 mit ca. 7 Mio. Zeilen Quellcode und ca. 100000 Funktionsaufrufen zeigt die Problematik der Wartung heutiger Systeme deutlich auf. Durch die zunehmende Größe der Systeme wird die Lokalisierung der Änderungen, aber auch die Abschätzung der Auswirkungen von Änderungen erschwert bzw. verunmöglicht. Große Systeme mit besonders erschwerter Wartungssituation sind beispielsweise Flugreservierungssysteme, hochintegrierte Banksysteme, elektronische Vermittlungssysteme, Betriebssysteme, Flugverkehrskontroll-/leitsysteme oder militärische Kommando- und Leitsysteme.

Begriffe und Grundlagen

Software-Wartung ist der Prozess der Veränderung existierender Software und deren Dokumentation auf Grund eines Problems, des Bedarfs einer Verbesserung oder einer Adaption (ISO 12207).

Der Begriff Software-Evolution bringt das permanente Ändern existierender Software besser zum Ausdruck als der nicht sehr aussagekräftige Begriff Wartung. Die Hauptaufgaben der Software-Evolution sind:

- Verstehen einer existierenden Software und der Anforderungen, um sie zu ändern, zu erweitern oder zu korrigieren;
- Modifizieren einer existierenden Software;
- Bewertung der Korrektheit von modifizierter Software durch Prüfungen (Reviews) und Tests.

Lehman und Belady [Lehm80] stellten durch Analysen von großen Programmsystemen der IBM folgende Phänomene im Zusammenhang mit Software-Evolution fest:

- Phänomen des ständigen Änderns
  Ein Programm, das während seines Einsatzes nicht laufend geändert wird, verliert an Wert. Der Änderungs- bzw. der Verfallsprozess geht so lange weiter, bis es kostengünstiger ist, das Programm neu zu schreiben.
- Phänomen der wachsenden Komplexität (Ungeordnetheit)
  Die Komplexität (Ungeordnetheit) eines Programms, verursacht durch die große Anzahl realisierter Funktionen, Benutzerschnittstellen und Systemschnittstellen, wird laufend größer, es sei denn, man investiert Arbeit, um sie zu reduzieren.

Zum ersten Phänomen muss gesagt werden, dass nicht nur Fehler behoben werden, sondern dass laufend Verbesserungen und Anpassungen wegen der sich verändernden Umgebungs- und Benutzeranforderungen notwendig sind. Im Zusammenhang mit der Wartung des Betriebssystems OS/360 stellte man beispielsweise fest, dass immer mehr Module geändert werden mussten, um Seiteneffekte zu vermeiden.

Das zweite Phänomen ist für viele Wartungsverantwortliche eines der schwierigsten Probleme. Aus Zeit- und Kostengründen wird der Verlust an Struktur nicht verhindert. Stattdessen werden schnelle Änderungen („quick fixes“) durchgeführt und weitere Module dem System hinzugefügt, ohne deren Auswirkungen auf bestehende Module zu prüfen. Vielfach sind die Autoren der ursprünglichen Programme nicht mehr verfügbar, und die Wartungsaktivitäten werden oft durch unerfahrene Produktbetreuer durchgeführt. Durch solch einen Änderungsprozess geht die Integrität des Produkts verloren. Am schnellsten merkt man dies an der nicht mehr aktuellen Programmdokumentation. Weiters nimmt die Modulkopplung zu und die Modularisierung ab.

Aus Sicht der Qualitätsplanung stellt sich die Frage, welche Eigenschaften und Merkmale ein Produkt aufweisen muss, um es als leicht wartbar oder wartungsfreundlich bezeichnen zu können. Im Rahmen der Diskussion um Qualitätsmodelle sind wir bereits auf Beschreibungsansätze für Wartbarkeit eingegangen (siehe Kapitel 2 Grundlagen).

Das Informatik-Management wird durch zunehmende Wartungsprobleme und Wartungsanforderungen bei der Planung und Realisierung neuer Anwendungen behindert und eingeschränkt. Bessere Entwurfs- und Implementierungsmethoden führen zwar zu einer höheren Qualität bei neuen Applikationen, die Qualität älterer Systeme, an denen zahlreiche Änderungen durchgeführt wurden, bleibt jedoch schlecht. Aus wirtschaftlichen Gründen ist es nicht möglich, alle alten Produkte zu ersetzen. Es muss daher ein Weg gefunden werden, diesen oben besprochenen evolutionären Änderungsprozess sowohl aus wirtschaftlicher als auch aus qualitativer Sicht plan- und kontrollierbar zu machen. Wir schlagen dazu Maßnahmen vor, um

- die Wartungskosten zu reduzieren,
- die Wartbarkeit zu sichern und
- die Wartungsaktivitäten zu regeln.

