

2.1.6 Configuration Management

Im **Configuration Management** wird die gesamte IT-Infrastruktur eines Unternehmens in Form eines logischen Modells abgebildet. Ziel ist es, stets einen gesicherten aktuellen Zugriff auf die Daten aller IT-Assets und IT-Konfigurationen sowie den damit verbundenen IT Services innerhalb eines Unternehmens zu gewährleisten. Dies erfolgt mit Hilfe eines Datenbankmodells, der **Configuration Management Database (CMDB)**. Hier werden vielfältige unterschiedliche Informationen zu den IT Objekten gesammelt, gepflegt und anderen Prozessen zur Verfügung gestellt. Die **CMDB** ist somit die wichtigste Daten- und Informationsquelle innerhalb des IT-Servicemanagements, mit der alle Prozesse des **Service Support** und des **Service Delivery** interagieren. Vor allem die Prozesse Incident Management, Problem Management und Change Management können ohne CMDB nur mit großen Einschränkungen betrieben werden.

Die fünf Hauptaufgaben im **Configuration Management** sind

- **Planung**, die Planung und Definition von Konfigurationskomponenten (CIs) bezüglich Zweck, Umfang, technischem und organisatorischem Kontext
- **Identifikation**, Identifikation und Auswahl von CIs inklusive Abhängigkeiten und Verantwortlichkeiten (Ownership) sowie die eindeutige Versionierung und Kennzeichnung von CIs
- **Kontrolle**, die Sicherstellung, dass sich in der Produktivumgebung nur autorisierte und in der CMDB registrierte Komponenten befinden
- **Statusinformationen**, Aufzeichnung und Historienführung über alle Änderungen an CIs während des gesamten Lebenszyklus
- **Verifizierung und Audits**, regelmäßige Reviews und Prüfungen zum Abgleich der physikalisch vorhandenen Daten in der CMDB

In der **CMDB** werden nicht nur die einzelnen Komponenten eines IT-Systems selbst, sondern insbesondere auch die Beziehungen

2.1.6 Configuration Management

der Komponenten untereinander abgebildet. Die einzelnen Objekte innerhalb der **CMDB** werden **Configuration Items (CI)** genannt. Ein CI ist dabei eine beliebige Einheit, die sich nur aus einer einzelnen oder aus mehreren einzelnen Objekten zusammensetzen kann, z.B. eine Netzwerkkarte, eine Festplatte, ein Kabel, ein kompletter PC, ein Server-Cluster oder ein Netzwerksegment. Auch Dokumente werden in der **CMDB** gleichermaßen als CI eingebunden. Dazu zählen beispielsweise Verträge, Betriebs- und Installationsanleitungen, Notfallpläne und Unternehmensrichtlinien.

Die CMDB ist die zentrale Daten- und Informationsbasis für alle ITIL Prozesse. In Abb. 43 ist das gesamte Prozessumfeld der CMDB abgebildet.

