INSTITUT FÜR INFORMATIK
DER LUDWIG–MAXIMILIANS–UNIVERSITÄT MÜNCHEN

Projektarbeit

Evaluation des TUM-Trouble Ticket Systems als Ersatz für bestehende lokale Konfigurations-managementdatenbanken

Wenzhe Xia

INSTITUT FÜR INFORMATIK
DER LUDWIG–MAXIMILIANS–UNIVERSITÄT MÜNCHEN

Projektarbeit

Evaluation des TUM-Trouble Ticket Systems als Ersatz für bestehende lokale Konfigurations-managementdatenbanken

Wenzhe Xia

Aufgabensteller:	Prof. Dr. Heinz-Gerd Hegering
Betreuer:	Dr. Wolfgang Hommel Silvia Knittl Dr. Josef Homolka
Abgabetermin:	20. Januar 2010

Hiermit versichere ich, dass ich die vorliegende Praktikumsarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.
München, den 19. Januar 2010

.....

(Unterschrift des Kandidaten)

Abstract

IT-Service-Management (ITSM) bezeichnet die gesamt nötigen Maßnahmen und Methoden, um die Geschäftsprozessen durch die IT-Organisation zu untersützen. Die IT-Infrastructure-Library (ITIL) ist ein weltweiter De-facto-Standard im Bereich ITSM. Er beinhaltet eine umfassende Dokumentation zur Planung, Erbringung und Unterstützung von IT-Services. ITIL beschreibt in einem Rahmen, welche Prozesse eingeführt werden sollen und die Prinzipien und Vorschläge, wie diese Prozesse realisiert werden können. Der Kernprozess in ITIL ist Configuration-Management (CM). Alle anderen Prozesse, wie Change-Management und Incident-Management, werden basierend auf CM entwickelt. Die Kernkomponente von CM ist die Configuration-Management-Database (CMDB). Laut ITIL handelt es sich bei der CMDB um eine Datenbank, welche dem Zugriff und der Verwaltung von Configuration-Items (CIs) und Beziehungen dient.
An der Technischen Universität München (TUM) wurden schon einige Projekte nach ITIL-Konzepten eingeführt. Am Leibniz-Rechenzentrum (LRZ) wurde ein Werkzeug namens die Netzdoku zum Zweck der Dokumentation des Münchner Wissenschaftsnetzes (MWN), Hilfe bei Fehlersuche, Dokumentation der Informationen von Organisationen und relevanten Netzverantwortlichen (NV) selbst entwickelt. In der Fakultät für Physik an der TUM gibt es ein eignenes Konfigurationssystem, das zur Verwaltung der lokalen Netzkomponenten und Netztopologie dient. Gesucht wird allerdings eine gemeinsame CMDB, in die die Daten aus verschienden dezentralen Datenquellen integriert werden können. In dieser Arbeit wird vor allem eine geeignete Struktur für diese CMDB definiert.
Ein Prozess in ITIL ist das Incident-Management, welches das Ziel hat, die Unterbrechung von Geschäftsprozessen im Falle von Störungen für die Benutzer so gering wie möglich zu halten. An der TUM wird das Incident-Management durch das Werkzeug Open source Ticket Request System (OTRS) unterstützt. Jetzt gibt es von diesem Werkzeug eine ITSM-Lösung für das CM. Wir untersuchen, ob diese Lösung für unsere IT-Umgebung geeignet ist.
In dieser Arbeit wird weiterhin die Theorie des ITIL-Prozesses CM vorgestellt. Im Rahmen des CMs wird zuerst das ideale Datenbankmodell entworfen, damit die Daten aus lokalen CMDB zusammengefasst und intergriert werden können. Anschliessend wird die ideale Datenbankstruktur in OTRS abgebildet. Danach entwickeln wir einen Integerationsprozess, der die Daten aus den bestehenden Konfigurationsdatenbanken sammelt und in die OTRS Datenbank integriert. Am Ende wird evaluiert, ob die ITSM-Erweiterung von OTRS für unsere IT-Umgebung geeignet ist. Zusätzlich wird noch abgeschätzt, welche Weiterentwicklungen noch durchgeführt werden sollten.

Inhaltsverzeichnis

1 Einleitung **1**
- 1.1 Beschreibung des Szenarios . . . 1
  - 1.1.1 IT-Ressourcen und Services der Technische Universität München am Beispiel der Fakultät für Physik . . . 2
  - 1.1.2 Das Leibniz-Rechenzentrum (LRZ) . . . 2
  - 1.1.3 Das Münchner Wissenschaftsnetz (MWN) . . . 3
- 1.2 Problemstellung . . . 3
- 1.3 Aktuelle Lösungen zur Verwaltung von Konfigurationsdaten . . . 5
  - 1.3.1 Netzdoku . . . 5
  - 1.3.2 Konfigurationssystem der Physikfakultät . . . 7
  - 1.3.3 Das Open source Ticket Request System . . . 7
  - 1.3.4 Ablauf der Arbeit . . . 8

2 Configuration-Management Grundlagen **11**
- 2.1 ITSM . . . 11
- 2.2 ITIL . . . 12
- 2.3 Configuration-Management . . . 15
  - 2.3.1 CM Überblick . . . 15
  - 2.3.2 Theoritische Grundlage von CM . . . 16
  - 2.3.3 Configuration-Management-Database . . . 18
- 2.4 Zusammenfassung . . . 21

3 CMDB Planung **23**
- 3.1 Schritt 1: Anforderungsammlung . . . 23
- 3.2 Schritt 2: Anforderungsanalyse . . . 24
  - 3.2.1 Organisatorische Informationen . . . 25
  - 3.2.2 Netzbezügliche Resourcen . . . 25
  - 3.2.3 Übrige Resourcen . . . 26
  - 3.2.4 Zusammenfassung der Anforderungsanalyse . . . 27
- 3.3 Schritt 3: CMDB Analyse . . . 27
  - 3.3.1 Methodik . . . 27
  - 3.3.2 CIs und deren Eigenschaften . . . 29
  - 3.3.3 Beziehungen . . . 29
- 3.4 Schritt 4: CMDB-Definition . . . 30
  - 3.4.1 Methodik . . . 31
  - 3.4.2 Zusammenfassung . . . 31

4 CMDB Implementierung **33**
- 4.1 Überblick über die ITSM Lösung von OTRS . . . 33
- 4.2 CM-Pakete Installation . . . 33

4.3 CI-Verwaltung in OTRS . . . 35
4.3.1 CIs und Beziehungen . . . 37
4.3.2 CI Suche . . . 38
4.4 Datenaustausch zwischen OTRS und bestehenden Datenbeständen . . . 38
4.4.1 Grundidee . . . 39
4.4.2 Schema-Definition . . . 39
4.4.3 Synchronisationsvorgang: Export aus der Netzdoku . . . 45
4.4.4 Synchronisationsvorgang: Export aus Physik . . . 53
4.4.5 Synchronisationsvorgang: Import ins OTRS . . . 54
4.5 Zusammenfassung . . . 59

5 Zusammenfassung **61**
5.1 Evaluation . . . 61
5.2 Weiterentwicklungen . . . 63
5.3 Zusammenfassung . . . 64

Abbildungsverzeichnis **67**

Literaturverzeichnis **69**

1 Einleitung

Bevor wir über das Thema Configuration-Management in der „Information Technology Infrastructure Library“ (ITIL) [AC07a] diskutieren, schauen wir zuerst ein konkretes Beispiel an:
Der Administrator der Physikfakultät ist zuständig für die IT Abteilung der Physikfakultät der Technischen Universität München (TUM). Seine Aufgabe besteht darin, alle technischen Probleme bzgl. Rechner, Software oder Netz zu lösen und Webinhalte von lokalen Webanwedungen zu pflegen. Heute ist das für ihn besonders schwer. Er bekommt ständig E-Mails von Professoren, dass die Übungen vor 12 Uhr im Webauftritt zur Verfügung stehen sollen. Dazwischen können zwei Kollegen sich nicht ins Internet verbinden. Schlimmer ist, dass der Mail-Server innerhalb der Fakultät nicht erreichbar ist. Was soll er zuerst machen? Wie kann er alle Problem lösen? Ein einfaches Neustarten des Mail-Servers kann den Service in Ordnung bringen, aber es ist keine gute Lösung. Man muss noch untersuchen, worin das Problem liegt, damit eine Lösung schneller gefunden wird, wenn das Problem noch einmal vorkommt.
Heutezutage werden immer mehr neue Technologien, Methoden und Ideen in den Firmen und Organisationen eingesetzt und die Anforderungen an die Verwaltung von IT-Resourcen und IT-Services wachsen. Leider konzentrieren sich die IT-Abteilungen oft auf die Technik, wie ein konkretes Problem gelöst werden soll. Sie merken aber oft nicht, dass eine effiziente Verwaltung der Resourcen und Services durch eine struktuelle Problemlösung erreicht werden kann. Eine mögliche Lösung bietet ITIL an. ITIL beschreibt eine Reihe von Prozessen, die durch praktische IT-Service-Management (ITSM) Realisierungen von verschiedenen Organisationen erfasst wurden und zur guten Verwaltung der IT-Resourcen hilfsreich sind. Mit ITIL kann der Administrator von täglichem Chaos gerettet werden.
Da die vollständige Implementierung aller ITIL Prozesse in unserer IT-Umgebung eine systematische und umfangreiche Arbeit ist, konzentrieren wir uns in dieser Arbeit nur auf die sogenannte Configuration-Management-Database (CMDB). Eine ideale CMDB Struktur wird aufgebaut, die für das Speichern unserer IT-Resourcen geeignet ist. Grundlage sind die Anforderungen der Netz- und Systemverwaltung der Fakultät der Physik der TUM und die CMDB der sogenannten Netzdoku von Leibniz-Rechenzentrum (LRZ). Die Daten werden aus diesen verschiedenen Datenquellen gesammelt.

1.1 Beschreibung des Szenarios

Ein wichtiges Ziel des Configuration-Managements (CM) ist es, die IT-Resourcen innerhalb der Organisationen effzient zu verwalten. Bevor wir das CM einführen können, müssen wir den Umfang dieses Vorhabens festlegen, damit wir die entsprechenden Resourcen und Services erfassen können. Die IT-Resourcen und IT-Services, die von dem CM erfasst werden sollen, sind hauptsächlich über die folgenden Organisationen, Institute und Netze verteilt: das Münchner Wissenschaftsnetz (MWN), das LRZ und die Technische Universität München mit ihren Fakultäten. Wir betrachten hier nicht alle Fakultäten der TUM, sondern schauen

wir die Fakultät Physik als Beispiel an. Hier werden die beteiligten Organisationen einzeln vorgestellt.

1.1.1 IT-Ressourcen und Services der Technische Universität München am Beispiel der Fakultät für Physik

Wie die Abbildung 1.1 zeigt, erstreckt sich das MWN auf das LRZ und über die Hochschulen. Das Netz wird vom LRZ betrieben. Darunter hängen noch die lokale Netze auf der zweiten Stufe. Diese lokalen Rechnernetze sind Erweiterungen von dem Kernnetz und befinden sich normalerweise in den Gebäuden von verschiedenen Fakultäten und Instituten.
Obwohl das LRZ allen Fakultäten die Netzverantwortlichen (NV) und Ansprechspartner anbietet, stellen die Fakultäten eigene Administratoren ein. Diese Administratoren sind zuständig dafür, die lokale Netze zu überwachen und zu betreiben, die EDV-Fragen der Mitarbeiter zu beantworten, die IT-Resourcen wie IP-Adresse, Mailaccount und verschiedene Kennungen zuzuteilen. Es gibt eine enge Zusammenarbeit mit den NV, bestimmte Anfragen werden dort bearbeitet. In dieser Arbeit konzentrieren wir uns examplarisch bei den Fakultäten der TUM auf die Physik.
Die IT-Resourcen, die in Fakultäten erhalten sind, werden erfasst als:

- Personen, die für die lokale Netzerweiterungen zuständig sind.
- Die Netzkomponenten, aus denen das lokale Netz besteht.
- Lokale IT-Diestleistungen, wie E-Mail Server, Web Server.
- Hardware wie PCs, Drucker, Patches usw.

1.1.2 Das Leibniz-Rechenzentrum (LRZ)

Das LRZ ist zuständig dafür, das Kernnetz des MWNs zu planen, auszubauen und zu betreiben. Das LRZ ist vor allem zuständig für das gesamte Backbonenetz (siehe Abbildung 1.1) und die angeschlossenen Institutsnetze. Eine Ausnahme ist, dass die medizinischen Fakultäten der LMU und TUM, die FH München und die Fakultät der Informatik der LMU die Rechnernetze lokal betreiben und betreuen, aber das LRZ ist jedoch für die Anbindung dieser Netze an das MWN zuständig. Zur Administration der Netze stellt das LRZ die Netzverantwortlichen für die Einrichtungen bereit. Die NV verwalten die zugeteilten Namens- und Adressräume und dokumentieren die an das MWN angeschlossenen Endgeräte bzw. Netze. Außerdem arbeiten sie zusammen mit Mitarbeitern von Fakultäten, um die Netzerweiterungen in lokalen Gebäudenetzen zu planen und aufzubauen.
Außer der Verwaltung- und Betriebsarbeit liefert das LRZ den Kunden noch viele IT-Dienste wie E-Mail, DNS, Datenbackup usw. Die schnell wachsende Nutzerschaft des MWNs erfordet immer mehr Fragen und Wissensbedarf über die Netze und Dienste. Aus diesem Grund bietet das LRZ fachliche Beratung an. Dazu gehören ein umfangreiches Kursangebot, individuelle Beratung und ausführliche Anleitungen. Über das LRZ können die Kunden auch Software mit Lizenzen beziehen. Die IT-Resourcen, die vom LRZ erfassen werden, sind folgend:

- Die Organisationsinformationen, wie z.B. Forschungsgruppe, Abteilungen und Labors.

- Die Informationen von den Personen, die am LRZ arbeiten. Insbesondere die Netzverantwortlichen.
- Die IT-Dienstleistungen, die vom LRZ angeboten werden, wie z.B. Name-, E-Mail-, Webserver und Speicher-, Backupgeräte.
- Software und Lizenzen.

1.1.3 Das Münchner Wissenschaftsnetz (MWN)

Das MWN ist eine leistungsfähige Kommunikationsinfrastruktur, die die dezentralen Rechner im Münchner Wissenschaftsraum mit einander verbindet und Zugang zu weiteren Netzen anbietet. Durch das MWN können die Studenten der Münchner Hochschulen wie TUM, LMU, Angehörige der Bayerischen Akademie der Wissenschaften, Fachhochschule Weihenstephan, die Professoren und Mitarbeiter aus wissenschaftlichen Einrichtungen wie dem Max-Planck-Institut oder die Bewohner der Studentenwohnheime das Internet benutzen. Das MWN umfasst vor allem zahlreichen Netzkomponenten wie z.B. Router, Switche, die über 65000 Systeme wie Server, Rechner, Drucker usw. verbinden. Zum Netzaufbau gehören auch die physischen und logischen Verbindungskanäle wie verschiedene Kabel, Glasfaser usw. Die folgende Abbildung zeigt uns eine grobe Netztopologie des MWNs.

Wie die Abbildung 1.1 zeigt, befindet sich das Kernnetz des MWNs hauptsächlich in den Gebäuden des LRZ und der Hochschulen. Die IT-Resourcen, die im MWN enthalten sind, können wir wie folgt erfassen:

- Standorte, Gebäude und Räume, wo die Netzkomponenten sich befinden.
- Die Netzkomponenten, aus denen das MWN besteht.
- Netztopologie sowie physische Verbindungen zwischen Netzkomponenten.
- Unterschiedliche Systeme wie Server, PCs usw, die zum MWN gehören.