4.3.1 Reduzierung der Wartungskosten

Wie Erfahrungen bei der Wartung großer Software-Produkte (z. B. eines Betriebssystems) gezeigt haben, sind Software-Fehler nicht gleich über alle Module verteilt, sondern treten gehäuft in Modulen auf, die oft geändert werden. Andere Erfahrungen zeigen, dass oft geänderte Module weitere Wartungsaktivitäten nach sich ziehen. Wie bereits erwähnt, verschlechtert Wartung die Qualität der Programme hinsichtlich der Modularität, der Strukturierung und der Lesbarkeit. Es stellt sich die Frage, wie diese änderungsträchtigen Module identifiziert werden können. Durch nachfolgende organisatorische Maßnahmen lässt sich das Problem lösen.

Das Wartungspersonal zeichnet genau auf:

- welche Module geändert wurden;
- welcher Aufwand für die Wartungsaktivitäten verbraucht wurde;
- den Grund, warum gewartet wurde;
- eine kurze Beschreibung der durchgeführten Wartungsarbeit.

Der Aufwand wird dabei nicht in Personenstunden, sondern durch die Anzahl der geänderten Zeilen (Quellcode und Dokumentation) angegeben.

Mit diesen Informationen lassen sich Aussagen treffen, welche Module welchen Wartungsaufwand verursachen. Es sind daher auch jene Module feststellbar, die zu sehr hohen Wartungskosten führen. Ab einer bestimmten Anzahl von Änderungen ist es billiger, diese Module neu zu entwickeln.

Zur Verdeutlichung der Problematik geben wir folgende Erfahrungswerte von Boehm an: Schlecht wartbare Produkte reduzieren die Produktivität der Mitarbeiter, die mit Wartungsaufgaben betraut sind, gegenüber jenen, die Neuentwicklungen durchführen, im Verhältnis von bis zu 40:1. Kostet die Entwicklung einer Zeile Code 25 $, so können bei schlecht wartbaren Produkten Wartungskosten bis zu 1000 $ je Zeile Code entstehen [Boeh79a].

Die Angaben, warum gewartet und welche Wartungsarbeit durchgeführt wurde, helfen bei der Entscheidung, ob ein Modul weiter gewartet oder neu programmiert wird. Wir empfehlen Folgendes:

- Vernichte und entwickle das Modul neu, wenn es klein ist.
- Restauriere das Modul, wenn es groß, sehr trickreich programmiert oder stark mit anderen Modulen gekoppelt ist.

In [Schä84] wird berichtet, dass sich mit den oben angeführten Maßnahmen bis zu 45 % der Wartungskosten einsparen lassen.

4.3.2 Sicherung der Wartbarkeit

Typische Lebenszyklen von kommerziellen Produkten umfassen ca. 10 bis 15 Jahre Nutzungsdauer, in der das Produkt eine Evolution durchmacht und gepflegt werden muss. Aus Sicht der Qualitätsgestaltung stellt sich die Frage, wie die Wartbarkeit bereits bei der Entwicklung positiv beeinflusst werden kann, um den Aufwand in der Pflegephase möglichst gering zu halten.

Parnas [Parn79] schlägt in diesem Zusammenhang Folgendes vor:

- Strukturiere die Anforderungen in Teilmengen, und unterscheide Anforderungen, die später Erweiterungen nach sich ziehen. Diese so gewonnenen Teilmengen von Anforderungen erleichtern den Software-Entwurf und fördern die Flexibilität.
- Verwende beim Entwurf des Produkts Abstraktionsschichten. Dadurch können bei Änderungen Funktionen ohne Seiteneffekte hinzugefügt oder entfernt werden.

Um beide Vorschläge von Parnas erfüllen zu können, ist es zweckmäßig, die Anforderungen und den Entwurf in Form von Modellen darzustellen. Analyse- und Entwurfsentscheidungen lassen sich damit einfacher überprüfen. Die systematische Anwendung der Unified Modeling Language (UML) hilft bei der Erstellung von Anforderungsmodellen.

Der erste Vorschlag von Parnas lässt sich durch eine Produktplanung (Produktmanagement) in Form von Releases umsetzen. Produktfunktionen, die als nicht dringlich gelten, werden für ein nachfolgendes Release dokumentiert. Voraussetzung ist natürlich, dass die Weiterentwicklung (Evolution) des Produkts bereits bei der Erstentwicklung berücksichtigt wird und eine rechtzeitige Budgetierung (Zeit- und Kostenplan) des Verbesserungsrelease erfolgt.

Der zweite Vorschlag führt zu einer Schichtenarchitektur von Software-Systemen. Dies entspricht den Forderungen der Entwurfslehre im Software Engineering. Dabei werden einzelne Aspekte des Software-Systems konzeptionell einer Schicht (tier, layer) zugeordnet. Die erlaubten Abhängigkeitsbeziehungen zwischen den Aspekten werden bei einer Schichtenarchitektur dahingehend eingeschränkt, dass Aspekte einer „höheren“ Schicht nur solche „tieferer“ Schichten verwenden dürfen. Ein System mit Schichtenarchitektur bezeichnet man auch als mehrschichtig. Die den Schichten zugeordneten Aspekte können dabei je nach Art des Systems oder Detaillierungsgrad der Betrachtung z. B. Funktionalitäten, Komponenten oder Klassen sein. Häufig finden sich in Software-Systemen wie auch bei verteilten Anwendungen folgende Schichten:

- Präsentation
- Steuerung
- Anwendung
- Datenzugriff
- Datenhaltung

Die unterste Schicht dient der Verwaltung der Datenbasis, die durch Dateien oder ein Datenbanksystem realisiert wird. Die Struktur dieser Schicht leitet sich aus dem konzeptionellen Datenmodell ab. Die darüberliegende Schicht enthält Grundoperationen (Datenzugriffe) der

Applikation, beispielsweise Operationen, die Daten aufbereiten oder Daten in bestimmter Reihenfolge zur Verfügung stellen.