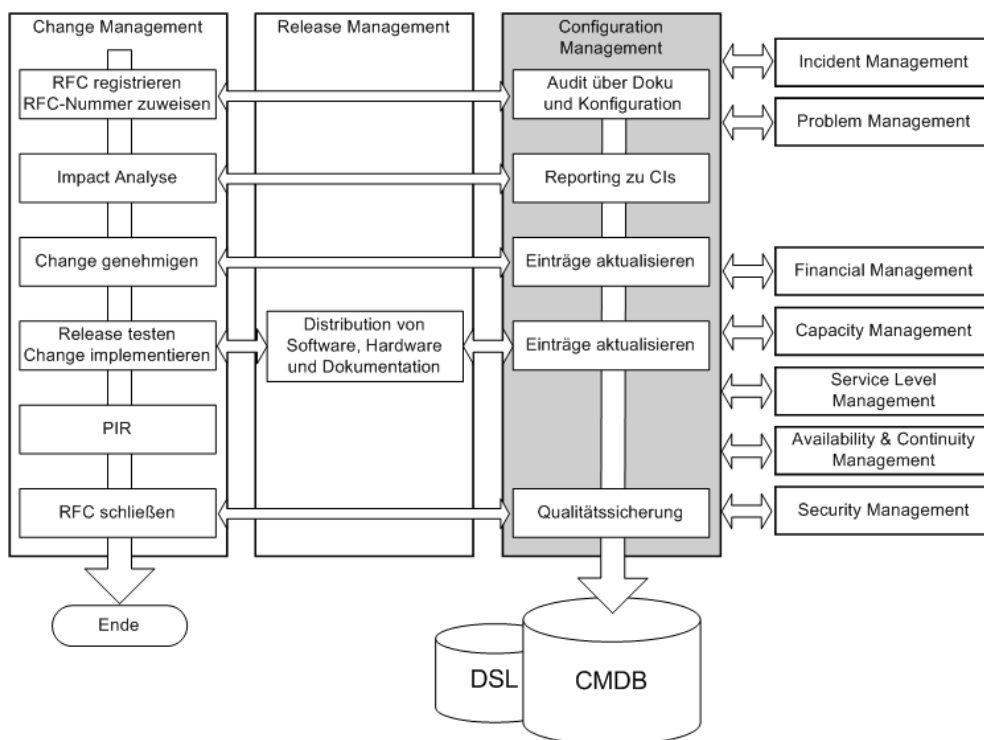


Abb. 43 CMDB-Prozessumfeld

Welcher Detaillierungsgrad im Einzelnen erforderlich ist, hängt von den jeweiligen Anforderungen im IT-Gesamtsystem ab. Bei

2.1 Service Support

der Definition der CIs sollte man jedoch darauf achten, die **CMDB** nicht mit zu filigranen Details zu überladen. Das Ganze sollte nicht in einer Datenzyklopädie ausufernd sein. Andererseits dürfen aber auch keine Informationsdefizite entstehen. Die geschäftlichen Anforderungen müssen abgedeckt sein und es sollte genügend Raum für Erweiterungen geben.

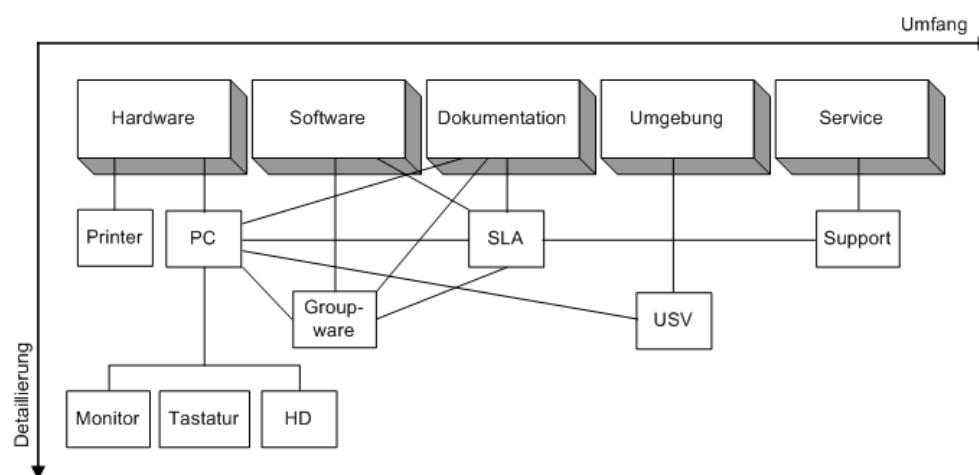


Abb. 44 CMDB-Detaillierungsgrad

Jedes CI in der **CMDB** besteht aus einem Datensatz aus individuellen Eigenschaften (**Attributen**) sowie Verlinkungen zu anderen Datensätzen und Dokumenten.

Abb. 45 gibt ein Beispiel, wie die Attribute von CIs strukturiert werden können. In Abb. 46 sind die von ITIL empfohlenen Attribute aufgelistet.