Während das MWN die physische Infrastruktur bezeichnet, übernimmt dessen Betriebs- und Verwaltungsarbeit das LRZ.

1.2 Problemstellung

Wie in den oberen Abschnitten vorgestellt, gibt es zahlreiche und umfangreiche IT-Resourcen und IT-Dienstleisungen in den beteiligten Organisationen zu verwalten. Im Prinzip sollen alle Daten von solchen Resourcen vorhanden und jederzeit auswertbar sein, damit im MWN und dessen Erweiterungen ein effktiver und gesicherter Betriebszustand garantiert werden und die tägliche Arbeit der Organisationen unterstützt wird. Wünschenswert wäre eine zentrale Verwaltung der Konfigurationsdaten, die als Basis für bestehende Prozesse wie z.B. Incident-Management und weitere zu entwickelnde Prozesse steht. Die zentrale Verwaltung der Daten ist erforderlich, weil die Daten zur Unterstüzung des Netzbetriebs insbesondere hilfsreich sind. Wir schauen hier ein Beispielszenario an:
Der Administrator der Physikfakultät bekommt eine Anfrage, dass ein Virtual Local Area

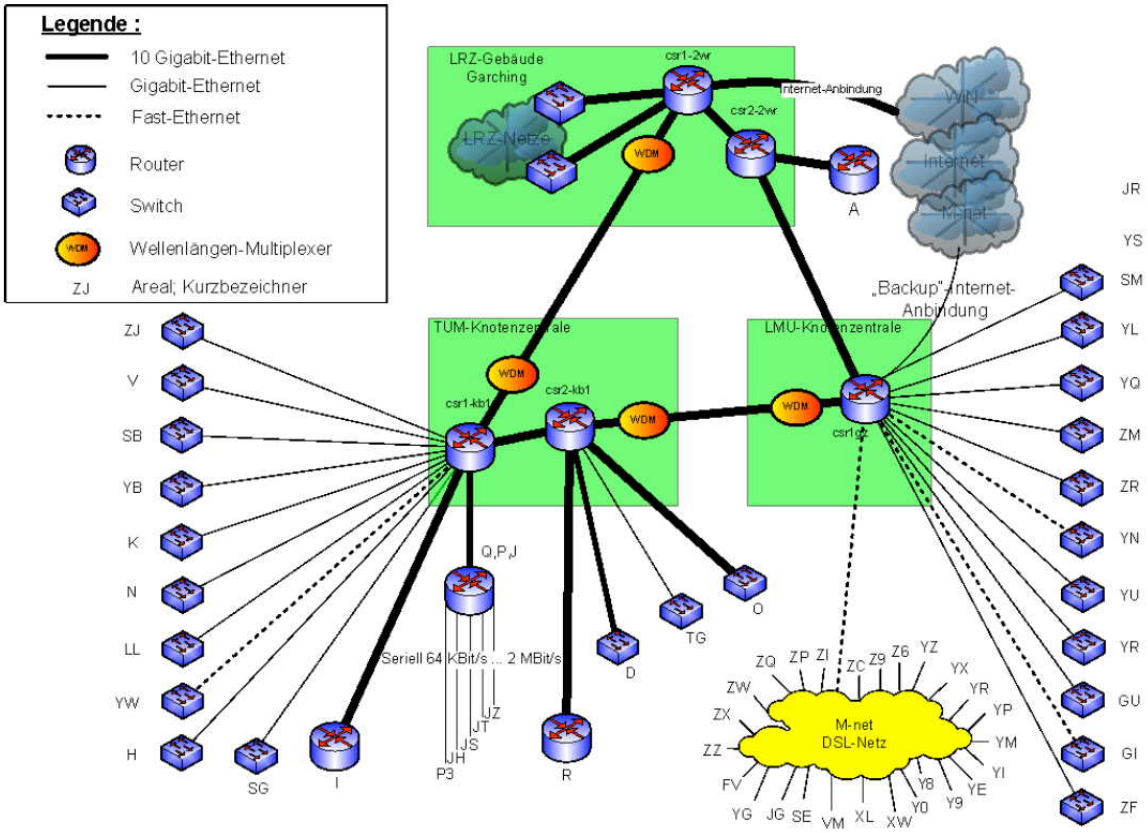


Abbildung 1.1: MWN Backbone
[Bila]

Network (VLAN) zwischen Raum 101, 218 und 310 eingerichtet werden soll. Um diese Aufgabe zu erledigen, muss die Netztopologie für den Admin klar sein. Die drei Räume besitzen jeweilig 4 bis 12 Netzdosen. Manche Dosen sind inaktiv und werden nie benutzt. Die aktiven Dosen werden weiter mit Patches verbunden, die zur Zentralverwaltung der Dosen eingeführt werden. Die Ports von Patches werden noch weiter mit den Ports von Switches verbunden. Die Einrichtung von dem VLAN wird auf die Ports von den Switches durchgeführt. Um die Aufgabe zu erledigen erwartet der Admin bzw. der NV folgende Informationen:

- Welche Netzdosen haben die Räume?
- Wie werden die Patches mit Switches verbunden?
- Wie werden die Dosen mit Patches verbunden?
- Welche Ports von welchen Switches sind relevant?
- Wer kann die Einrichtungsarbeit übernehmen?
- Weiterhin zur Überwachung und Sicherung des VLANs muss man noch die geographische Konfigurationsdaten haben:
  1. Wo befinden sich die Räume?
  2. Wo befinden sich die Patches?
  3. Wo befinden sich die Switche?
  4. Wer ist für die Switche zuständig?

Ohne eine CMDB muss der Administrator aus verschiedenen Quellen diese Informationen holen. Er muss beispielsweise in seiner Inventarliste schauen, welche Netzdosen die Räumen haben. Er muss in der lokalen Datenbank schauen, wie die Netztopologie aufgebaut ist. Er muss dann die Netzverantwortlichen fragen, wie die Netztopologie ist und ob er diese Aufgabe alleine erledigen kann. Falls nicht, muss er sich noch an die entsprechenden Personen wenden. Es wird noch ungelücklicher, falls er keinen Netzverantwortlichen kennt. Falls wir eine Konfigurationsdatenbank hätten und diese Datenbank alle diese benötigten Informationen enthielte, kann die Aufgabe einfach erledigt werden.

1.3 Aktuelle Lösungen zur Verwaltung von Konfigurationsdaten

Zur Lösung obiger Problemstellung, sind innerhalb des LRZ und an der TUM schon einige Konfigurationssysteme verfügbar. Beispiele davon sind Netzdokumentation des LRZ (Netzdoku), Konfigurationswebanwendung der Physikfakultät und das Open source Ticket Request System (OTRS) der TUM. Diese werden nachfolgend vorgestellt.

1.3.1 Netzdoku

Da das LRZ für den Betrieb des MWNs zustandig ist, wurde die Netzdoku vom LRZ entwickelt zum Zweck der Dokumentation des MWNs, Hilfe bei Fehlersuche, Dokumentation der Informationen von Organisationen und relevanten NVs. Die Netzdoku ist ein dreistufiges System, das aus Datenbank, Applikationsserver und Webinterface besteht. In der Datenbank

der Netzdoku werden folgende Informationen dokumentiert:

- Netzkomponenten (Router, Switche, Server usw.)
- Logische und physische Ports von Netzkomponenten.
- Verbindungen zwischen Komponenten bzw zwischen Ports.
- Subnetze und VLANs.
- Personen, die für obengenannte Komponenten zuständig sind.
- Lagerinformationen von Komponenten.

Die folgende Abbildung 1.2 zeigt uns die Informationen von dem Router csr2-2wr, der das MWN-Hauptnetz im LRZ und die Netzteile auf dem Campus Garching verbindet:

Abbildung 1.2: Beispiel 1 aus der Netzdoku
[Bilb]

Vom Suchergebniss der Netzdoku wissen wir, dass der Router als csr2-2wr bezeichnet wird. Hier wird das Cisco Produkt Catalyst 6509 eingesetzt. Dieser Router liegt im LRZ Hauptgebäude Raum R2.02.02. Auf dem Port 1/1 von dem Router sind mehrere VLANs eingerichtet. Die folgende Abbildung zeigt 20 VLANs an, an denen der Router csr2-2wr beteiligt.

Dem Port 1/1 von csr2 wird die IP Adresse 129.187.1.6 zugeordnet und dem Port 7/1 von csr1 die IP Adresse 129.187.1.5. Die zwei Ports befinden sich in einem Subnetz. Die Netzdoku ist derzeit die größte Infomationsquelle für unser zu entwickelndes CM System.

1.3.2 Konfigurationssystem der Physikfakultät

Ähnlich wie die Netzdoku am LRZ, wurde von der Fakultät Physik an der TUM ein eignenes Konfigurationssystem mit einer Datenbank entwickelt. Dieses Konfigurationssystem beinhaltet folgende Informationen:

- Lokale Netzkomponenten (Netzdosen, Patches, Ports von Patches)
- Lokale Netztopologie (Verbindungen zwischen Switches, Patches und Dosen)
- Ortsinformationen der Komponenten.
- Räume, Lehrstühle und die Zuordnung von Räumen und Lehrstühlen.
- Systemadministratoren, die für verschiedene Lehrstühle verwantwortlich sind.

Basierend auf dieser lokalen CMDB wurde eine eigene Webanwendungen entwickelt. Die folgende Abbildung zeigt uns ein Webinterface an, das die aktiven Netzdosen in Gruppe E15, Raum 3510 anzeigt. Die Abbildung 1.3 zeigt, dass der Raum 3510 vier aktive Netzdosen enthält. Die erste wird mit der Nummer 1502-05-14 bezeichnet, wobei 1502 den Patchraum, 05 die Panelnummer und 14 die Portnummer bezeichnet.

Netzwerkraum:	1502	erste 4 Ziffern auf Netzdose
Raum:	3510	Raumnummer, kann regulärer Ausdruck sein, z.B ^11.*$ für alle Raumnummern die mit 11 beginnen
Gruppe:	E15	Lehrstuhl, Dekanat (av), Verwaltung(zv), andere zentrale Einrichtung
Status:	aktiv	none = unbekannt oder aktiv/inaktiv

Suchen

Netzanschluss	Raum	Gruppe	Status
1502 05--14	3510	e15	aktiv
1502 05--15	3510	e15	aktiv
1502 05--16	3510	e15	aktiv
1502 05--17	3510	e15	aktiv

Abbildung 1.3: Beispiel von Physik Webanwendung
[Bilb]

1.3.3 Das Open source Ticket Request System

Da sich die oben genannten beiden Systeme auf die konkreten zu verwaltenden Komponenten konzentrieren, wurde OTRS als ein Ticketsystem an der TUM eingesetzt [Här07].
Das OTRS ist eine Webapplikation, die mit jedem HTML kompatiblen Browser benutzt werden kann. Das OTRS ist vor allem ein Ticketsystem auf open source Basis, welches

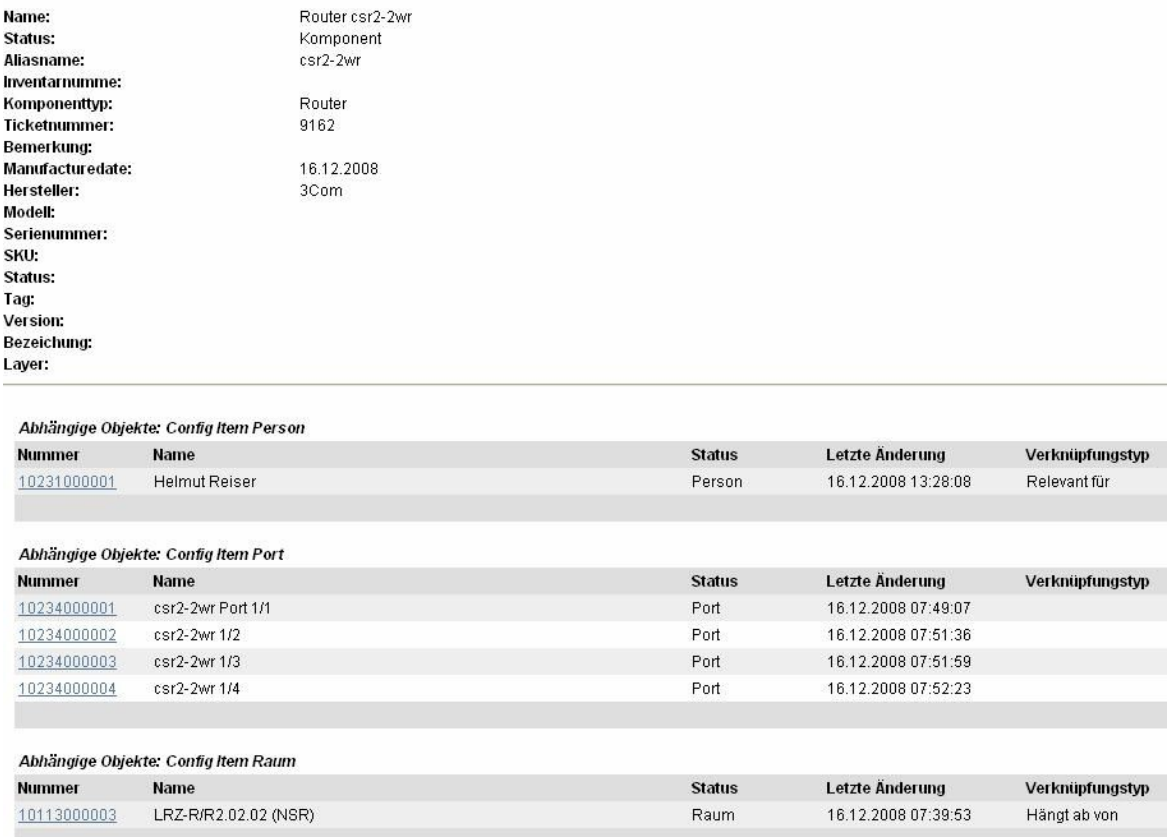
es erlaubt, Benutzeranfragen kanalisiert zu bearbeiten, auch wenn die Supportmitarbeiter in verschiedenen Organisationseinheiten angesiedelt sind. Mittlerweile gibt es eine ITSM Lösung von diesem Werkzeug. Diese Lösung unterstützt vor allem drei wesentliche Prozesse nach dem ITIL Standard: Incident-Management, Problem-Management und Configuration-Management. Ein Schwerpunkt dieser Arbeit ist herauszufinden, ob das von OTRS angebotene Configuration-Management unseren CM Ansprüchen genügt. Das CM von OTRS basiert auf einer integrierten CMDB, die den Grundstein der ITSM Prozesse darstellt. Die CMDB bildet sowohl die Configuration-Items (CIs) als auch deren komplexen Beziehungen und Abhängigkeiten untereinander bzw. innerhalb der Service-Ketten ab.
Die weiteren Eigenschaften von OTRS CM sind:

- Die Typen von CIs, die Eigenschaften von CIs und die Beziehungen zwischen CIs kann man flexibel definieren und konfigurieren.
- Der IT-Service-Katalog und geltende Vereinbarungen wie Service level Areement (SLA), Operational Level Agreement (OLA), Underprinning Contract (UC) werden von OTRS abgebildet.
- Management historischer, aktueller und zukünftiger CI-Status.
- Chronologisches Life-Cycle-Management für CIs, von Eingang bis Entsorgung.
- Protokollierung aller Konfigurationsänderungen an CMDB-Daten.