Die nächst höhere Schicht (Anwendungsschicht) enthält applikationsspezifische Operationen. Sie wird als Schicht der Benutzeroperationen bezeichnet. Liegt eine gute Modularisierung vor, so sind diese Operationen beispielsweise in Form von Datenkapseln realisiert. Durch Verkettung dieser Operationen wird der Dialogablauf festgelegt. Die Steuerung und Aktivierung des Dialogablaufs erfolgt in der nächsten Schicht (Steuerung, Dialogsteuerung). Durch eine solche Schichtenstruktur lassen sich Änderungen rasch lokalisieren und implementieren.

Wir schlagen daher folgende Maßnahmen für eine bessere Wartbarkeit vor:

- Spezifiziere konkrete Wartungsziele in Form konkreter Ausprägungen von Merkmalen, und lege Wartbarkeitsanforderungen bei der Formulierung des Projektauftrags oder im Rahmen des Applikationmanagements fest.
- Pattern-basiertes Software-Reengineering
  Anstatt Systeme mit ihrer Originalarchitektur zu warten, werden sie durch Verwendung von Design-Pattern reengineert. Design-Pattern kombinieren erfolgreiche und etablierte Designerfahrungen von Experten in Form einer Menge von Software-Komponenten, die ein bekanntes Verhalten und eine bessere Struktur aufweisen ([Chu00], [Pree94]).
- Etabliere regelmäßig analytische Qualitätsmaßnahmen, wie z. B. Audits und Reviews, und prüfe die Realisierung der Wartbarkeitsanforderungen bzw. die Einhaltung der Regeln für die Wartung.
- Wähle eine wartungsfreundliche Programmiersprache aus, wie beispielsweise Java oder C++.
- Verbessere die Programmdokumentation in Richtung adressatengerechte Dokumentation und Selbstbeschreibungsfähigkeit des Programms.

Zusammenfassend lässt sich feststellen, dass durch einen ingenieurmäßigen Entwicklungsansatz die Situation bei der Wartung verbessert wird. Die gegenwärtige Problematik in der Praxis besteht darin, dass zwar ein generelles Einverständnis über die Notwendigkeit der oben beschriebenen Maßnahmen existiert, sie aber als langfristig wirksam eingestuft werden. Der heute tätige Software-Manager besitzt leider in der Regel einen sehr kurzfristigen Entscheidungshorizont. Maßnahmen, die auf diesen Horizont abgestimmt sind, bringen meist nur einen kurzfristigen Nutzen. Dies führt häufig zu einer Diskrepanz zwischen einem längerfristigen Software-Engineering-Konzept und der dann tatsächlich realisierten Software-Lösung.

4.3.3 Organisation der Wartungsaktivitäten

Wartungsaktivitäten können entweder von den Entwicklern selbst oder durch eine eigene, von den Entwicklern unabhängige Wartungsmannschaft durchgeführt werden. Beide Formen haben Vor- und Nachteile.

„Entwickler warten selbst“

Vorteile:

- Die Entwickler besitzen die besten Kenntnisse über ihre Produkte. Die Wartungsarbeit kann rasch durchgeführt werden.
- Die Benutzer haben nur mit einer Gruppe von Informatikern zu tun, die für eine Applikation oder ein Produkt verantwortlich sind.
- Die Entscheidung („Wer darf entwickeln?“, „Wer muss warten?“) wird erleichtert, da jeder Entwickler auch warten muss.
- Die Wahrscheinlichkeit, dass bei der Entwicklung die Anforderungen an die Wartbarkeit berücksichtigt werden, ist groß.

Nachteile:

- Wenn ein Entwickler zu viel Wartungsarbeit erledigen muss, ist die Wahrscheinlichkeit groß, dass er die Informatik-Organisation verlässt.
- Probleme können entstehen, wenn ein Entwickler mit ausgezeichnetem Produktwissen die Informatik-Organisation verlässt und es für ihn keinen entsprechenden Ersatz gibt.
- Es besteht die Gefahr, dass die Entwickler zu viel Zeit damit verbringen, ihr Produkt zu perfektionieren.
- Wenn große Entwicklungsprojekte zu realisieren sind, besteht die Gefahr, dass die Wartungsarbeit gekürzt, verschoben oder an unerfahrene Entwickler übertragen wird.