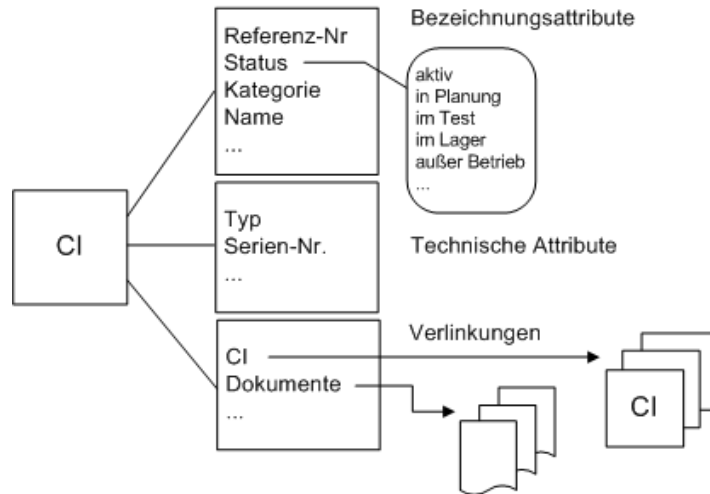


Abb. 45 CI-Attributstruktur

CI-Name	Parent CIs Relationship
Copy or Serial Number	Child CIs Relationship
Category	Relationships
Type	RFC Numbers
Model Number (hardware)	Change Numbers
Warranty Expiry Date	Problem Numbers
Version Number	Incident Numbers
Owner Responsible	Comment
Responsibility Date	Location
Source/Supplier	Accepted Date
Licence	Current Status
Supply Date	Scheduled Status

Abb. 46 CMDB-Attribute nach ITIL

Durch die CI-Verknüpfungen (Relationen) kann das Datenmodell der CMDB eine gegliederte objekthierarchische Struktur annehmen, wie sie nachfolgend in Abb. 47 schematisch dargestellt ist. Objektorientierte Ansätze sind bei einer CMDB besonders vorteilhaft, da mit den Mechanismen, Vererbung und Polymorphie, ein hohes Maß an Transparenz und Flexibilität erreicht werden kann.

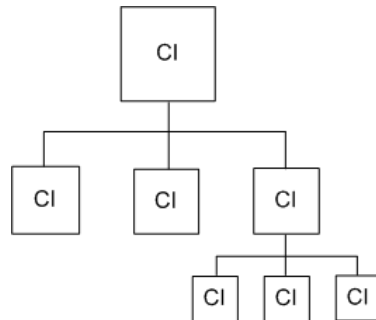


Abb. 47 CI-Objekthierarchie

Jedes CI trägt eine systemweit eindeutige Referenznummer (ID), mit der es jederzeit eindeutig im System identifiziert werden kann. Mit weiteren Attributen, z.B. Kategorie und Status, können Gruppen- und Workflowinformationen hinzugefügt werden. Die Inhalte der Attribute können sich ändern und müssen daher stetig gepflegt werden.

Wie schon erwähnt, sind auch die Verknüpfungen und Gruppierungen der einzelnen CIs in der **CMDB** abgebildet. Die Informationsqualität erreicht dadurch eine ganz neue Dimension. Im Gegensatz zu reinen Bestandsführungssystemen (**Asset Management-Systemen**), liefert die **CMDB** damit nicht nur Mengengerüste, sondern ermöglicht auch fundierte Folgen- und Schwachstellenanalysen. Die Verknüpfungen der CIs spiegeln die IT-Architektur wider. Dazu werden unterschiedliche Beziehungen definiert, wie z.B.

- X ist Teil von Y (z.B. eine Festplatte ist Teil eines Servers)
- X ist verbunden mit Y (z.B. ein PC ist mit einem Server verbunden)
- X benutzt Y (z.B. zwei Netzwerke benutzen eine Standleitung für VPN)
- X ist eine Ausprägung von Y (z.B. gleicher Drucker, jedoch mit größerem Papierfach)