Die folgende Abbildung 1.4 zeigt uns, wie der Router csr-2wr in OTRS CM dargestellt wird:

1.3.4 Ablauf der Arbeit

Der Schwerpunkt dieser Arbeit ist zuerst ein ideales Datenbankmodell zu entwerfen, damit die Daten aus lokalen CMDB in der idealen CMDB intergriert werden. Es wird nicht die komplette Infrastruktur der TUM und des LRZ betrachtet, sondern konzentrieren wir auf das oben beschriebene Szenario. Dieses ideale Datenbankmodell wird basierend auf den Anforderungen in diesem Szenario entwickelt. Anschliessend wird eine Evaluationsmethode entwickelt und mit Hilfe dieser Methode wird evaluiert, ob die ITSM-Erweiterung von OTRS für unsere IT-Umgebung geeignet ist. Da ITSM von OTRS nach dem ITIL Standard aufgebaut wird, werfen wir zuerst in Abschnitt 2 einen Blick auf ITIL. Der ITIL Standard besteht aus zwei Prozessengruppen und jede Gruppe besteht weiterhin aus mehreren Prozessen. Die Einführung von allen ITIL Prozessen ist eine aufwendige und langfristige Arbeit. Im Rahmen dieser Arbeit konzentrieren wir uns auf den ITIL CM Prozess. Durch die Schritt für Schritt Implementierung des CMs wird es klar, welche Anforderungen von OTRS erfüllt werden sollen (siehe 3.1). Wenn das OTRS endlich als geeignet bewertet wird, wird im Abschnitt 4 ein Prototyp aufgebaut, der alle bestehende Daten integriert. Basierend auf diesem Prototyp wird weiterhin abgeschätzt, welche weiteren Änderungen eingeführt werden sollen.



Name:	Router csr2-2wr
Status:	Komponent
Aliasname:	csr2-2wr
Inventarnumme:	
Komponenttyp:	Router
Ticketnummer:	9162
Bemerkung:	
Manufacturedate:	16.12.2008
Hersteller:	3Com
Modell:	
Seriennummer:	
SKU:	
Status:	
Tag:	
Version:	
Bezeichung:	
Layer:	

Abhängige Objekte: Config Item Person

Nummer	Name	Status	Letzte Änderung	Verknüpfungstyp
10231000001	Helmut Reiser	Person	16.12.2008 13:28:08	Relevant für

Abhängige Objekte: Config Item Port

Nummer	Name	Status	Letzte Änderung	Verknüpfungstyp
10234000001	csr2-2wr Port 1/1	Port	16.12.2008 07:49:07	
10234000002	csr2-2wr 1/2	Port	16.12.2008 07:51:36	
10234000003	csr2-2wr 1/3	Port	16.12.2008 07:51:59	
10234000004	csr2-2wr 1/4	Port	16.12.2008 07:52:23	

Abhängige Objekte: Config Item Raum

Nummer	Name	Status	Letzte Änderung	Verknüpfungstyp
10113000003	LRZ-R/R2.02.02 (NSR)	Raum	16.12.2008 07:39:53	Hängt ab von

Abbildung 1.4: Beispiel Anzeige Router in OTRS

2 Configuration-Management Grundlagen

In diesem Kapitel werden die Begriffe wie ITSM und ITIL vorgestellt. Danach schauen wir im Rahmen von ITIL die technische Grundlage von CM an. Die folgende Fragen sind für uns interessant:

- Was ist Configuration-Management?
- Welche Vorteile kann CM erbringen?
- Was ist eine CMDB?
- Welche Struktur kann eine CMDB haben?
- Wie kann eine CMDB geplant und realisert werden?
- Wie kann man CM in einer IT-Umgebung realisieren?

2.1 ITSM

Die ITIL (Version 3) definiert Service und Service-Management wie folgt: [AC07a]

Service A service is a means of delivering value to customers by facilitating outcomes customers want to achieve without the ownership of specific costs and risks.

Service-Management Service-Management is a set of specialized organizational capabilities for providing value to customers in the form of services.[AC07a]

Beim Service gibt es zwei Rollen: der Servicelieferant und Serviceempfänger. IT-Services sind die Services in einer IT-Umgebung. Ein einfaches Beispiel ist der E-Mail-Service. Einfacher ausgedrückt ist das Service-Management die Methode, mit denen die Services geliefert werden.
Um IT-Services zweckmäßig und effektiv zu kontrollieren und zu steuern, ist das Thema IT-Service-Management gefragt. ITSM bezeichnet die Gesamtheit von Maßnahmen und Methoden, die nötig sind, um die bestmögliche Unterstützng von Geschäftsprozessen durch die IT-Organisation zu erreichen[wiki]. ITSM ist eine Kombination von drei Faktoren: Personen, Prozesse und Technik. Im Allgemeinen ist ein Prozess eine Folge von logischen zusammenhängenden Aktivitäten, die einen IT-Service erstellen. Vor ITSM konzentrierte man sich meistens auf die Technik. Man dachte hier an die Methoden, mit denen ein technisches Problem gelöst oder eine Anforderung erfüllt wird. Unter der Idee von ITSM ist der Schwerpunkt nicht mehr die technische Methoden, sondern IT-Service selbst. Die Technik ist hierbei nur der wichtige Hintergrund. [AC07a]

2.2 ITIL

Aufgrund der oben beschriebenen Situation wurde ITIL entwickelt, um ITSM sinvoll und erfolgreich durchzuführen. ITIL wurde erst um 1989 im Auftrag der britischer Regierung entwickelt und ist dann weltweit ein Standard geworden. ITIL besteht aus einer Reihe von Dokumentationen und beschreibt verschiedene Prozesse, die zur Einführung des ITSMs notwending sind. ITIL ist ein sogenanntes Best-Practices-Framework. Es wurde in der Einführung des ITSMs in verschiedenen Organisationen aus der Praxis erfasst und verfeinert. ITIL liefert keine Regeln, wie man Schritt für Schritt IT-Services verwalten kann, sondern beschreibt einen Rahmen, welche Prozesse eingeführt werden sollen und die Prinzipien und Vorschläge, wie diese Prozesse realisiert werden. Die IT-Prozesse müssen sich an den Geschäftsprozessen orientieren. ITIL liefert Vorgaben und Anregungen, mit deren Hilfe das ITSM an neue Anforderungen effizienter angepasst werden kann. [AC07a]

Wie oben vorgestellt, wird ITIL durch das Feedback aus der Praxis stetig bessert. In dieser Library befinden sich viele Dokumentationen, die unterschiedliche Prozesse beschreiben und von verschiedenen Firmen publiziert werden. Fünf davon wurden ausgewählt und als Standard konsolidiert. Diese fünf Dokumentationen entsprechen genau den fünf Stufen des Lebenszyklus von einer ITIL Realisierung.

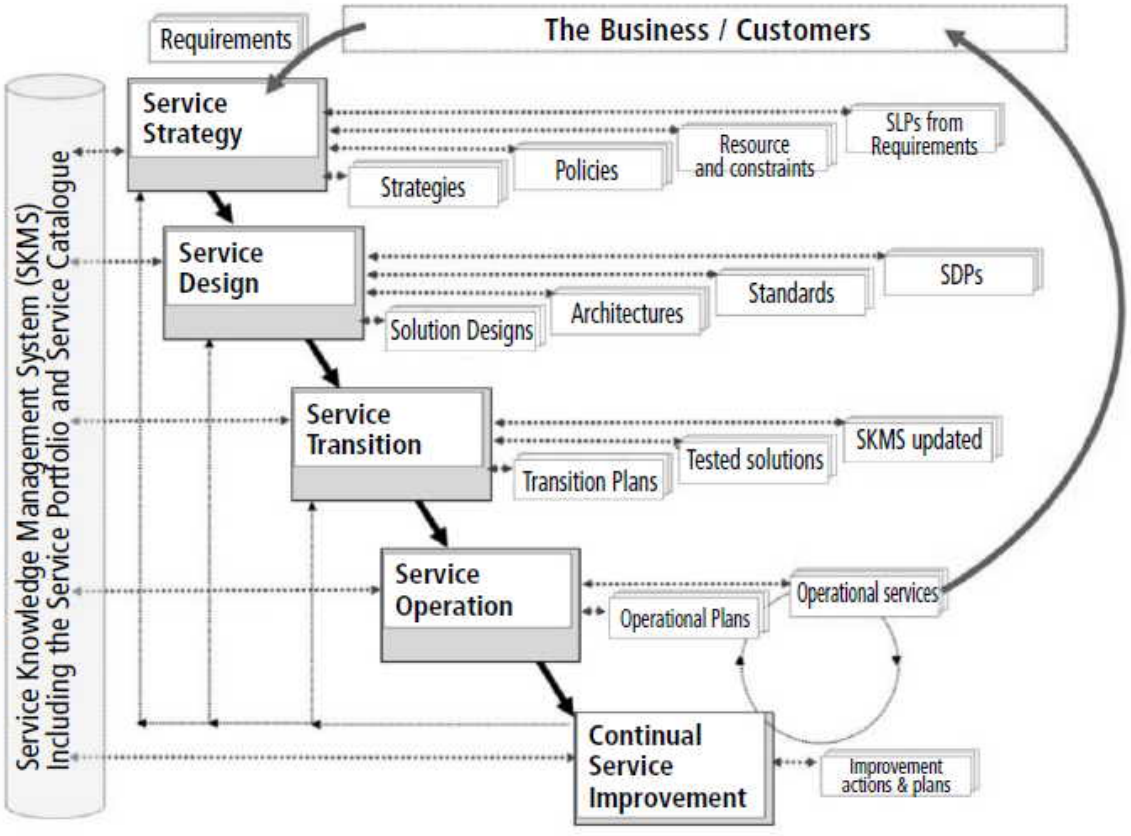


Abbildung 2.1: ITIL Lebenszyklus nach [AC07a]

Wie die Abbildung 2.1 darstellt, sind die fünf Kernstufen Service Strategy, Service Design,

Service Transition, Service Operation und Continual Service Improvement. In der Phase Service Strategy werden hauptsächlich zwei Fragen beantwortet: wer ist der Serviceempfänger und was ist genau der Service. Die Anforderungen von Serviceempfängern werden aus realen Umgebung zusammengefasst.
Die Aufgabe in der Phase Service Design ist es, eine Lösung herauszufinden, zu planen und beschreiben. Die vier wichtigesten Aspekte sind Architektur, Prozesse, Realisierungsstrategie, und Änderungs- und Verbesserungsmöglichkeiten des Services.
Im Schritt Service Transition soll die in Service Design entwickelte Lösung in realen Umgebungen getestet werden. Erfüllt der entworfene Service alle Anforderungen? Kann der Service in der Umgebung angepasst werden? Nicht nur in einer normalen Umgebung sondern auch in einer außergewöhnlichen. Beispielsweise kann man per PC und Internet den E-Mail Service benutzen, aber durch ein Mobiltelephon ist er derzeit nicht erreichbar.
In der Phase Service Operation werden die geplanten und getesteten Services mit der Implementierung kombiniert. Aus der Idee werden sichtbare Produkte den Kunden bereitsgestellt und die Rückmeldung von Kunden werden gesammelt und als Argument der Serviceverbesserung geliefert. In der letzten Phase werden die Services verbessert und wahrscheinlich muss man den Zyklus von Schritt 1 an nochmals durchlaufen. Von den vielen Prozessen und Methoden, die von ITIL zusammengefasst werden, werden diejenigen, die in folgender Abbildung vorgestellt werden, als die wichtigesten bezeichnet.

Wie die Abbildung 2.2 zeigt, definiert die Spezifikation „Service Strategy“ folgende Kernprozesse: Financial-Management, Service-Portfolio-Management (SPM) und Demand Management. Financial-Management kümmert sich um die Kostenkontrolle. Über SPM definiert man die Verwaltungsmethoden, wodurch die Risiken abgeschätzt und bewertet werden. Ziel des Demand-Managements ist es, die Anforderungen der Kunden zu verstehen, damit man einen Überblick über den Umfang von Services hat. [AC07e]
Service Design beschreibt die Prinzipien und die Methoden, wie man einen passenden und zielführenden Service-Management Prozess plant und entwickelt. Die folgenden Prozesse werden von ITIL in diesem Schritt betrachtet: Service-Catalogue-Management (SCM), Service-Level-Management (SLM), Capacity-Management, Availability-Management, ITService-Continuity-Management (ITSCM), Information-Security-Management (ISM) und Supplier-Management.
Das SLM dient zur Definition, Überwachung und Optimierung von IT-Services. Primäre Zielsetzung dabei ist, dauerhaft die Leistung der Services im Einklang mit den geschäftlichen Erwartungen über die Formulierung der sogenannten „Service Level Objectives (SLO)“ zu bringen. Das SLM trägt die Verantwortung für die Effizienz der Prozessbeziehungen zwischen dem Geschäfts- und dem IT-Management. Das Rahmenwerk der Prozesse wird dabei in Form von Vereinbarungen, Verbindlichkeiten und Beziehungen beschrieben, welche über den sogenannten Service Katalog verwaltet werden. Eine wichtige Aufgabe von SLM ist es, die sogenannten Service-Level-Agreements (SLAs) zu definieren. Das SLA bezeichnet die Schnittstelle zwischen Auftraggeber und Auftragnehmer für wiederkehrende Services. Ziel ist es, die Kontrollmöglichkeiten für den Auftraggeber transparent zu machen, indem zugesicherte Leistungseigenschaften wie etwa Leistungsumfang, Reaktionszeit und Schnelligkeit der Bearbeitung genau beschrieben werden. Wichtiger Bestandteil ist hierbei die Servicequalität. OTRS bietet auch eine ITSM Lösung für SLM an. [AC07c]
Service Translation konzentriert sich auf die Transktion von Service zu Operationen. In diesem Schritt werden Services in konkreten Operationen realisiert und dazwischen wird das Risiko kontrolliert. [AC07f]