„Die Wartung erfolgt durch eigenes Wartungspersonal“

Vorteile:

- Eine qualitativ bessere Dokumentation liegt vor. Es existieren formale Übergaberichtlinien, wenn ein Produkt in die Wartung gelangt.
- Formale Richtlinien („Vorgehen für die Wartung“) werden etabliert, um Wartungsanforderungen zu implementieren.
- Es entsteht eine spezifische Infrastruktur (Werkzeuge, Know-how etc.), und es gibt Spezialisten für die Wartung.

Nachteile:

- In bestimmten Wartungssituationen ist die Unterstützung durch die Entwickler nötig. Dies führt zu einem zusätzlichen Kommunikationsaufwand.
- Die Benutzer haben mit zwei Gruppen von Informatikern zu tun.
- Um ein Produkt warten zu können, muss man es zuerst genau kennen. Dafür ist ein Lernaufwand nötig.

Bei beiden Formen müssen eine zentrale Erfassung von Wartungsanforderungen und ein Gremium existieren, das die Wartungsanforderungen prüft, genehmigt, ablehnt oder abändert. Wir nennen dieses Gremium Change Control Board (CCB).

Die Zusammensetzung des Gremiums hängt von der Informatik-Organisation ab. Es ist sinnvoll, dass es Vertreter des Topmanagements, der Entwicklerorganisation oder der Wartungs-

organisation, der Benutzerorganisation (entweder Benutzervertreter oder direkt aus der Linie der Fachorganisation), einer zentralen Informatik-Planungsgruppe und der Prozess- und Qualitätsgruppe umfasst.

Unabhängig von der Organisationsform ist ein einfacher Prozess für die Wartung zu etablieren. Ein solcher Wartungsprozess könnte folgendermaßen aussehen:

- Entgegennahme der Wartungsanforderung und Grobschätzung der Kosten
- Analyse der Wartungsanforderungen und Erstellen eines Lösungskonzepts
- Aktualisieren der Dokumentation
- Aktualisieren von Quell- und Objektcode
- Freigabeprüfung
- Abschluss der Wartungsaktivität

Im Folgenden wird dieser Prozess zur Abwicklung von Wartungsaufträgen im Detail beschrieben.

Entgegennahme der Wartungsanforderung und Grobschätzung der Kosten

Folgende Aktivitäten sind durchzuführen:

- Identifikation des Wartungsfalls für die Aufzeichnung aller Kosten und Aufwände;
- Einordnung und Abgrenzung der Wartungsanforderung in Bezug auf die Art der Anforderung (z. B. Fehlerkorrektur, Anpassung, Tuning) und auf die vorhandenen Produkte/Applikationen;
- Grobschätzung der Kosten der Wartungsanforderung unter Berücksichtigung des Zustandes und der Wartbarkeit des Produkts:
  - Kosten der Lokalisierung des Fehlers/der Änderung
  - Kosten der Aktualisierung der Dokumentation
  - Kosten der Änderung des Codes und der Datenbank
  - Kosten der Software-Prüfung (Testen, Review)
  - Kosten der Installation der geprüften Module in der Produktionsumgebung
  - Kosten der Erstellung des Wartungsberichts

Die Grobschätzung der Wartungsanforderungen und der tatsächliche Aufwand werden im Wartungsbericht dokumentiert.

Analyse der Wartungsanforderungen und Erstellen eines Lösungskonzepts

Im nächsten Schritt sollte eine detaillierte Analyse erfolgen, welche Art von Wartung vorliegt und welche Auswirkungen die Änderungen/Erweiterungen auf die vorhandenen Software-Elemente und deren Beziehungen haben. Je besser das Produkt modularisiert ist (z. B. durch den Einsatz von Datenkapseln oder abstrakten Datentypen), umso geringer sind die Auswirkungen von Änderungen und Erweiterungen. Je größer und je schlechter modularisiert ein Produkt ist, desto aufwendiger ist dieser Analyseschritt. Werkzeuge können ihn unterstützen.

Ob und mit wie viel Aufwand Software gewartet werden muss, hängt stark ab von der betreffenden Software (Fehlerdichte, Wartbarkeit), ihrem Einsatz (unterschiedliche Anwendungsszenarien, dem Wunsch, bestimmte Attribute zu verbessern) und der Einsatzdauer (änderndes Umfeld). Bei unternehmenskritischer Software leistet die Software-Wartung in der Regel einen erheblichen Beitrag zur Investitionssicherheit, stellt andererseits aber auch einen erheblichen Kostenfaktor dar. Daher sind Wartungsvereinbarungen bei unternehmenskritischer Software häufig zu finden. Je nach vereinbartem Service Level liegen die jährlichen Kosten dabei üblicherweise in der Größenordnung von 10 % bis 30 % der Investitionssumme der Software.

Wenn bereits beim Systementwurf das notwendige Augenmerk auf eine angemessene Wartbarkeit gelegt wird, kann unnötig hohen Aufwänden für die Software-Wartung vorgebeugt werden. Bei hohen Wartungsaufwänden wird die Software-Wartung in der Regel von einer fest organisierten Gruppe von Mitarbeitern (Wartungsorganisation) in einem geordneten Wartungsprozess betrieben.