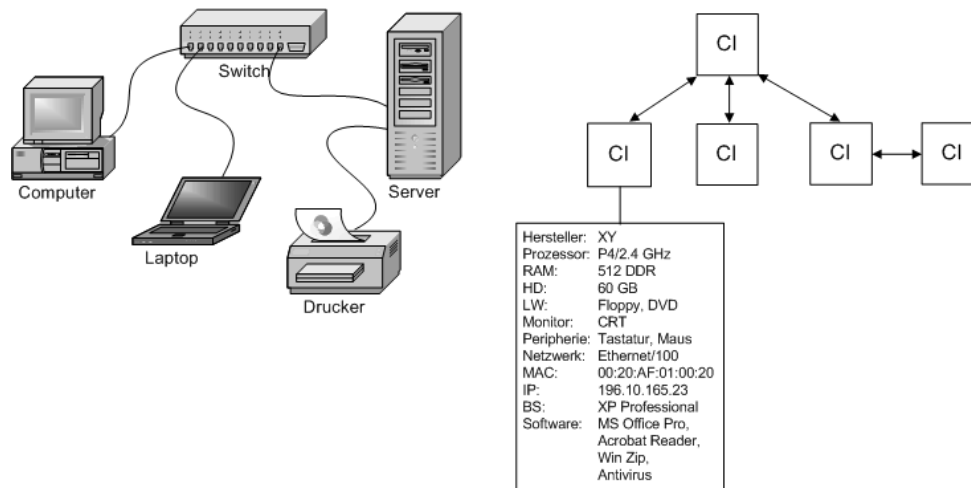


Abb. 48 CI-Verknüpfung

Abb. 48 veranschaulicht das CI-Prinzip der CI-Verknüpfung anhand eines einfachen Netzwerks. Es ist klar zu erkennen, dass durch einen Ausfall des Switches das komplette Netzwerk lahm gelegt wird. Der Switch ist hier also eine klassische Schwachstelle – ein Single **Point of Failure (SPOF)**. Fällt hingegen der Server aus, wird die Arbeitsfähigkeit sicher stark beeinträchtigt, indem weder Shares noch Drucker zur Verfügung stehen. Eine Kommunikation unter den übrigen beiden PC-Geräten wäre als Workgroup aber weiterhin prinzipiell möglich.

Anhand der **CMDB** kann man schnell einen Überblick gewinnen, welche Systeme bei bestimmten Ereignissen betroffen sind und welche Auswirkungen sich daraus ergeben könnten. Besonders in Krisensituationen, z.B. bei Virenbefall, kann das Change Management somit schneller und präziser reagieren.

CI-Lebenszyklus

Jedes CI hat immer nur eine begrenzte Lebensdauer, nach der es entsorgt und ggf. wieder ersetzt werden muss. Der Zeitraum, in dem ein CI im IT System präsent ist, wird als CI-Lebenszyklus (**CI Lifecycle**) bezeichnet. Während dieser Zeit durchläuft ein CI verschiedene Phasen und Status, die nachfolgend beschrieben werden. Der CI-Lifecycle beginnt, wie in Abb. 49 dargestellt,

ursächlich mit der Beschaffung (**Procurement**) eines CIs. Danach folgt die Betriebsphase (**Operating**) und am Ende des Lebenszyklus steht die Entsorgung (**Disposal**).



Abb. 49 CI-Lifecycle

(P) – Procurement

Diese Phase bildet den kompletten Beschaffungsprozess eines CIs, von der Bestellung bis zur Lieferung, aus Sicht des IT-Betriebs ab.

(O) – Operating

Diese Phase bildet den gesamten Zeitraum, in dem ein CI potentiell für den operativen Betrieb zur Verfügung steht, ab.

(D) – Disposal

Diese Phase beschreibt die Ausmusterung/Entsorgung von CIs aus dem produktiven Bereich.

In jeder Phase des CI-Lifecycles können die CIs unterschiedliche signifikante Zustände einnehmen, die durch eindeutige Status gekennzeichnet sind. Der Status eines CIs beschreibt jeweils den aktuellen Zustand, in dem sich ein CI momentan befindet. Es kann dabei zwischen veränderbaren (V) und finalen (F) Status unterschieden werden. Final bedeutet, dass hier das Ende eines Workflows erreicht wurde und somit dieser CI-Status nicht mehr verändert oder rückgängig gemacht werden darf.

Die nachfolgende Tabelle enthält ein Beispiel, welche Statusdefinitionen in den einzelnen Phasen definiert werden können.