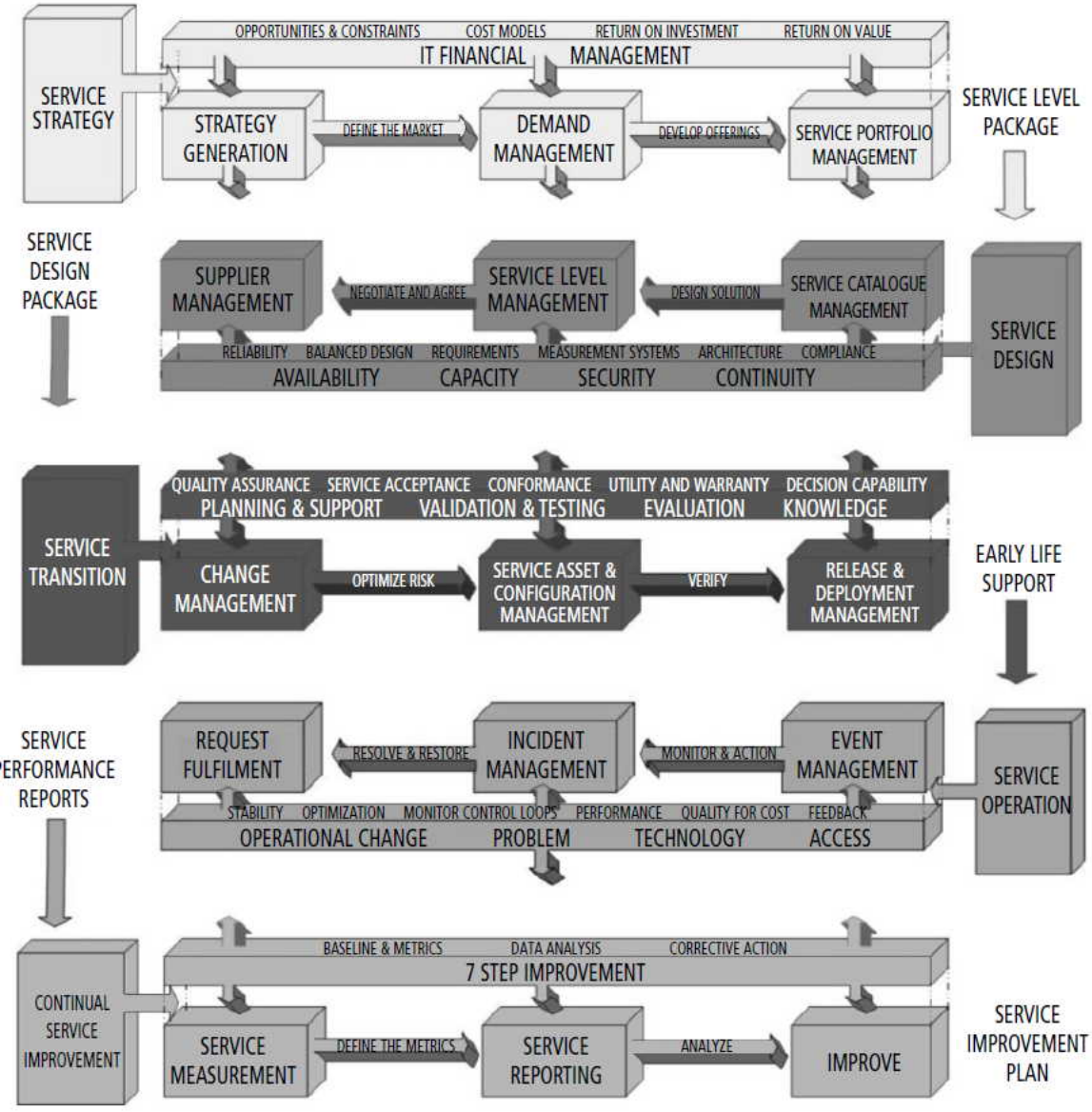


Abbildung 2.2: ITIL Hauptprozesse nach [AC07a]

Service Operation ist eine Implementierungsphase, diese bezieht sich auf die Aufgabe, dass die definierten Services in der Praxis umgesetzt und den Kunden angeboten werden. ITIL erfasst die Prozesse in dieser Stufe im Wesentlichen in zwei Gruppen: Service Delivery und Service Support. Es gibt fünf Basisprozesse, die in Service Support erfasst werden und in jeder ITIL Implementierung eingeführt werden sollen. Die fünf Prozesse heißen: Incident-Management mit Service Desk, Problem-Management, Change-Management, Configuration-Management und Release-Management. Jeder dieser Prozesse ist in sich abgeschlossen und bildet zusammen eine logische Einheit. Diese Prozesse definieren die alltäglichen Funktionalitäten und Aufgaben des IT-Betriebs. In der Prozessgruppe Service Delivery werden weitere ITIL Disziplinen eingeordnet wie: Financial-Management, Availability-Management, Capacity-Management und Continuity-Management [AC07d].

- Incident-Management
  Incident ist die unerwartete Unterbrechung im Lauf der IT-Services. Incident kann die Qualität der IT-Services beeinträchtigen. Ziel des Incident-Managements ist, dass IT-Services schnellst möglich wieder in Ordnung gebracht werden, wenn ein Incident passiert.

- Problem-Management
  Ein Problem kann man als die Ursachen von Incidents auffassen. Zum Problem-Management gehören Incidentdokumentierung, Ursacheuntersuchung, Problemlösung, und Incident nachhaltig verhindern.

- Change-Management
  Eine Change ist die planmäßige Änderung des Systems. Changes sollen dokumentiert werden, damit man in Notfällen ein System wiederherstellen kann. Ziel des Change-Managements ist, dass alle Changes im Configuration-Management System dokumentiert werden und das Systemrisiko maximal reduziert wird.

- Release-Management
  Release bedeutet die Freigabe der neu entwickelten oder geänderten Services. Release-Management kontrolliert den Releasevorgang.

2.3 Configuration-Management

Configuration-Management befindet sich in ITIL Service Operation und Service Support. CM ist der Kernprozess in Service Support. Alle anderen Prozessen nehmen CM als Grundstein und werden basierend auf CM entwickelt.

2.3.1 CM Überblick

- Was erwartet man von CM?
  Durch CM sollen alle Konfigurationsdateien von IT Komponenten und Services zur Verfügung gestellt werden. CM soll einen soliden, zuverlässigen Grundstock für andere ITSM Prozesse anbieten. Die Änderungen an der IT-Infrastruktur sollen durch CM abgebildet und Fehler während der Prozesse sollen durch CM schnell festgestellt und korrigiert werden.

- Beziehung zwischen CM und anderen Prozessen
  Die folgende Abbildung 2.3 zeigt uns die Beziehung zwischen CM und anderen Prozessen. Fast alle Prozesse, die vom ITIL-Framework definert werden, sind mit dem CM

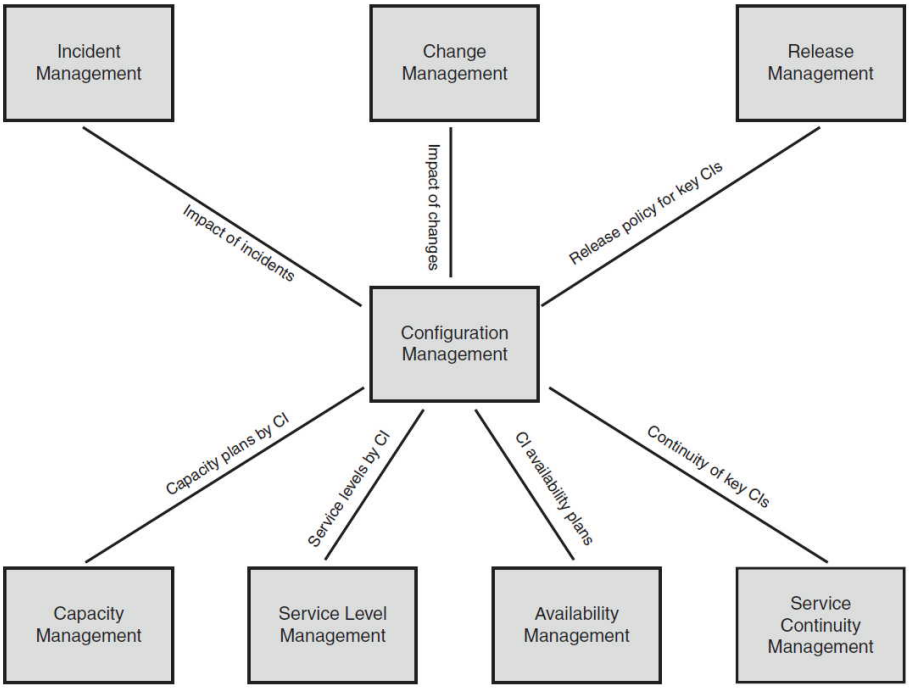


Abbildung 2.3: CM Beziehung zu anderen Prozessen nach [AC07b]

  verbunden. Sie greifen immer auf die CM Daten zu und ohne CM können sie nicht optimal funktionieren. Deshalb zählt CM zu den wichtigsten Prozessen in ITIL. Bei der Entwicklung von CM soll man deshalb auch prüfen und absichern, ob das CM auch die Anforderungen anderer Prozesse erfüllt.

- Was soll Configuration-Management berücksichtigen?
  Laut der ITIL Service Support Spezifikation dient das Configuration-Management zur Identifizierung, Kontrolle und Aufrechterhaltung der IT Komponenten. Im Prinzip soll alles, das die IT Umgebung bildet, von CM betrachtet werden. Davon sind die physischen Beziehungen und logischen Abhängigkeiten zwischen Komponenten, Subkomponenten, Applikationen, Server, Netz und Dokumentationen am wichtigsten.

2.3.2 Theoritische Grundlage von CM

Ziel von CM

Die Ziele, die durch CM erreicht werden sollen, sind:

- Die Informationen von allen Objekten in der IT-Ungebung sollen dokumentiert werden und ein Zugang zu den Informationen soll bereitgestellt werden.
- Die dokumentierten Informationen können von anderen ITSM Prozessen benutzt werden.

- CM soll eine solide Basis für Incident-Management, Problem-Management, Change-Management und Release-Management anbieten.

- Die Systemfehler können durch CM verhindert oder das System kann bei Fehlern schnell wieder in Ordnung gebracht werden.

- Anhand der CM Infrastruktur werden die Konfigurationsdaten überprüft und die Abweichungen werden beseitigt.

Aufgaben von CM

CM beinhaltet normalerweise folgende Aufgaben: Planen, Identifizieren, CI-Kontrolle, Reporting, CI-Prüfung.

- Planen
  Die Ziele, der Umfang, die verwendete Technik, Prozeduren usw. von CM werden klar geplant und definiert.

- Identifizieren
  Die CIs und Beziehungen werden identifiziert und eine passende Struktur der CMDB wird definiert.

- CI-Kontrolle
  Die CIs werden kontrolliert, so dass nur authentifizierte CIs in der CMDB gespeichert werden. Die Änderungen von CIs und Beziehungen werden geprüft und dokumentiert.

- Reporting
  Der akutelle und historische Status kann durch CM nachgeschlagen werden.

- CI-Prüfung
  Die CIs plus Beziehungen, die in der CMDB dokumentiert werden, sollen in der IT-Umgebung geprüft werden. Wenn sie nicht übereinstimmen, muss die CMDB geändert werden.

Vorteile und Nachteile von CM

Die Einführung von CM kann die Effizienz von ITSM erhöhen, weil: [AC07a]

1. CM bietet ausführliche Information von CIs.
   Diese Informationen unterstützten alle anderen ITSM Prozessen wie Release-Management, Change-Management, Incident-Management, Problem-Management usw.

2. Wertvolle CIs können durch das CM gut kontrolliert werden.
   Als Beispiel schauen wir ein Szenario an:
   Im MWN ist ein Netzteil defekt. Nach Untersuchung muss ein Router ausgetauscht werden. Durch das CM wissen wir sofort, wo der Router steht, welches Produkt eingesetzt war und wie die Einstellungen wiederherstellt werden sollen.

3. Nicht autorisierte CIs können einfach vom System ausgeschlossen werden.
   Software ist ein Typ von CI, der in allem CM Systemen dokumentiert werden soll. Illegale Software kann einfach identifiziert werden, wenn sie nicht in der Softwareliste des CM Systems steht.

4. CM kann die Kostenplanung unterstützen.
   Mit CM kann man ohne Aufwand die Liste von allen CIs erstellen. Von dieser Liste ausgehend kann man einfach verschiedene Kosten wie Wartungskosten von Hardware, Lizenzkosten von Software, CI-Ablaufsdatum, Austauschskosten usw. berechnen.

5. Systemänderungen sind sichtbar.
   Durch das CM weiß man, welche Teile vom System wann ausgetauscht wurden.

6. CM hilft dem Entscheidungsfinden.
   Wenn man über vollständige Daten verfügt, ist es einfacher und genauer, eine Entscheidung zu treffen.

Außer den Vorteilen kann CM auch negative Auswirkungen haben.

1. Die CI-Definition ist wichtig.
   Wenn wir ein CI in einer falschen Stufe zuordnen, besitzt dieses CI zu viel der zu wenig Eigenschaften. Diese falsche oder ungenaue Definition kann die Speicher- und Sucheeffizienz einschränken.

2. Definition ganz korrekter CM ist oft schwierig.
   Wenn z.B. in einer IT-Umgebung selbsständig Software von Internet heruntergeladen oder installieren werden kann, kann das CM nicht rechtzeitig darauf reagieren. In diesem Fall sind die Informationen in der CMDB oft nicht korrekt bzw. aktuell.

3. Manchmal ist CM nur sinnvoll, wenn man die anderen Prozessen wie Change-Management oder Release-Management auch mit einführt. Wenn man nur die IT-Komponenten verwalten möchte, sind ggf. Informationen wie Versionen im CM überflüssig.

4. Es kann sein, dass die CMDB die Entitäts-Relation (ER) Eigenschaft verliert. Da CI und CI-Beziehung die zwei wichtigsten Begriffe in CM sind, bauen viele CM Systeme die CMDB nur mit diesen zwei Haupttabellen. Solche CMDB erfüllen nicht mehr das ER Prinzip.

5. CM kann wahrscheinlich nicht allen Wünschen entsprechen.
   Ein Netzadministrator wüscht beispielsweise, dass ein Tool die ganz Netztopologie formulieren kann. Mit CM können alle Netzkomponenten verwaltet werden, aber eine automatische Darstellung der Netztopologie kann sehr wahrscheinlich nicht angeboten werden. Eine weitere Software oder Prozess muss entwickelt werden, um die Wünsche zu erfüllen. [AC07b]

2.3.3 Configuration-Management-Database

Die CMDB ist ein Datenspeicher, der alle CIs der IT-Umgebung enthält.

Welche Struktur kann eine CMDB haben?

Im Laufe der Zeit hat sich die Struktur der CMDB geändert, von einer Sammlung von Daten der isolierten ITIL Prozesse, über einen integrierten Datenspeicher zu einer einzelnen, zentralen Datenbank. [DC07]

- CMDB aus Daten der isolierten ITIL Prozessen
  Die CMDB dieses Typs bestand einfach aus verschiedenen Anwendungen, die jeweils ihre eigenen Daten speicherten, und vielfach weiteren Datenbanken mit Konfigurationsdaten. Die Nachteile sind offensichtlich. Die Daten bleiben lokal und es gibt viel Aufwand, die Daten zwischen den Prozessen auszutauschen.

- CMDB aus integrierten Datenspeichern
  In diesem Fall besitzt jeder Prozess eine Datenquelle und jeder diese Datenquelle wird mit Datenanbietern (oft Datenbank anderer Prozesse) verbunden. Die CMDB ist eine direkte Integration solcher Datenquellen. Der Nachteil dieser CMDB-Struktur ist, dass erheblich viele Ressourcen erforderlich sind, um die verschiedenen Integrationen herzustellen und zu pflegen.

- Eine einzelne, zentrale CMDB für alle Prozesse
  Die CMDB ist eine einzelne, alles umfassende Datenquelle für Konfigurationsdaten, auf die alle Anwendungen zugreifen können. Aber diese Struktur hat wiederum Nachteile. Bei einem großen System erfordert sie eine große Kapazität an einem einzigen Ort und schafft einen Engpass, weil alle Anforderungen von Daten und deren Aktualisierungen über denselben Pfad laufen. Weiterhin ist ein Migrationsaufwand erforderlich, um alle Daten in die einzige Datenbank zu überführen. Oft wird ein Kommunikationsprozess gebraucht.

In unserer IT-Umgebung stellen wir eine einzelne zentrale CMDB her. Zwar nehmen wir eine Federated-CMDB Architektur. Unter einer Federated-CMDB versteht man eine logische Datenbank, deren Inhalte über verschiedene Datenquellen verteilt sind. Mit zunehmender Reife im Configuration-Management werden immer mehr Entitätstypen identifiziert, die im Sinne einer ITIL CMDB verwaltet oder referenziert werden müssten (z.B. Personen, Gebäude, Netz usw). Die typische Federated-CMDB Architektur ist in Abbildung 2.4 dargestellt: Die Federated-CMDB Architektur enthält fünf Grundelemente:

- Federated CMDB

- Consumers
  Consumers sind die Clients, die Daten aus der CMDB anfragen.