Aufgrund der Grobschätzung und der Analyse fällt eine Entscheidung, ob

- ein Vorschlag für ein Release-Projekt erarbeitet wird, um die Wartungsanforderung befriedigen zu können, oder
- der Wartungsfall durchgeführt wird.

Im ersten Fall führt die Wartungsanforderung zu einem neuen Release des Produkts. Der dem Produkt zugehörige Release-Plan muss daraufhin geändert werden. Dies löst auch projektübergreifende Planungsprozesse aus, um Termine und Ressourcen für das Release-Projekt zu reservieren.

Im zweiten Fall ist es sinnvoll zu entscheiden, ob der Wartungsfall sofort erledigt werden muss, oder ob man erst mit den Änderungen beginnt, wenn mehrere kleine, nicht dringende Wartungsfälle vorliegen. Der Vorteil bei der Sammlung von Wartungsfällen zu einem Paket ist ein ökonomischer Einsatz der Wartungsressourcen wie Zeit und Personal.

Eines der Hauptprobleme bei der Wartung ist das Erzeugen von Fehlern durch Änderungen und Erweiterungen, aber auch durch Fehlerkorrekturen selbst. Einige Bemerkungen bezüglich der durch Wartungsaktivitäten verursachten Fehlertypen [Coll87]:

- 10 % der Fehler entstanden durch die Korrektur von Fehlern.
- 21 % der Fehler traten in den Erweiterungen auf.
- 53 % der Fehler entstanden durch Auswirkungen von Erweiterungen auf bereits existierende Produktteile.
- 16 % der Fehler waren nicht klassifizierbar.

Nachdem Ursachen und Auswirkungen der Änderungen abgeschätzt sind, ist ein Lösungskonzept zu erstellen, was alles geändert werden muss und welche Bausteine neu zu entwickeln sind. Die Struktur des geänderten oder erweiterten Systems ist darzustellen, z. B. durch grafische Hilfsmittel, Datenfluss-, Hierarchie- und Aufrufdiagramme, oder auch formalisierte Hilfsmittel wie attributierte Grammatiken. Das Lösungskonzept enthält auch eine Beschreibung, in welcher Reihenfolge die Änderungen und Erweiterungen zu implementieren sind.

Aktualisieren der Dokumentation

Bevor noch irgendein Teil des Codes geändert wird, ist festzustellen, auf welcher Ebene der Produktbeschreibung Änderungen durchzuführen sind. Es hat sich als praktikabel erwiesen, folgende Produktbeschreibungsebenen zu unterscheiden:

- Spezifikationsebene
  z. B. Anforderungsdefinition in Form von Datenfluss- und Entity-/Relationship-Diagrammen, Benutzerhandbuch
- Entwurfsebene
  z. B. Systementwürfe in Form von Modulabhängigkeitsgraphen, Modulentwürfe in Form von Struktogrammen
- Code-Ebene
  z. B. Modulbeschreibungen in Form von Kommentaren

Der Einsatz von Werkzeugen kann das Lokalisieren von Dokumentationsänderungen wesentlich erleichtern. Auch existieren bereits Hilfsmittel, um die Verfolgbarkeit von Änderungen in der Dokumentation und dem Code zu unterstützen.

Aktualisieren von Quell- und Objektcode

Bereits bei der Erstellung der Wartungsdokumentation (spätestens zum Zeitpunkt der Inbetriebnahme) muss ein Leitfaden für die Aktualisierung des Quell- und des Objektcodes angelegt werden, der im Wartungsfall benutzt wird. Er enthält Hinweise, in welchen Dateien des Archivierungssystems welche Programme zu finden bzw. welche Jobs zur Übersetzung und zum Binden nötig sind. In der Regel wird der Code zuerst in einer Entwicklungs- oder Wartungsbibliothek geändert und nicht sofort in der Produktionsbibliothek.

Freigabeprüfung

Unter Freigabeprüfungen verstehen wir Tests sowie Reviews von modifizierten Produkten. Besondere Bedeutung für die Produktivität in der Wartung hat die Wiederholbarkeit der Tests und die Archivierung der Testfälle. Alle geänderten Produktteile sind einem Review zu unterziehen.

Abschluss der Wartungsaktivität

Für einen ordnungsgemäßen Abschluss sind folgende Aktivitäten durchzuführen:

- Die Quellprogrammlisten und die Testergebnisse werden archiviert.
- Im Modulheader jedes geänderten Moduls wird unter dem Punkt „Änderungen“ die Wartungsaktivität beschrieben und der Name des Verantwortlichen vermerkt.
- Die Dokumente Wartungsanforderung und Wartungsbericht werden vervollständigt, archiviert und eine Kopie an den Auftraggeber geschickt.

Der Status aller Wartungsaktivitäten ist in Form eines Wartungsübersichtsberichts periodisch aufzuzeigen. In diesem Bericht werden die offenen und die abgeschlossenen Wartungsaktivitäten aufgezählt. Es sind auch jene Maßnahmen anzuführen, die durch Notfälle verursacht wurden.