Status	Phase	Art	Beschreibung
ordered	P	V	Bestellung wurde ausgelöst
in progress	P	V	Bestellvorgang befindet sich in Bearbeitung
cancelled	P	F	Bestellung wurde storniert
delivered	P	V	Ware wurde geliefert, Wareneingang bestätigt
returned	P	F	Ware wurde zurückgesendet (Retoure)
active	O	V	CI befindet sich produktiv im Einsatz
inactive	O	V	CI ist temporär nicht im produktiven Einsatz (z.B. Wartungsfenster)
in store	O	V	CI befindet sich zur weiteren Verwendung im Lager
planning	O	V	CI befindet sich in Planung/Vorbereitung
test	O	V	CI befindet sich im Test
stolen	D	V	CI wurde gestohlen
sold	D	F	CI wurde verkauft
discarded	D	F	CI wurde verworfen/ausgemustert/entwertet

Neben der Definition der einzelnen Status, müssen auch die Statusübergänge unter Berücksichtigung der Workflow-Anforderungen aller Steuerungsprozesse festgelegt werden. Die Statusübergänge sagen konkret aus, welche Status aufeinander folgen dürfen. **State Transition Diagramme** sind ein elegantes Mittel, die Statusübergänge grafisch abzubilden. Die nachfolgenden drei

2.1 Service Support

Diagramm-Abbildungen beziehen sich auf die eben genannten CI-Status und die jeweiligen Lifecycle-Phasen aus der zuvor stehenden Beispiel-Tabelle.

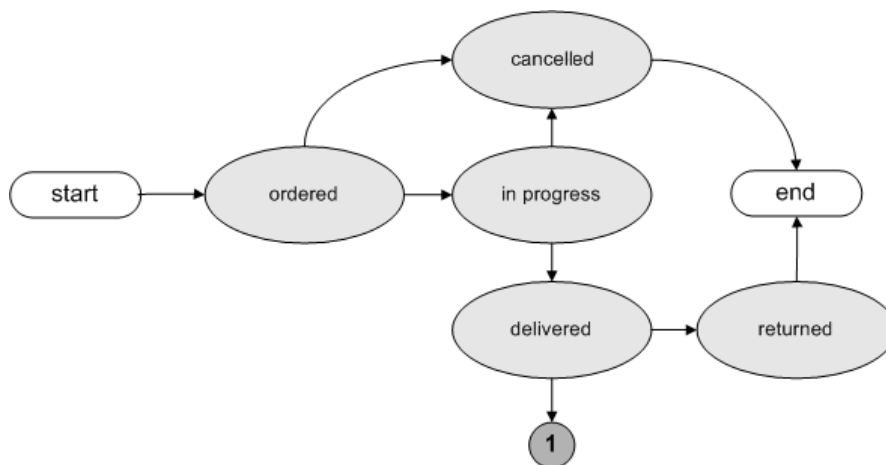


Abb. 50 CI-Statusübergänge – Procurement (P)

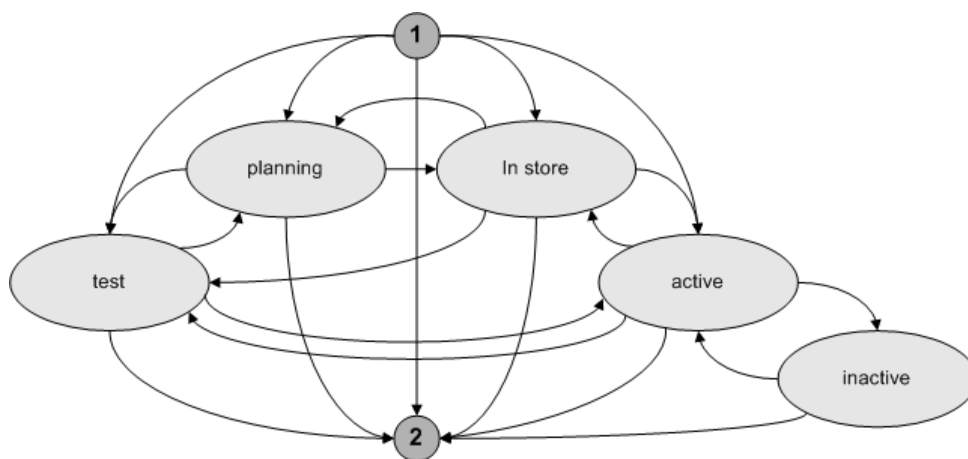


Abb. 51 CI-Statusübergänge – Operating (O)