- Provider
  Provider sind die Datenquellen, aus denen die Daten in die CMDB integriert werden.

- Data-Provider-Interfaces
  Die Data-Provider-Interfaces kommunizieren mit dem Provider und integrieren die Daten in die CMDB.

- Data-Consumer-Interfaces
  Die Data-Consumer-Interfaces kommunizieren mit Clients, holen die Daten aus der CMDB und bewantworten die Anfragen der Clients.

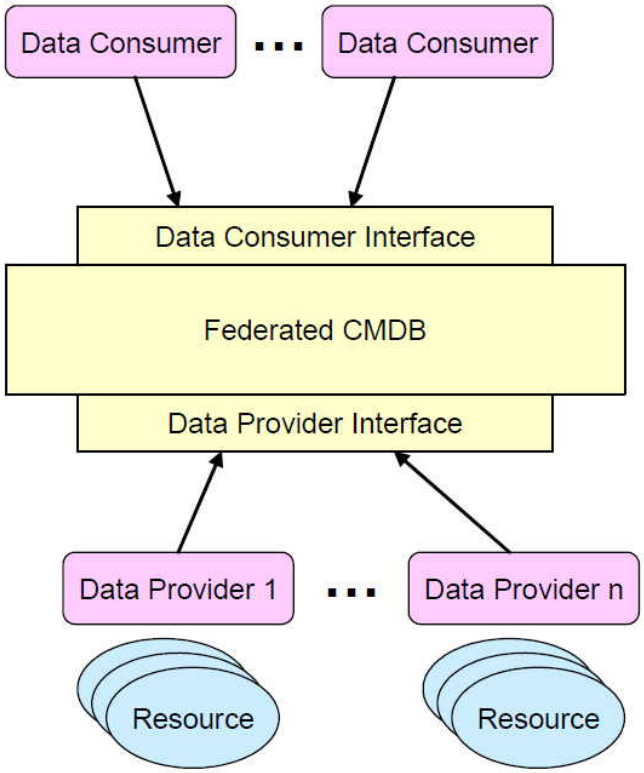


Abbildung 2.4: Federated-CMDB Struktur nach [DC07]

Im Rahmen der CM-Implementierung in Abschnitt 4 analysieren wir die Consumers, Provider und Data-Provider-Interfaces.

2.4 Zusammenfassung

In disem Kapital haben wir die Begriffe ITSM, ITIL, CM und CMDB erläutert. Wir haben die Fragen, die am Anfang dieses Kapitals ausgestellt wurden, beantwortet. In den nächsten Kapital analysieren wir, wie wir die CMDB in unserer IT-Umgebung vernüftig planen und implementieren können.

3 CMDB Planung

Seit den letzten beiden Kapiteln wissen wir, was eine CMDB ist und was in einer CMDB dokumentiert werden soll. Nach dem Konzept von ITIL planen und implementieren wir in den folgenden zwei Kapiteln den Umfang der IT-Resourcen in unserer IT-Umgebung für die CMDB in OTRS. Die ganze Arbeit wird in zwei Schritte aufgeteilt: Planung und Implementierung. Die Planung der CMDB lässt sich auf vier Schritte aufteilen: Anforderungsammlung, Anforderunganalyse, CMDB-Analyse und CMDB-Definieren.

3.1 Schritt 1: Anforderungsammlung

In diesem Schritt werden die Wünsche von unseren Systembenutzern gesammlt, welche Ziele durch die CMDB erreicht werden sollen. Die Anforderungssammlung ist wichtig, da sie den Umfang der CMDB abgrenzt und die Kriterien für die Systemtest liefert.
Im Allgemeien soll eine minimale CMDB die folgenden Informationen erhalten:

- Die CMDB soll alle Informationen von relevanten CIs und CI-Beziehungen erhalten.
- Die Personen, die keine Berechtigung haben, sollen vom CM-System isoliert werden. Deshalb soll die Authentifizierungsinformationen auch in der CMDB gespeichert werden.
- Das CM soll die hierarchische CI-Struktur unterstützen. Beispielsweise haben wir in unserer IT-Umgebung einen CI-Typ Software. Unter Software gibt es weitere Subtypen wie Betriebssysteme, Kommunikationssoftware, Officeserie usw. Die CMDB soll auch für die Speicherung solcher hierarchischer CI-Strukturen geeignet sein.
- Die Informationen von CIs können einfach in die CMDB eingefügt und aus der CMDB entfernt werden.
- Da CM normalerweise die Suchefunktion nach CIs bzw. Beziehungen anbieten soll, soll die effziente Suche von der CMDB unterstützt werden.
- Das CM-System soll jedes CI eindeutig identifizieren. Dieses CI soll auch in der CMDB eindeutig dargestellt werden.
- Das CM soll die effziente Suche nach CIs anbieten, damit wir ein bestimmtes CI oder eine bestimmte Beziehung auswählen können.
- Das CM-System verwaltet zentral die Informationen von CIs. Die Struktur von CI-Definitionen, wie ein CI im System dargestellt werden soll, ist in der CMDB enthalten.
- Das CM-System soll historische Informationen von CIs liefern, deshalb sollen solche Informationen auch in der CMDB enthalten sein. Wenn die Definition von einem CI-Typ geändert wird, sollen die CIs an die neue Definition angepasst werden und nicht

verloren werden. Beispielsweise definieren wir den CI-Typ Computer mit einer Eigenschaft IP-Adresse und mehrere dieser CIs werden dokumentiert. Wir wollen dann das CM-System vergrößern und den CI-Typ Computer in zwei Typen splitten: Computer und Netzkarte. Die Eigenschaft IP-Adresse wird vom Typ Computer entfernt und gehört zum neuen Typ Netzkarte. In der CMDB sollen die historischen IP-Adressen von Computer aber nicht gelöscht werden, da sie für die Systemwiederherstellung sehr wichtig sind.

- Die CMDB ist die Basis für andere Prozesse. Falls andere Datenbanken wie z.B eine Problem-Management Datenbank bestehen, soll sie ein Interface für andere Datenbank anbieten.
- Die CMDB soll die Änderungen von Daten dokumentieren. Sie sind besonders wichtig für das Change-Management.
- Die CMDB soll zu bestimmten Zeitpunkten die aktuellen Daten sichern und im Notfall alle CIs in einer gesicherten Version wiederherstellen.

Neben den oben vorgestellten allgemeinen Anforderungen haben wir als Benutzer eigene Wünsche. Wenn wir hier einen Schritt tiefer gehen, erwarten die Benutzer von unserem CM-System mehr von der CMDB.

- Die CMDB soll die Informationen von allen Netzkomponenten, Verbindungen zwischen Komponenten, Lageplänen, Organisationen erfassen, die in der Netzdoku dokumentiert werden.
- Die CMDB lässt sich für die möglichen IT-Resourcen wie Dienste, Software usw. erweitern.
- Die CMDB soll die Information, die in verschiedenen Datenbanken von Fakultäten erhalten sind, integrieren.
- Die CMDB soll ein Import- und Exportinterface anbieten, damit sie mit anderen Konfiguratiossystemen kommunizieren kann.
- Die CMDB soll die CIs von verschiedenen Systemen identifizieren und die Duplikate beseitigen. Beispielsweise verbindet der Switch phy-sw2 das MWN und das lokale Netz der Physik. Genau dieser Switch wird allerdings in beiden Systemen unterschiedlich dargestellt. Wenn wir eine integrierte CMDB haben, soll dieser Switch in der CMDB eindeutig identifiziert werden.

3.2 Schritt 2: Anforderungsanalyse

Vor allem sollen die IT-Resourcen in unserer IT-Umgebung in der CMDB dokumentiert werden. In diesem Schritt schauen wir uns genau an, welche IT-Resourcen zu verwalten sind und in der CMDB erhalten sein sollen.
Die IT-Resoucen, die zu verwalten sind, kann man grundsätzlich auf drei Gruppen aufteilen: organisatorische Informationen (z.B. Institute, Person usw.), netzbezügliche Resourcen und übrige Resourcen (z.B. Software, Rechner).

3.2.1 Organisatorische Informationen

Die folgende Informationstypen sollen von unserer CMDB berücksichtigt werden:

- Account
  Es gibt verschiedene Systeme auf dem Campus TUM. Beispielsweise hat jeder TMU-Student eine sogenannte Campus Kennung. Mit dieser Kennung kann man auf die Email, Uniinfos, Fakultätinfos usw. zugreifen. Die Accountinformationen wie Username und Priorität sollen in der CMDB bewahrt werden.

- Organisation
  Organisationen sind die großen Verwaltungseinheiten wie TUM, LRZ usw.

- Institut
  Ein Institut ist ein Bestandteil einer Organisation. Eine Organisation wie die TUM beinhaltet verschiedene Fakultäten, Verwaltungsstellen usw.

- Arbeitsgruppe
  Wenn man die Institute noch verfeinert, bekommt man die Arbeitsgruppen. Beispielsweise beinhaltet die Fakultät Physik für TUM mehrere Arbeitsgruppe, wie etwa Lehrstuhl für Experimentalphysik, Lehrstuhl für Physik-Biophysik, Lehrstuhl für Technische Physik usw.

- Person
  In unserer IT-Umgebung spielen viele Menschen eine Rolle. Die Professoren leiten die Lehrstühle. Die Studenten haben Hiwistellen. Die Netzverantwortlichen sind für die Netzkomponenten zuständig. Wenn man Probleme hat, kann man durch die Personeninfos den richtigen Mensch finden.

- Gelände und Gebäude
  Das Gelände und Gebäude sind geographische Begriffe. Sie werden mit Bezirken gekennzeichnet. So wird z.B. dem Garchinger Hochschulgelände der Bezirk A zugeordnet. Das Physikgebäude, das im Garchinger Hochschulgelände liegt, hat den Bezirk AT.

- Raum
  Der Raum ist die kleinste geographische Einheit.

3.2.2 Netzbezügliche Resourcen

- Netzkomponente
  Im MWN befinden sich viele verschiedene Netzkomponenten: Accesspoint, Bridge, Cheapernet-Segment, Hubs, Switches, Firewall, Router, ISDN-Terminaladapter, Koax-Kopplung, Medien-Konverter, Modems, Repeater, Tranceiver, VoIP-Anlage usw.

- VLAN
  Ein VLAN ist ein separates Teilnetz innerhalb eines physisch geswitchten Netzes, das aus einem oder mehreren Switches bestehen kann. Die Zuordnung von Datenverkehr zu einem VLAN kann statisch über Ports an den Switches erfolgen oder dynamisch zum Beispiel durch MAC-Adressen, IP-Adressen, bis hin zu TCP- und UDP-Port und höheren Protokollen. Ein VLAN wird aus zwei Gründen eingeführt:

Performanz-Aspekte: Häufig möchte man einfach die Broadcast-Domänen verkleinern, so dass sich ein Broadcast nicht über das gesamte Netz ausbreitet, sondern nur innerhalb des VLANs.
Sicherheitsaspekte: Geswitchte Netze sind als unsicher anzusehen, denn für sie existieren eine Vielzahl von Angriffsmöglichkeiten, wie zum Beispiel ARP-Spoofing. Routing, was schließlich die einzige Kommunikationsmöglichkeit zwischen VLANs ist, weist hier eine wesentlich höhere Sicherheit auf. Außerdem bietet Routing auch die Möglichkeit Firewalls einzusetzen, wodurch die Sicherheit erhöht wird.

- Subnetz
  Das Subnetting bezeichnet ein Verfahren, um Netzklassen in kleinere Teil-Netze aufzuteilen. Die Abtrennung des vom Subnetz erfassten Bereichs erfolgt mittels bitweiser Maskierung eines bestimmten Teils der IP-Adresse durch die Subnetzmaske. Dadurch erhält man aus einer beliebigen Adresse das Subnetz, zu dem die Adresse unter Annahme dieser Maske gehört.

- Teil von Subnetz
  Ein Subnetz ist manchmal auch zu groß. Die Subnetze kann man auch in weitere Teile aufteilen.

- Produkt
  An der Stelle einer Komponente kann man verschiedene Produkte von verschiedenen Firmen verwenden.

- Leitung
  Die Leitung ist die physische Verbindung zwischen Komponenten. Wir können die Richtung einer Verbindung definieren, so dass die Leitung die Komponente von A nach B verbindet, wenn die Komponente näher zum Netzkern ist.

- physische Ports
  Die physischen Ports beziehen sich auf die Ports von Netzkomponenten. Die pyhsischen Ports besitzen die Eigenschaften wie Schnittstellentyp und Geschwindigkeit. Der Router csr2-2wr ist z.B. ein Cisco Produkt Catalyst 6509. Auf Port 1/1 davon ist ein 10 Gigabit Ethernet eingerichtet und die Geschwindigkeit des Ports ist 10 GBit/s.

- logische Ports
  Die logischen Ports einer Komponente tragen die logischen Eigenschafts der Ports. Beispielsweise kann einem logischen Port eine IP-Adresse zugeordnet werden und auf diesem Port können mehrere VLANs eingerichtet werden. Es können mehrere physische Ports zu einem logischen Port zusammengefasst werden, um die Bandbreite zu steigern; dies kann über statische oder dynamische Verfahren, z.B. Link Aggregation Control Protocol (LACP) oder Port Aggregation Protocol (PAgP), erfolgen.

3.2.3 Übrige Resourcen

Die folgenden klassischen Resourcentypen, die in fast allen IT-Umgebungen enthalten sind, sollen auch in unserem CM enthalten sein.

- Software

- Hardware
- Dokumentationen
- verschiedene Dienstleistungen

3.2.4 Zusammenfassung der Anforderungsanalyse

Die zu verwaltenden IT-Resourcen sind im UML Diagramm 3.1 zusammengefasst.

In der Abbildung 3.1 gibt es drei Typen von Linien. Die grüne Linie bedeutet Aggregation; z.B. bilden die fünf Teilkomponenten, VLAN, Subnetz, Produkt und Netzdieste zusammen ein Netz. Die blaue Linie bedeutet Ableitung. Die Klasse Student und Netzverantwortlicher sind beispielsweise von der Klasse Person abgeleitet. Die rote Linie zeigt die Assoziation an; ein Netzverantwortlicher betreut z.B. ein Gebäude.

3.3 Schritt 3: CMDB Analyse

In Schritt 2 haben wir analysiert, was zu verwalten ist. In diesem Schritt werden die CIs sowie die Definition von CIs und die Beziehungen von CIs von den IT-Resourcen verfeinert. Das Ergebnis hilft uns, eine ideale CMDB-Struktur in Abschnitt 3.4.2 aufzubauen.

3.3.1 Methodik

- CI-Typ
  Grundlegend können wir aus jeder Klasse in der UML-Abbildung einen CI-Typ definieren. Die Attribute der Klasse kann man als CI-Eigenschaften übernehmen. Aus der Klasse CM_Komponente können wir z.B. den CI-Typ CIKomponente konstruieren.

- CI-Subtyp
  Zu beachten sind die Ableitungen. Manche Ableitungen können wir als Subtyp von CIs definieren. In der Abbildung 3.1 werden die Klassen Professor, Student, Mitarbeiter und Sachbearbeiter usw. von der Klasse Person abgeleitet. Wir können in der CI-Definition der CIPerson ein neues Feld PersonTyp hinzufügen, wobei dieses Feld die Werte Professor, Student usw. hält.

- Beziehungen
  Wir können aus den Assoziationen zwischen Klassen die Beziehungen definieren. Wir haben z.B. aus der UML Abbildung zwei Klassen Netzverantwortlicher und Komponente und eine Assoziation dazwischen, dass der Netzverantwortliche die Komponenten betreut. Aus dieser Assoziation können wir eine n:m-Beziehung BETREUEN definieren.