Für das Management sind, basierend auf den Wartungsberichten, verschiedene Kenngrößen und Statistiken zu erstellen, die die Transparenz sowie das Problem- und Kostenbewusstsein bei den Entscheidungsträgern fördern. Einige Kenngrößen, die sich bewährt haben, sind die folgenden:

- Anzahl der pro Quartal von Wartungsaktivitäten verursachten Fehler /1000 NLOC;
- Anzahl aufgewendeter Personentage je Wartungskategorie (z. B. Korrektur) pro Quartal;
- durchschnittliche Anzahl von Programmänderungen pro Programm;
- durchschnittliche Zeitdauer von der Erfassung bis zum Abschluss einer Wartungsanforderung.

Diese Kenngrößen stellen eine Basis für Entscheidungen dar, ob Verbesserungen des Planungs- und Entwicklungsprozesses, beim Personaleinsatz oder bei der Projektinfrastruktur nötig sind. Sie fördern ein besseres Verständnis für die Probleme und Anliegen der Wartung.

4.4 Bedeutung des Qualitätsmanagements für die Wartung

Das Qualitätsmanagement stellt sicher, dass die Anforderungen an das Software-Produkt im Rahmen der zeitlichen Evolution konsistent erfüllt werden. Neben den schon beschriebenen organisatorischen und technischen Maßnahmen gewinnt die Sanierung von Alt-Software („Re-Engineering“) immer mehr an Bedeutung. Unter Sanierung verstehen wir die Restrukturierung eines Software-Produkts mit dem Ziel, Qualitätsmerkmale wie z. B. Lesbarkeit, Effizienz oder Wartbarkeit zu verbessern.

Der erste Schritt bei der Sanierung ist die statische Programmanalyse, um ein Modell des Programms (Struktur, Schnittstellen, Funktionen) zu bekommen. Anschließend wird versucht, die alte Struktur des Produkts zu verbessern oder neu festzulegen („Redesign“). Danach werden alte, aber noch verwendbare und neu geschriebene Software-Bausteine zum sanierten Produkt montiert. Bekannte und bewährte Hilfsmittel und Werkzeuge des Qualitätsmanagements für die Wartung sind:

- Standards
- Kenngrößen
- Reviews und Audits
- Software-Informationssysteme
- Wartungswerkzeuge
- Aufklärung und Training

Standards

Die zunehmende Komplexität der Produkte und die Personalfluktuation zwingen sowohl zu konsequenter Durchsetzung von Entwicklungsstandards als auch zu standardisiertem Vorgehen bei der Wartung. Es gibt international gültige Standards für die Wartung wie z. B. ANSI/IEEE Std 1219-1993 (Standard for Software Maintenance), den Standard ISO/IEC 12207-1996 (Standard for Information Technology – Software Lifecycle Processes), der auch einen Wartungsprozess vorschlägt, und ISO/IEC 14764 (Standard for Information Technology – Software Maintenance).

Der IEEE Standard 1219 beschreibt ein iteratives Vorgehen bei der Durchführung der Wartung. Die Software-Wartung beginnt nach Auslieferung der Software. Der Trigger für eine Iteration des Wartungsprozesses liefert immer eine Änderungsanfrage. Diese wird als Modification Request erfasst und kategorisiert (korrektiv, adaptiv oder perfektiv).

Bei der Bearbeitung einer Änderungsanfrage werden verschiedene Phasen durchlaufen. Zunächst wird eine Analyse der Anfrage bezüglich ihrer Umsetzbarkeit und Auswirkungen auf das bestehende System durchgeführt. Auf Grundlage dieser Analyse entscheidet ein Change Control Board (CCB), ob dem Request entsprochen wird. Bei einer positiven Entscheidung durch das CCB wird in der Design-Phase die Umsetzung der Änderungsanfrage geplant und in der Implementierungsphase umgesetzt. Anschließend werden alle Änderungen mittels eines Systemtests durch die Entwickler und eines Akzeptanztests vom Kunden überprüft. Abschließend wird das System in der Auslieferungsphase allen Benutzern zur Verfügung gestellt.

Der Prozess von ISO/IEC 12207-1995 umfasst folgende Aktivitäten:

- Prozess-Implementation
- Problem- und Modifikationsanalyse
- Modifikations-Implementation
- Wartungsreview/-freigabe
- Migration
- Software-Rückzug (Retirement)

Dieser Wartungsprozess enthält alle Aktivitäten und Aufgaben für das Wartungspersonal, legt aber nur indirekt eine Reihenfolge der Aufgaben fest. Er ist mit einer geeigneten Wartungsinfrastruktur auszustatten.

ISO/IEC 12207, Standard for Information Technology – Software Lifecycle Processes verfeinert die Wartungsaufgaben von ISO/IEC 12207. Zu Beginn der Wartung wird der Prozess implementiert (eingeführt). Die Migration oder die Ablösung der Software erfolgt einmal am Ende des Wartungsprozesses. Dagegen werden die Aktivitäten Problem- und Änderungsanalyse, Änderungsumsetzung und Review sowie Abnahme iterativ bei jeder Änderungsanfrage durchlaufen.