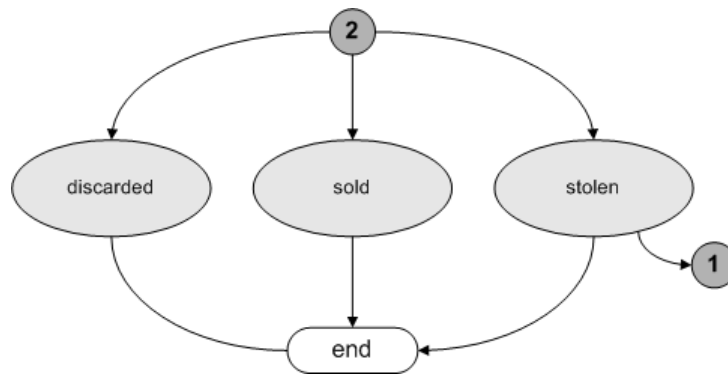


Abb. 52 Statusübergänge – Disposal (D)

Beachten Sie in Abb. 52 beim Status „stolen“ den Wiedereintrittspunkt nach 1, der es ermöglicht, wiedergefundene CIs jederzeit ins System zurückzuführen, da „stolen“ nicht als finaler Status definiert worden ist.

CMDB – Implementierung in der Praxis

In bestehenden Unternehmen haben sich im Laufe der Zeit, sozusagen historisch bedingt, viele unterschiedliche eigenständige Informationsträger etabliert, mit denen das Wissen des Unternehmens verwaltet und kommuniziert wird. Verschiedene Datenbanksysteme auf unterschiedlichen Plattformen, Excel-Tabellen, E-Mails, Textdokumente, usw. Der interne Datenaustausch erfolgt dabei in der Regel nur über punktuelle Schnittstellen, die selten standardisiert sind. Auch wenn man von einer durchgängigen homogenen Lösung somit meist weit entfernt ist, bedeutet die Implementierung einer **CMDB** nicht, alles Bestehende über den Haufen zu werfen und komplett von neuem zu beginnen.

ITIL schreibt nicht vor, wie ein CMDB-System inhaltlich und in seiner Architektur genau auszusehen hat oder gar welches Tool sich dazu am besten eignet. Idealerweise bietet sich dazu eine standardisierte Basis in Form einer leistungsfähigen relationalen Datenbank an. Die Funktionalität einer **CMDB** kann aber durchaus

auch problemlos in einer heterogenen Umgebung in einem virtuellen CMDB-Modell realisiert werden. Auch eine schrittweise Implementierung ist möglich. Im Vorfeld sollte auf jeden Fall eine gründliche IST-Stand-Analyse durchgeführt werden. Daraus können dann die weiteren Schritte abgeleitet werden, welche Schnittstellen am sinnvollsten sind, welche Altdaten migriert werden sollten, etc. Die frühzeitige Implementierung eines **Configuration Managements** zahlt sich sicher aus.

Objekte innerhalb der IT-Infrastruktur, die nicht als CI in der **CMDB** enthalten sind, können keine Prozesse durchlaufen. D.h., es gibt für sie dann weder Incidents noch RFCs!

Benefits des Configuration Managements

- Strukturierte Abbildung der IT-Infrastruktur
- Qualitätsgesicherte Daten- und Informationsbasis für alle Prozesse zur Erbringung wirtschaftlicher IT-Services
- Einheitliche Methoden und Tools zur Administration und Diagnose
- Unterschiedliche Sichtweisen können abgebildet werden wie z.B. für Impact-Analysen und Service Reporting
- Verbessertes Asset Management, leichteres Auffinden von „Leichen“ und mehrfachen Eintragungen
- Wirksame Kontroll- und Nachweismöglichkeiten in Bezug auf unterschiedliche Vertragsgegenstände (SLAs/OLAs/ UCs), Ressourcen und Lizenzen.
- Präzise Informationsgewinnung in der Planung, im Produktionsbetrieb und in Krisenfällen
- Einfachere Umsetzung von Changes. Besser nutzbare Synergie Effekte