Nach dieser Methodik können wir jetzt die CIs zusammengefassen. Als Beispiel schauen wir die Definition von CIKomponent an. Die vollständigen Definitionen findet man in der am Lehrstuhl abgegebenen Version dieser Ausarbeitung.

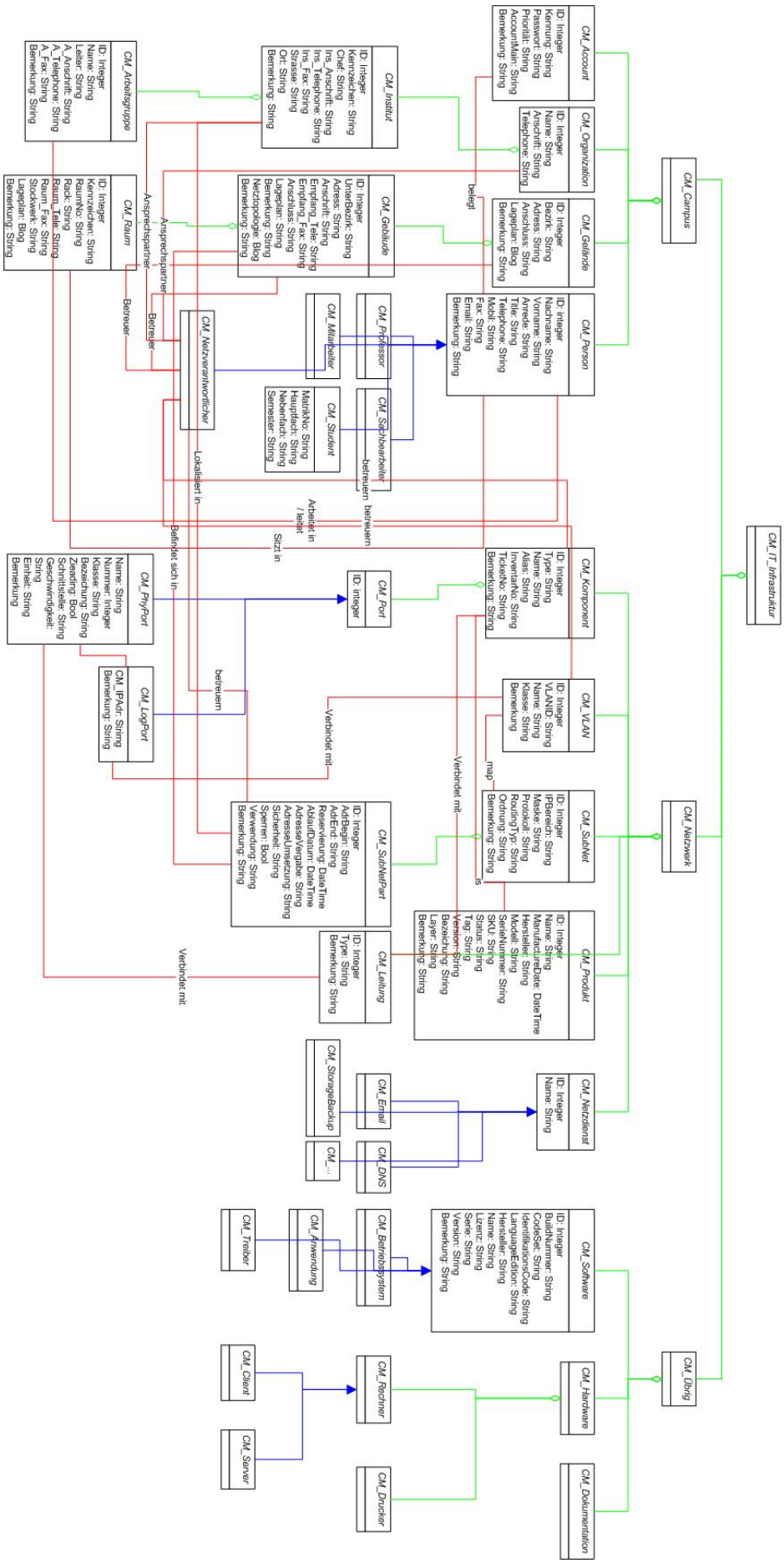


Abbildung 3.1: CI Typ Deklaration

3.3.2 CIs und deren Eigenschaften

Wir nehmen hier den Tpy CIKomponente als ein Beispiel. Die genaue Definitionen von allen Typen werden in der Abgabeversion dieser Ausarbeitung beschrieben.

- CIKomponente mit Eigentschaftstabelle 3.1

Eigenschaft	Beschreibung	Beispiel
Typ	Router, Switch usw.	Router
Name		Cisco Router - 7206
Alias	Systemname	bro01-0q1
InventarNo	ID in der Inventarliste	
TicketNo	Ticketnummer	ID in Incident-Management
RackNo	Die Position in einem Raum	1C
Bemerkung	Beschreibungen der Komponente	

Tabelle 3.1: Komponente Eigenschaft

- CIAccount, CIOrganisation, CIInstitut, CIArbeitsgruppe, CIGelände, CIGebäude, CI-Raum, CIPerson, CIPhyPort, CILogPort, CIVLAN, CISubnetz, CISubNetzTeil, CI-Produkt

3.3.3 Beziehungen

- CIOrganisation und CIInstitut (1:n Beziehung)
  Ein Institut gehört zu einer Organisation.
- CIInstitut und CIArbeitsgruppe (1:n Beziehung)
  Eine Arbeitsgruppe gehört zu einem Institut.
- CIGelände und CIGebäude (1:n Beziehung)
  Ein Gebäude gehört geographisch zu einem Gelände.
- CIGebäude und CIRaum (1:n Beziehung)
  Ein Raum gehört geographisch zu einem Gebäude.
- CIPerson und CIAccount (1:n Beziehung)
  Eine Person besitzt einen Account
- CIInstitut und CIPerson (n:m Beziehung)
  Eine Person arbeitet oder studiert bei einem Institut.
- CIArbeitsgruppe und CIPerson (n:m Beziehung)
  Eine Person arbeitet in einer Arbeitsgruppe.
- CIRaum und CIPerson (n:m Beziehung)
  Eine Person arbeitet in einem Raum.
- CIKomponente und CIPhyPort (1:n Beziehung)
  Ein Router besitzt mehrere Ports.

- CIPhyPort und CILogPort (1:n Beziehung)
  Ein Logik-Port liefert als wichtige Information die IP-Adresse. Einem CIPhyport kann man mehrere IP-Adressen zuordnen.

- CIVLAN und CIPhyPort (1:n Beziehung)
  Ein VLAN wird auf mehreren Ports aufgebaut.

- CIVLAN und CISubnetz (n:m Beziehung)
  Innerhalb von einem Subnetz werden mehrere VLANs eingerichtet. Die Komponenten von einem VLAN können sich auch auf mehreren Subnetze verteilen.

- CISubnetz und CISubNetzTeil (1:n Beziehung)
  Wenn ein Subnetz zu groß ist, können wir es noch aufteilen.

- CISubnetz und CILogPort (1:n Beziehung)
  Ein CISubnetz beinhaltet einen IP-Bereich. Das CILogPort, das eine IP-Adresse besitzt, wird von dem CISubnetz erfasst.

- CIKomponente und CIProdukt (n:1 Beziehung)
  An der Stelle einer Komponente wird ein Produkt eingesetzt. Ein Produkt kann für mehrere Komponenten eingesetzt werden.

- CIPhyPort und CIPhyPort (1:1 Beziehung)
  Eine Verbindung zwischen Komponenten wird durch eine 1 zu 1 Beziehung zwischen physischen Ports gekennzeichnet.

- CIRaum und CIKomponente (1:n Beziehung)
  In einem Raum werden mehrere Komponenten platziert. Weiterhin wird ein Raum in mehreren Racks aufgeteilt, falls er zu groß ist.

- CIPerson und CIKomponente (n:m Beziehung)
  Eine Person verwaltet mehrere Komponenten. Eine Komponente kann von mehreren Personen zusammen verwaltet werden.

- CIInstitut und CISubNetzTeil (n:m Beziehung)
  Innerhalb von einem Institut kann man mehrere Subnetzteile einrichten. Ein Subnetzteil kann auch mehr als ein Institut beinhalten.

- CIGebäude und CISubNetzTeil (n:m Beziehung)
  Diese Beziehung ist ähnlich wie Institut und Subnetzteil. Der Unterschied ist, dass das Gebäude ein geographischer Begriff ist.

3.4 Schritt 4: CMDB-Definition

Anhand von der vorangegangenen CI- und Beziehungsanalyse können wir jetzt eine ideale CMDB-Struktur entwickeln.

3.4.1 Methodik

Wir können grundlegend aus jedem CI eine Tabelle entwickeln. Die Felder der Tabelle bestehen aus CI-Eigenschaften und entsprechenden Steuerinformationen wir Primär-Schlüssel und Sekundär-Schlüssel. Die CIs, die ähnliche Eigenschaften haben, kann man in einer Tabelle zusammenfassen.
Eine 1:n Beziehung zwischen A und B kann dadurch dargestellt werden, dass wir in Tabelle B einen Sekundär-Schlüssel hinzufügen. Wir nehmen CIKomponente und CIPhyPort als Beispiel. In der Tabelle Phyport fügen wir ein Feld Komponente_ID ein, der zu Komponente.ID hinweist.
Eine n:m Beziehung soll durch eine zusätzliche Tabelle dargestellt werden. Wir nehmen CI-Person und CIKomponente als Beispiel. Wir haben zwei Tabellen Person und Komponente und bauen eine zusätzliche Tabelle Beziehung ein, in der zwei Primäry-Schlüsseln Person.ID und Komponente.ID gespeichert werden. Eine Paar von Person.ID und Komponente.ID bedeutet, dass die Person die Komponente betreut.

3.4.2 Zusammenfassung

In diesem Kapitel haben wir zuerst die Anforderungen, die unsere CMDB erfüllen soll, gesammelt. Diese Anforderungen sind auch die Kriterien, mit denen wir unsere endgültige CMDB bewerten. Diese Bewertungen erfolgt in Kapitel 5. Anschließend fassten wir nach den Anforderungen die CIs und CI-Beziehungen in unserer IT-Umgebung zusammen. Wir entwickelten eine Methodik, wie eine CMDB aufgebaut wird, und nach dieser Methodik haben wir eine ideale CMDB Struktur erstellt, wie die folgende Abbildung 3.2 zeigt:

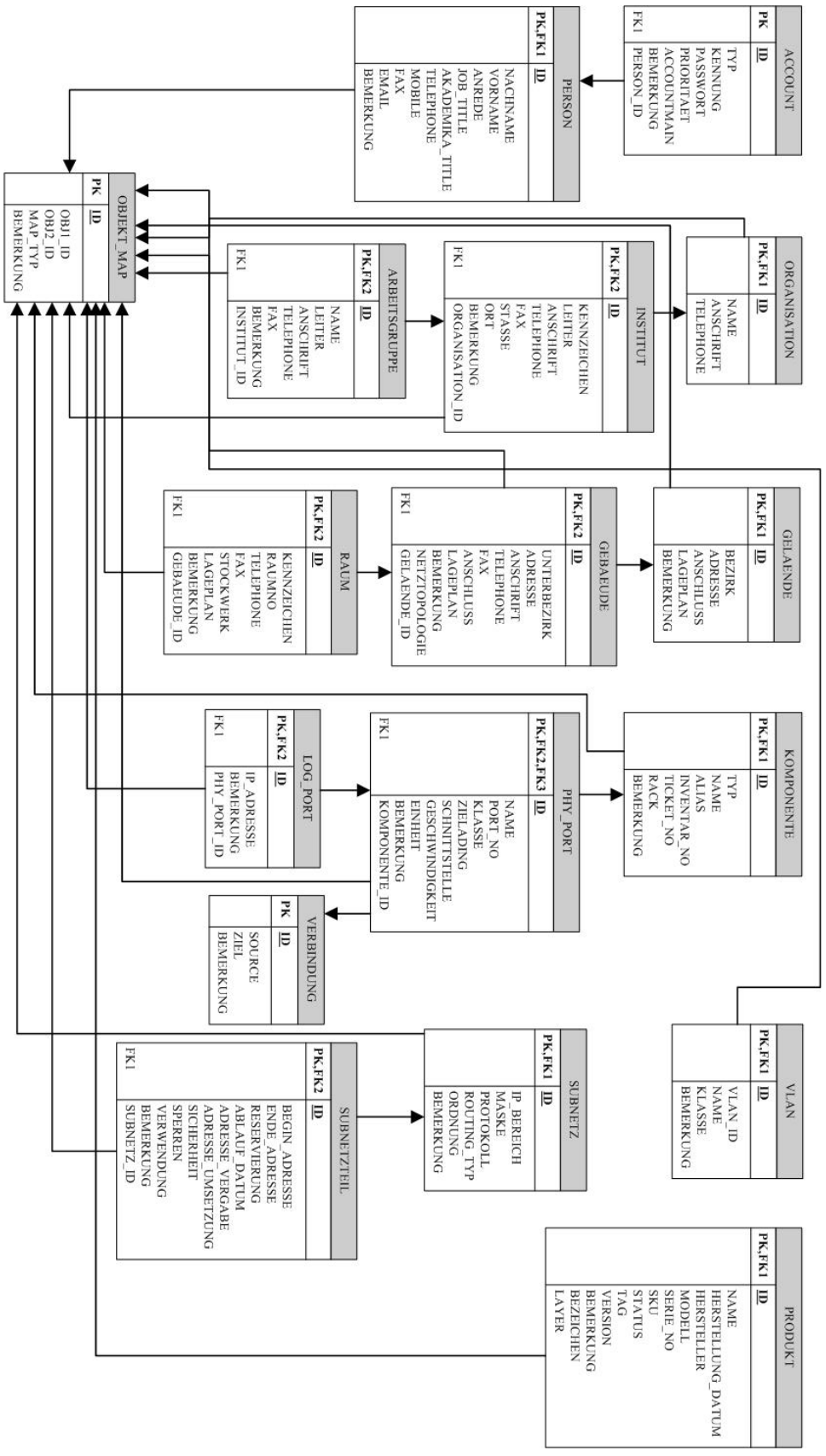


Abbildung 3.2: CI Typ Deklaration

4 CMDB Implementierung

Im Kapitel 3 haben wir unsere zu verwaltenden Resourcen analysiert und eine ideale CMDB-Struktur entwickelt. In diesem Kapitel bilden wir diese Struktur in der OTRS CMDB ab. Um die Daten zwischen OTRS und anderen Datenbanken auszutauschen, erweitern wir die entsprechenden Datenbanken und entwickeln ein Austauschsprotokoll sowie ein Programm „OTRS Adapter“.

4.1 Überblick über die ITSM Lösung von OTRS

Die Grundlage von OTRS ist ein Ticketsystem, das dem Incident-Management von ITIL entspricht. Im Rahmen des Configuration-Managements wird eine ITSM-Lösung von OTRS angeboten, damit das CM sowie das Problem-Management auch durch OTRS realisiert werden kann. Zum CM von OTRS gehören folgenden Komponenten: [OTR08a]

- Eine integrierte, individuell erweiterbare Configuration-Management Data Base (CMDB von OTRS).
- Rollen-, Verantwortlichkeits-, Berechtigungsmodell nach ITIL-Prinzipien
- Ein zentrales Webinterface, damit man die Berechtigung einstellen, die CI-Tyen definieren und CIs verwalten kann.
- Ein Suchinterface, damit man bestimmte CIs auswählen kann.
- CI Suche und Verknüpfung in Ticketerstellmaske. Aus der Ticketerstellmaske (Agenteninterface) heraus kann ein Popupfenster aufgerufen werden, über das ein neues Ticket direkt mit CIs oder bestehenden Tickets verknüpft werden kann.
- CMDB Import/Export Schnittstelle. Es wird zum einen die Möglichkeit geschaffen, die OTRS CMDB mit Daten aus CSV-Dateien zu befüllen, bzw. zu aktualisieren und zum anderen CMDB Daten als CSV zu exportieren. Aus Effizienzgründen wird diese Schnittstelle nicht von uns benutzt, sondern wir entwickeln ein eigenes Programm, das in den Funktionen mächtiger ist (siehe Abschnitt 4.4).