Die Wartung einer Software können deren Entwickler oder spezielle Wartungsingenieure durchführen. Eine Wartung durch Wartungsingenieure kann sowohl intern als auch extern über einen Lieferanten erfolgen. Dafür werden speziell ausgerüstete Wartungsumgebungen vorgeschlagen.

Diese Richtlinien und Standards helfen,

- eine Software-Wartung plan- und steuerbar zu machen;
- mögliche Verbesserungen der Wartbarkeit zu erkennen;
- die Transparenz von Wartungsarbeit in Informatik-Organisationen zu verbessern;
- ein Bonussystem für das Wartungspersonal einzuführen;
- vorhandene Standards zu verbessern.

Der Schlüssel für die Durchsetzung eines Standards liegt in seiner Einfachheit, Anpassbarkeit, aber auch in der Werkzeugunterstützung. Standards müssen gepflegt und permanent auf ihre Anwendbarkeit überprüft werden. Die Anwendbarkeit wird insbesondere dadurch verbessert, wenn explizit festgehalten wird, wann der Standard nicht angewendet werden soll. Akzeptierte und in der täglichen Arbeit verwendete Standards können einen hohen Nutzen erbringen. Ein nicht akzeptierter Standard oder das Beharren auf einem nutzlosen Standard ist für eine Informatik-Organisation gefährlich, ja sogar schädlich (z. B. durch Festhalten an überholter, veralteter Informatik-Technologie). Die Pflege der Standards ist eine klassische Aufgabe einer Prozess- und Qualitätsorganisation. Einhaltung und Durchsetzung von Standards müssen aber vom Management unterstützt werden.

Kenngrößen für Wartung und Pflege

Die Sammlung von Know-how über Wartungskenngrößen sind Aufgabe einer Prozess- und Qualitätsgruppe. Es gibt bereits einige ermutigende Erfahrungen beim Einsatz von Kenngrößen im Rahmen von Wartungsaktivitäten [Dumk00].

Kafura und Reddy [Kafu87] untersuchten Code- und Systemstruktur-Kenngrößen (McCabe-Maß, Halstead's Aufwandsmaß, LOC, Informationsflussmaß von Henry und Kafura, Steuerflussmaß von McClure, Maß von Woodfield und das Maß der logischen Stabilität von Yau und Collofello) in Zusammenhang mit der Wartung von vier Releases eines relationalen Datenbanksystems. Sie stellten fest, dass die Kenngrößen sehr gut die wachsende Komplexität eines Systems aufzeigten und waren sogar in der Lage, die schlecht strukturierten Komponenten des Systems zu identifizieren.

Für den praktischen Einsatz empfehlen wir, folgende Kenngrößen aufzuzeichnen und zu analysieren:

- Planungsmetriken
  - Anzahl Codezeilen
  - Anzahl Functionpoints
  - Anzahl Aufträge
  - Personalaufwände
  - Anzahl Wartungsfehler
  - Software-Wachstumsrate
  - Anzahl Software-Applikationen
  - Anzahl Releases

- Analysemetriken
  - Änderungsumfang (in LOC/Functionpoints)
  - Anzahl betroffene Module
  - Anzahl betroffene Datenelemente
  - Ablaufkomplexität
  - Datenflusskomplexität
  - Modulkopplungsgrad
- Testmetriken
  - Anzahl betroffener Pfade
  - Anzahl Prädikate
  - Anzahl Ablaufzweige
  - Anzahl Testfälle

Kenngrößen sind zu einem unentbehrlichen Instrument geworden, um einerseits mehr Transparenz im Wartungsprozess zu schaffen und andererseits die Auswirkungen von Wartungsaktivitäten auf das Produkt leichter beurteilen zu können.

Reviews und Audits

Reviews werden zum Abschätzen der Auswirkungen von Änderungen und für den Nachweis, dass ein Produkt korrekt geändert wurde, eingesetzt.

Zur Übergabe von Produkten von der Entwicklung in die Wartung haben sich Abnahme- und Übergabe-Audits bewährt. Bei Abnahme-Audits wird geprüft, ob die Abnahmekriterien erfüllt sind. Beim Übergabe-Audit wird einerseits überprüft, ob alle Produktelemente (Merkmal physische Vollständigkeit) und alle Produktfunktionen (Merkmal funktionale Vollständigkeit) vorhanden sind, und andererseits, ob alle Voraussetzungen für die Wartung (Checkliste) erfüllt sind.

Darüber hinaus empfiehlt es sich, periodisch Audits zur Wartbarkeit jener Programme abzuhalten, die sich in der Betriebsphase befinden. Durch diese Art von Audits werden sehr rasch potenzielle und aktuelle Wartungsprobleme aufgedeckt.