4.2 CM-Pakete Installation

Die aktuelle Version der CM-Pakete ist v1.08. Die Installation von Paketen in OTRS kann nur mit Administratorrechten durchgeführt werden. Man kann die CM-Pakete, die eigentlich XML-Dateien sind, auf der Seite Admin/Sonstiges/Paket Verwaltung installieren. Insgesamt werden 5 Pakete installiert werden: [OTR08b]

- GeneralCatalog
  Diese Paket beschreibt die Änderungen, die zur Anpassung der internen Einstellungen dienen.

- ITSMCore und ITSMService
  Diese zwei Pakete beschreiben die allgemeinen Einstellungen wie z.B. Datenbankstrukturänderungen, die zur Vorbereitung der Installation des CM-Systems gebraucht werden.

- ITSMConfigItem
  Dieses Paket beschreibt die Definitionen von CIs. Wenn wir einen neuen CI-Typ definieren, müssen wir an drei Stellen das XML ändern.

- LinkObject2
  Dieses Paket erweitert die Datenbankstruktur mit einer Tabelle link_object2. Alle Beziehungen zwischen CIs werden in dieser Tabelle gespeichert.

- ITSMLocation
  Die Lokation von CIs werden von OTRS intern gebraucht und als CI Location implementiert werden. Dieses CI wird im Paket ITSMLocation beschrieben. Dieses Paket führt einen CI-Typ Location in das System ein. Da die Lageinformationen auch von anderen Prozessen wie Incident-Management und Problem-Managemnent benutzt werden, wird dieser Typ von OTRS als ein Sonderfall ausgenommen und als ein einziges Paket angeboten.

Es gibt folgende zwei Methoden, wie man das vollständige ITSM in OTRS installiert.

Methode 1: Erweiterung der Pakete
Die CM-Pakete, die man auf dem Interface installiert, liefern die Skripte der Änderungen, die dann in der OTRS CMDB durchgeführt werden müssen. Deshalb können wir die neue Datenbankstruktur in den Paketen darstellen. Die Hauptaufgabe ist, ein neues Paket ITSM-ConfigItem herzustellen. Wir nehmen den CI-Typ Computer als Beispiel und schauen die Paketstruktur genau an:

- Stelle 1: CI-Typ Deklaration

- Stelle 2: CI-Typ Definition

- Stelle 3: CI-Typ Beschreibung

Hier werden die Eigenschaften des CI-Typs genau beschrieben. Die ganze Definition ist in Perl-Grammatik.
Nach den Änderungen an diesen drei Stellen bekommen wir dann einen neuen CI-Typ Computer. Änlich können wir die übrigen CI-Typen in der ConfigItem XML hinzufügen, die in Kapitel 3 definiert wurden. Wenn wir die fehlerfreien und erweiterten Paketen haben, ist die Installation und Konfiguration nach dem Durchlaufen der Skripte fertig.
Methode 2: Vorinstallieren und Nachkonfigurierung
Auf diese Weise installieren wir die Standardpakete, die von OTRS angeboten werden. Die CMDB und die Konfiguration werden auf einem intialen Status gebracht. Es sind vier CI-Typen vordefiniert: Computer, Hardware, Software und Network. Nach der Installation der Pakete müssen wir die folgende Konfiguration durchführen: [OTR08c]

- Schritt 1: CI-Typ Deklaration
  Da im Standartpaket nur vier Typen von CIs bestehen, müssen wir manuell die übrigen

hinzufügen. Die CI-Typ Deklaration wird auf der Seite Admin/System/Allgemeiner Katalog durchgeführt. Man wählt im Katalog die Klasse: ITSM::ConfigItem::Class, als Standardklasse, die von den CM-Paketen intial eingeführt wird, und fügt die CI-Typen hinzu. Die folgende Abbildung 4.1 zeigt, wie man einen neuen Typ VLan hinzufügt.

Abbildung 4.1: CI-Typ Deklaration

- Schritt 2: CI-Typ Definition
  Im Schritt 1 haben wir einen neuen CI-Typ eingeführt. In diesem Schritt werden die Eigenschaften von diesem Typ genau beschrieben und definiert. Wie die Abbildung 4.2 zeigt, ist der Typ VLAN ist sehr einfach und besitzt nur drei Eigenschaften: VLAN ID, Klasse und Bemerkung.

- Schritt 3: CI-Subtyp Definition
  Manchmal müssen wir noch die Subtypen definieren, die in CI-Typ Definitionen verwendet werden. Wir definieren z.B. den CI-Typ Person mit einer Eigenschaft Anrede. Wir wollen die Werte von Anrede begrenzen, d.h. der Wert von Anrede darf nur Mr, Mrs, Miss, Dr. oder Prof. sein. Auf der Seite Admin/System/Allgemeiner Katalog fügen wir einen neuen Typ ITSM::ConfigItem::Person::Anrede ein. Zu diesem Typ können wir die Werte nacheinander hinzufügen (siehe Abbildung 4.3).

Der Schwerpunkt der Methode 1 ist es, die Konfigurations-XMLs zu generieren. Am Anfang der Systementwicklung hat man normalerweise keinen Überblick über das ganze System, deshalb ist es schwerer, die Methode 1 durchzuführen. Aber wenn wir eine vollständige Konfigurations-XML haben, ist es sehr einfach, das System auf dem neuen Server zu installieren oder im Notfall wiederherzustellen. In dieser Arbeit, wird das System zuerst nach Methode 2 konfiguriert und damit wird dann ein komplettes Installpaket generiert. Dieses Paket befindet sich in der Abgabeversion dieser Ausarbeitung.

4.3 CI-Verwaltung in OTRS

In diesem Schritt schauen wir an, wie die CIs im OTRS verwaltet werden.

Config Item Klasse: VLAN

Definition:

```
[
    {
        Key => 'VLANID',
        Name => 'VLANID',
        Searchable => 1,
        Input => {
            Type => 'Text',
            Size => 50,
            MaxLength => 50,
        },
    },
    {
        Key => 'Klasse',
        Name => 'Klasse',
        Searchable => 1,
        Input => {
            Type => 'Text',
            Size => 50,
            MaxLength => 50,
        },
    },
    {
        Key => 'Bemerkung',
        Name => 'Bemerkung',
        Searchable => 1,
        Input => {
            Type => 'TextArea',
        },
    },
];
```

Abbildung 4.2: CI-Typ Definition in OTRS

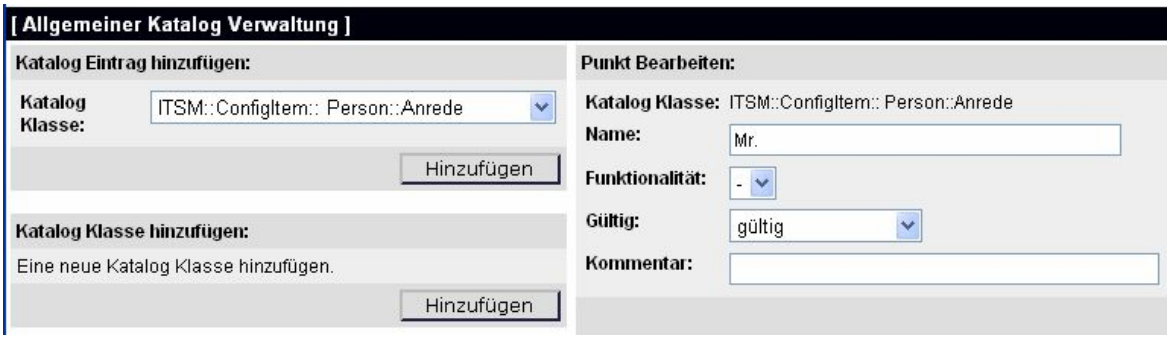


Abbildung 4.3: CI-Subtyp Definition

4.3.1 CIs und Beziehungen

OTRS bietet ein Interface an, mit dem die CIs mit ihren Beziehungen angezeigt werden. Die folgende Abbildung zeigt uns das CI crs1-2wr an. Wie die Abbildung 4.4 zeigt, kann man

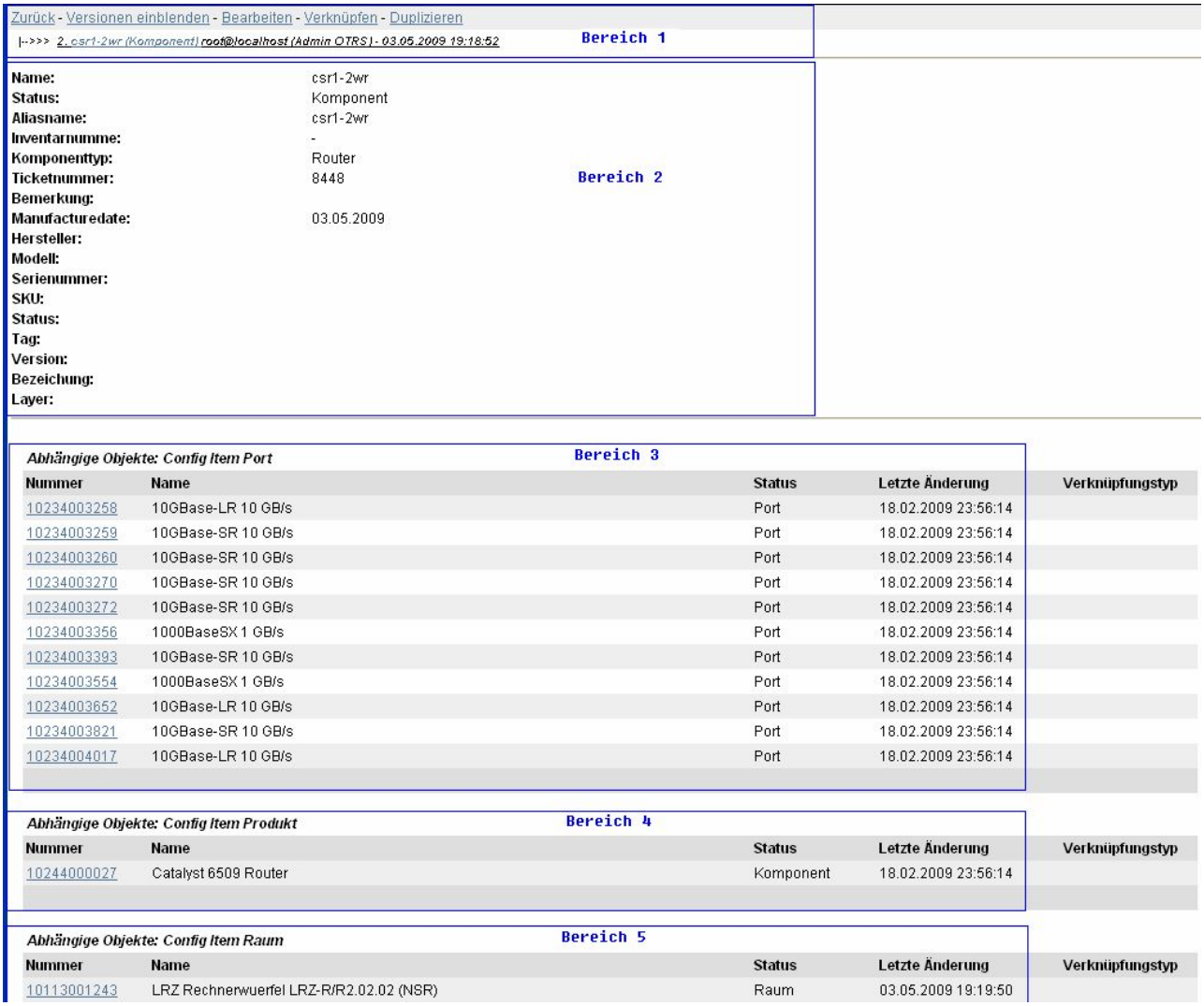


Abbildung 4.4: CI Beispiel

durch die Buttons „Bearbeiten“ und „Verknüpfen“ im Bereich 1 die Informationen eines CIs bearbeiten und eine Beziehung zwischen crs1-2wr und anderen CIs einrichten. Im Bereich 2 werden die Basisinformationen von csr1-2wr angezeigt. Es ist ein Router und hat die LRZ-Remedy-Ticketnummer 8448. Wenn es Störungen bei diesem Gerät gibt, sollte man in der E-Mail die Nummer 8448 als Referenz angeben. Das erleichtert die Störungsbehandlung.
Die Bereiche 3, 4 und 5 zeigen alle CIs, die mit dem Router verknüpft sind. Elf Ports von diesem Router werden verwendet, damit eine Netzverbindung zwischen anderen Komponenten und dem Router eingerichtet werden. Durch Klicken auf die Links können wir die Verbindungsinformationen genauer anschauen. Als Router hat man das Cisco Produk Catalyst 6509 verwendet. Durch Klicken auf die Links von diesem Produkt können wir mehre Infor-

mationen über diesen speziellen Router bekommen. Insgesammt arbeiten 12 Cisco Catalyst 6509 im MWN. Bereich 5 zeigt die Lokation von dem Router.