Software-Informationssysteme

Liegen umfangreiche Software-Systeme (z. B. Tausende Programme, Hunderte Datenbanken etc.) vor, benötigt die Wartungsorganisation ein adäquates Hilfsmittel, um ihre Aktivitäten wirkungsvoll durchführen zu können. Nur wenn es gelingt, Struktur- und Übersichtsinformationen über Applikationen, Module, Datenbanken, Dateien etc. rasch und vollständig bereitzustellen, kann die Wartungsarbeit produktiv gestaltet werden. Hier spielen Repository-Systeme mit geeigneten Analyse- und Auswertungswerkzeugen eine entscheidende Rolle.

Wartungswerkzeuge

Die Wartungsverantwortlichen haben zunehmend Schwierigkeiten, die wachsende Menge an altem Code zu verstehen, zu prüfen und zu handhaben. Insbesondere die Analyse und Verwal-

tung von Schnittstellen bzw. die redundante Verwendung von Datenstrukturen sind mit Abstand die größten Herausforderungen, wenn beispielsweise ca. 8 Millionen PL1-Codezeilen einer Zürcher Großbank zu warten sind. Diese Codemenge ist in den letzten 30 Jahren evolutionär gewachsen und kann beim gegenwärtigen Tagesgeschäft der Bank mit den laufenden neuen Anforderungen nicht einfach ersetzt werden. Die Verantwortlichen haben sich daher für eine evolutionäre Pflegestrategie entschieden, bei der jährlich ca. 20 Millionen Franken ausgegeben werden, um die Applikations- und Software-Architektur in Stand zu halten. Ähnlich wie bei einem alten Haus oder Stadtteil ist eine kontinuierliche Renovierung und Erneuerung der vorhandenen Strukturen notwendig. Dabei werden keine neuen Funktionen in die Applikationssysteme eingebaut, sondern nur Strukturbereinigungen durchgeführt, wie beispielsweise Datenredundanzen beseitigt und Schnittstellen vereinfacht bzw. neu definiert. Die Konsequenz daraus ist, dass man Hilfsmittel und Werkzeuge benötigt, um aus den umfangreichen Software-Mengen Struktur- und Schnittstelleninformationen zu extrahieren. Beispiele für solche Werkzeuge sind Informationsgeneratoren (Dokumentationsgenerierung auf Programm- und Systemebene) und Browser.

Eines der besten Werkzeuge in einer .NET-Umgebung ist ReSharper. Wichtige Features sind Code Refactoring, Code Navigation und Code-Analyse. Dies sind einige der vielen Features, die helfen, effiziente Wartungsarbeit zu leisten. Typische Werkzeuge für die Wartung:

- Architektur- und Codeanalysewerkzeuge
- Compiler und Querverweis-Generatoren
- Code- und Daten-Komparatoren
- Trace-Werkzeuge
- Testdatengeneratoren
- Testabdeckungsanalysatoren
- Debuggerwerkzeuge
- Source-Verwaltungswerkzeuge
- Bibliotheksverwaltungssysteme
- Archivierungs- und Versionskontrollsysteme
- Data-Dictionary-Systeme
- Werkzeuge zur Lösung und Verwaltung von Fehlern und Problemen
- Auftragsverwaltungswerkzeuge
- Configuration-Management-Werkzeuge

Aufklärung und Training

Viele Manager sind für technische und organisatorische Wartungsprobleme unzureichend ausgebildet. Die Folgen davon sind eine fehlende Kostenkontrolle bzw. Investitionsplanung, aber auch chaotische Zustände beim Einsatz und bei der Pflege alter Produkte.

Software-Evolution kann auf verschiedenen Ebenen betrieben werden (Erstentwicklung, Wartung zwischen Releases, Weiterentwicklung über verschiedene Releases oder Entwicklung

von Nachfolgesystemen) und muss vom Management aktiv geplant (Release-Konzept), gesteuert und kontrolliert werden (Release-Abnahme). Dazu ist eine geeignete organisatorische und technische Wartungsumgebung bereitzustellen.

Sicherlich wird ein Großteil der heutigen Wartungsproblematik durch die Auslieferung von qualitativ minderwertiger Software verursacht, die besonders bezüglich ihrer Wartungsfreundlichkeit Defizite aufweist. Neben Qualitätsmaßnahmen benötigt man eine Vielzahl von Werkzeugen, mit deren Hilfe sich Verbesserungen erzielen lassen. Diese Werkzeuge dienen auch im eigentlichen Wartungsprozess als Hilfsmittel, um die Ausbreitung sowie Fortpflanzung von Fehlern zu vermeiden und eine qualitativ hochwertige Wartung sicherzustellen. Sie unterstützen notwendige Arbeiten wie z. B. die Dokumentation.

Trotz aller technischen Vorgaben und Unterstützungen ist der psychologische Aspekt nicht zu vernachlässigen. Sämtliche Maßnahmen, zu denen unter anderem der Einsatz eines speziellen Wartungsprozesses, spezieller Methoden und Werkzeuge gehört, müssen unter Berücksichtigung der in den Wartungsprozess einbezogenen Beteiligten durchgeführt werden. Da dieser Bereich auf eine überdurchschnittliche Mitarbeiterleistung angewiesen ist, liegt im Einsatz von Führungsmaßnahmen ein erhebliches Produktivitätspotenzial.