4.3.2 CI Suche

Das OTRS bietet auch ein Such-Interface an, damit man mit angegebenen Informationen ein CI finden kann. Die Suche in OTRS ist nicht sehr effzient. Man muss schon vorher ganz genau die Suchbedigung angeben. Wenn wir den Router crs1-2wr suchen, müssen wir genau den Namen angeben. Eine Wildcard-Suche wird nicht unterstützt. Deshalb ist es notwendig, einen effizienteren Suchmodus zu entwickeln. Dies ist eine Erweiterungsarbeit der vorliegenden Arbeit. Das folgende Programm OTRS-Searcher (siehe Abbildung 4.5) liefert eine Möglichkeit (den entsprechenden Code findet man in der Abgabeversion dieser Ausarbeitung):

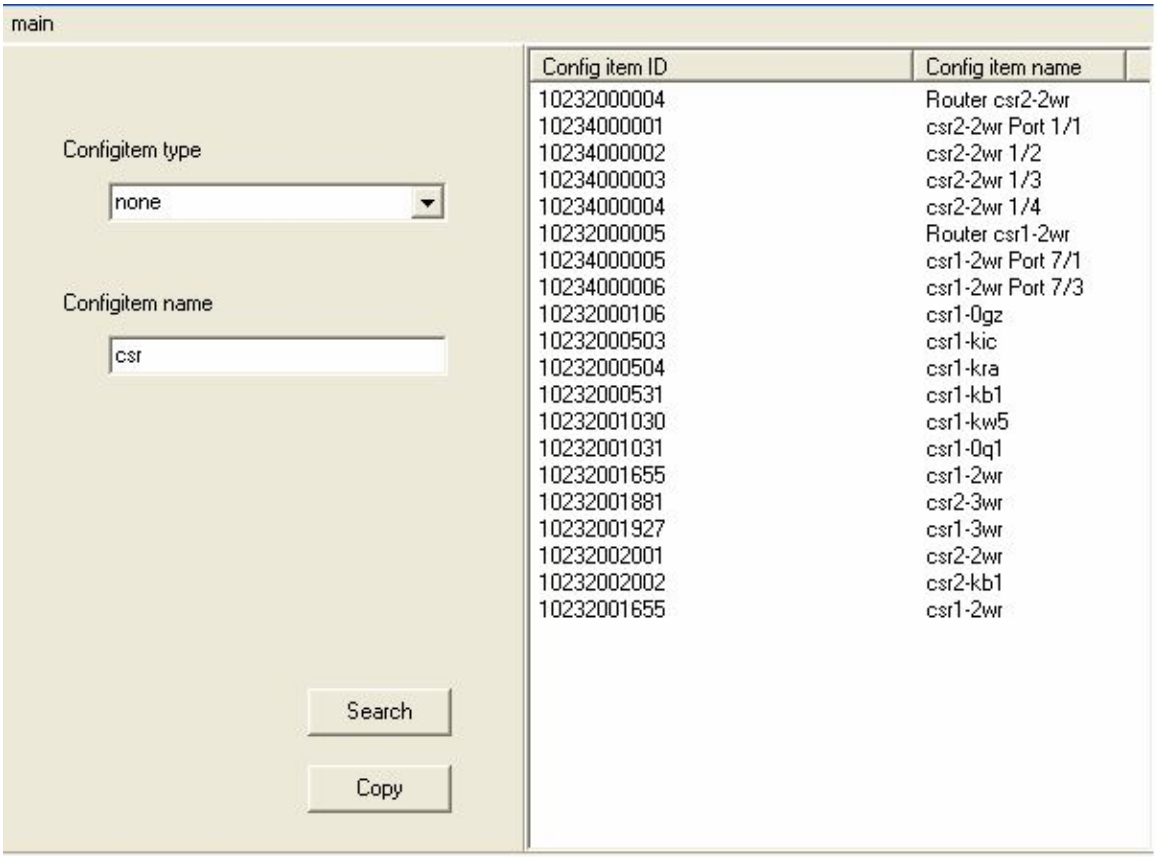


Abbildung 4.5: OTRS Suche, Erweiterung mit Wildcard-Suche

4.4 Datenaustausch zwischen OTRS und bestehenden Datenbeständen

Bis jetzt haben wir die ITSM Lösung von OTRS implementiert, aber es befinden sich noch keine Daten in der CMDB von OTRS. Da es in unserem Beispiel über 10.000 zu erfassende CIs in unserer IT-Umgebung gibt, ist die manuelle Dateneingabe durch das Interface nicht

effizient. Ein Synchronisationsprozess wird in dieser Arbeit entwicklt werden, um unsere CMDB zu befüllen.

4.4.1 Grundidee

Die CIs und die Informationen von CIs, die von OTRS verwaltet werden, kommen aus verschiedenen Datenbanken wie z.B Netzdoku und lokaler Physikdatenbank (vgl. Abschnitt 1.3.2). Da die Datenbanken unterschiedliche Datenstrukturen haben, muss ein Adapter „OTRS Sync“ entwickelt werden, um die Informationen zwischen Quelldatenbank und OTRS zu synchronisieren. Die Grundidee ist, den Datenaustausch via XML zu realisieren. Der OTRS Sync entspricht einer Datenflussrichtung, also von der Quelldatenbank (Netzdoku, Fakultät Physik) nach OTRS CMDB. Der OTRS Sync besteht aus zwei Teilen: Netzdoku-Exporter und OTRS-Importer. Der OTRS Sync läuft zu bestimmten Zeitpunkten (z.B. täglich 00:00). Der Netzdoku-Exporter sucht nach den neuen Änderungen in der Netzdoku und bildet sie in XML-Dateien ab. Die XML-Dateien werden vom OTRS-Importer ausgelesen und die Änderungen werden in der OTRS CMDB durchgeführt. Nach der Anforderungsanalyse ist jetzt nur eine Datenflussrichtung gefragt. Falls die Daten von OTRS nach Netzdoku zurückgehen sollen der OTRS Sync erweitert werden. Hier wurden die zwei neue Programme OTRS-Exporter und Netzdoku-Importer entwickelt. Das Prinzip der Syncprozesse ist:

- Vollständigkeit: Durch Datentransport von Quelldatenbank zu Zieldatenbank sollen keine nützliche Informationen verloren gehen.
- Möglichst wenig Informationsübertragung: Die Informationsübertragung soll minimal sein. Beispielsweise ist eine Verbindung zwischen Switch A und B unterbrochen. Diese Situation soll nicht verursachen, dass die ganze Netztopologie bezüglich A und B nochmal transportiert wird.

4.4.2 Schema-Definition

In diesem Abschnitt wird das Schema für Export/Import-XMLs definiert.

XML Schema 0.0.1

Die Grundidee ist, den Datenaustausch via XML-Dateien zu realisieren. Der Netzdoku-Exporter und OTRS-Importer sollen einem identischen XML Schema folgen. Ein Teil der XML Definition ist wie folgten XML-Abschnitt dargestellt. Die vollständige Schemadefinition befindet sich in der Abgabeversion dieser Ausarbeitung.

Listing 4.1: XML Schema Definition

```
<xs:element name="ITSMExport">
    <xs:annotation>
      <xs:documentation>External schema, base of interface contract.
      </xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="MessageHeader" type="ITSMHeaderType"/>
        <xs:element name="MessageBody">
```

```
            <xs:complexType>
              <xs:sequence>
                <xs:element name="User" maxOccurs="unbounded">
                  <xs:complexType>
                    <xs:complexContent>
                      <xs:extension base="ITSMUserType">
                        <xs:sequence>
                          <xs:element ref="Relation" maxOccurs="unbounded"/>
                        </xs:sequence>
                      </xs:extension>
                    </xs:complexContent>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Relation">
      <xs:complexType>
        <xs:complexContent>
          <xs:extension base="RelationType">
          <xs:sequence>
                    <xs:element name="CIAccount" type="AccountType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CIOrganization" type="OrganizationType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CIInstitut" type="InstitutType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CIWorkGrup" type="WorkGrupType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CIGelaende" type="GelaendeType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CIGebaeude" type="GebaeudeType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CIRaum" type="RaumType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CIPerson" type="PersonType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CIKomponent" type="KomponentType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CIPort" type="PortType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CIVlan" type="VlanType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CISubNetz" type="SubNetzType"
```

```
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CISubSubNetz" type="SubSubNetzType"
                    minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="CIProdukt" type="ProduktType"
                    minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:complexType name="ITSMHeaderType">
  <xs:annotation>
    <xs:documentation>"Mssage Header"</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="Sender" type="xs:string">
      <xs:annotation>
        <xs:documentation>"Sender = Netzduku or physik, Receiver = OTRS"
        means export from Netzdoku or Physik, import into OTRS
      </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Receiver" type="xs:string">
      <xs:annotation>
        <xs:documentation>"Sender = Netzduku or physik, Receiver = OTRS"
        means export from OTRS, import into Netzdoku</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Source" type="xs:string" minOccurs="0">
      <xs:annotation>
        <xs:documentation>The source system of the delivered data.
        This data is used as "SOURCE" parameter for parametrizable
        configurazion settings.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="TimeStamp" type="xs:string"/>
    <xs:element name="Id" type="String25" minOccurs="0"/>
    <xs:element name="MessageType" type="xs:string" minOccurs="0"/>
    <xs:element name="OrganizationCode" type="String10">
      <xs:annotation>
        <xs:documentation>Organization mapped to char-2 ISO code, e.g. "ND"
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="SchemaVersion" type="ITSMSchemaVersion">
      <xs:annotation>
        <xs:documentation>The schema version that is the base for the document.
```

```
          Only supported versions are allowed.</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="ControlInformation" type="ITSMControlInformation"
      minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation>Control switches,
          at the moment not relevant for DMS.</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
```

Eine genauere Definition des CI-Typs Raum zeigt:

Listing 4.2: Beispiel: XML Node Definition

```
  <xs:complexType name="RaumType">
    <xs:annotation>
      <xs:documentation>Raum type</xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:element name="RaumIntgID" type="String100" minOccurs="0"/>
      <xs:element name="RaumID" type="String100" minOccurs="0"/>
      <xs:element name="RaumName" type="String100" minOccurs="0"/>
      <xs:element name="RaumNo" type="String100" minOccurs="0"/>
      <xs:element name="Stockwerk" type="String100" minOccurs="0"/>
      <xs:element name="Rack" type="String100" minOccurs="0"/>
      <xs:element name="Telephone" type="String100" minOccurs="0"/>
      <xs:element name="Fax" type="String100" minOccurs="0"/>
      <xs:element name="LagePlan" type="xs:base64Binary" minOccurs="0"/>
      <xs:element name="GebaeudeName" type="String100" minOccurs="0"/>
      <xs:element name="Comments" type="String2000" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
```

Das Schema enthählt zwei Hauptteile: MessageHeader und MessageBody. Der Knoten MessageHeader enthält die Kontrollinformationen über die Transaktionen. Der MessageBody enthält die CIs und deren Beziehungen. Der folgende XML-Anschnitt zeigt ein Beispiel-XML.

Listing 4.3: Beispiel: XML Abschnitt

```
<ITSMExport>
  <MessageHeader>
    <Sender>NETZDOKU</Sender>
    <Receiver>OTRS</Receiver>
    <Source>NETZDOKU</Source>
    <TimeStamp>2009-01-19T06:57:19</TimeStamp>
    <MessageType>NDEXPORT</MessageType>
    <OrganizationCode>ND</OrganizationCode>
```

```
    <SchemaVersion>0.0.1</SchemaVersion>
  </MessageHeader>
  <MessageBody>
    <User>
      <UserID>Admin</UserID>
      <UserPriority>1</UserPriority>
      <Relation>
        <RelationIntgID>7-4648</RelationIntgID>
        <RelationName>KOMPONENT-PORT</RelationName>
        <RelationDir>TOPDOWN</RelationDir>
        <RelationType>HABEN</RelationType>
        <GegenRelationType>GEHABTVON</GegenRelationType>
        <CIKomponent>
          <KomponentIntgID>KO_7</KomponentIntgID>
          <KomponentIntgID>KO_701</KomponentIntgID>
          <Alias>swv1-1b1</Alias>
          <TicketNo>1419</TicketNo>
          <Comments>1.OG(Hr. Ebert)</Comments>
       </CIKomponent>
        <CIPort>
          <PortIntgID>PP4648</PortIntgID>
          <IsPhysik>Y</IsPhysik>
          <IsLogic>N</IsLogic>
          <Bezeichung>J1</Bezeichung>
          <Zielading>Y</Zielading>
          <Schnittstelle>1000BaseSX</Schnittstelle>
          <Geschwindigkeit>1 GB/s</Geschwindigkeit>
        </CIPort>
      </Relation>
  </MessageBody>
</ITSMExport>
```

MessageHeader/Sender bedeutet die Quelldatenbank (Netzdoku im Beispiel) und Receiver ist die Zieldatenbank (OTRS im Beispiel). MessageHeader.Source zeigt, woher die Information kommt. MessageHeader.TimeStamp ist der aktuelle Zeitpunkt, wann das XML generiert wurde. MessageHeader.Id ist die Folgenummer der XML-Dateien. MessageHeader.MessageType beschreibt den Typ der XML-Datei. Der Typwert „NDEXPORT“ bedeutet, dass die XML-Datei ein Ergebnis des Exports aus Netzdoku ist. OrganizationCode z.B. „ND“ bedeutet Netzdoku. SchemaVersion ist die aktuelle Version des XSD. Der OTRS-Importer prüft die Version und importiert nur die XML-Dateien, die eine gültige Schemaversion haben. Dies hilft der Syncsoftware festzustellen, ob sie die XML-Dateien bearbeiten kann.
MessageBody beinhaltet mehrere CIs und die Beziehungen zwischen CIs. Alle Informationen werden in einer Form Relation-Entitäten formuliert. Der Knoten Relation beschreibt die Beziehung zwischen zwei Entitäten. Momentan sind in XSD Version 0.0.1 15 Typen von Entitäten definiert: CIAccount, CIOrganization, CIInstitut, CIWorkGrup, CIGelände, CIGebäude, CIRaum, CIPerson, CIKomponent, CIPort, CIVlan, CISubNetz, CISubSubNetz,

CILeitung, CIProdukt. Diese 15 Entitätentypen entsprechen den 15 Typen von CIs in OTRS, die tatsächlich CIs beinhalten. Die Bedeutung und Definition dieser 15 Typen wurden in Kapitel 3 beschrieben.

Beziehungen

Die kleinste Einheit in XML ist die Relation mit zwei Entitäten. Die zwei Entitäten sind zwei unterschiedliche CIs und die Relation ist die Beziehung dazwischen. Die Relation erscheint immer als Paar. In Schema 0.0.1 werden insgesammt 10 Paare-Beziehungen definiert:

- BESTEHENAUS und TEILVON
  Beispiel: Eine Organisation besteht aus Instituten. Ein Institut ist ein Teil von einer Organisation.

- BESITZT und BESITZTVON
  Beispiel: Ein Router besitzt 8 Ports, ein Port ist besitzt von einem Router.

- STELLENEIN und ARBEITENFÜR Beispiel: Eine Arbeitsgruppe stellt eine Person ein, eine Person arbeitet für eine Arbeitsgruppe.

- BETREUEN und BETREUETVON
  Beispiel: Ein Netzverantwortlicher betreut ein Gebäude. Ein Gebäude ist von einem Netzverantwortlichen betreut.

- SITZENIN und PLATZANBIETENFÜR
  Beispiel: Eine Person sitzt in einem Raum, ein Raum bietet einen Arbeitsplatz für eine Person an.

- SICHBEFINDENIN und BEINHALTEN
  Beispiel: Eine Netzkomponente befindet sich in einem Raum, ein Raum beinhaltet Komponenten.

- DOWNTO und UPTO
  Diese zwei Typen beschreiben die Beziehung zwischen zwei Komponenten in unterschiedlichen Ebenen. Ein Netzkomponente befindet sich in einer höheren Stufe bedeutet, dass sie näher zum Netzkern ist. Wir haben z.B. einen Switch1 in LRZ und einen Switch2 in Fakultät Physik. Eine Verbindung zwischen den Ports dieser Switch wurde eingerichtet. Wir bezeichnen die Verbindung mit einer DOWNTO-Beziehung, wenn die Verbindung mit dem Port von Switch1 beginnt und mit dem Port von Switch2 endet.

- LINKTO
  LINKOTO bezeichnet die Beziehung zwischen zwei Komponenten, die auf der gleichen Ebene sind.

- NORELATION
  NORELATION dient nur zum Entitätenexport. Es gibt keine Beziehung zwischen den Entitäten. Die CIs werden nur in die Zieldatenbank gebracht, ohne Beziehungen herzustellen.

4.4.3 Synchronisationsvorgang: Export aus der Netzdoku

Wie oben vorgestellt wurde, ist die kleineste Informationseinheit in XML eine Beziehung mit zwei CIs. Die Netzdoku-Datenbank wurde nach dem Entitäts-Relationen-Prinzip entwickelt. Bevor der Netzdoku-Exporter die CIs mit ihren Beziehungen in XML schreibt, muss er solche Information aus der Netzdoku erfassen.

Netzdoku-Datenbankstruktur

Die folgende Abbildung 4.6 zeigt die Datenbankstruktur der Netzdoku-Datenbank:

Erfassung der CIs und Beziehungen per Netzdoku-Exporter

CI Erfassung:

- CIRaum
  CIs vom Typ Raum werden aus den Tabellen RAUM und UNTERBEZIRK zusammengefasst. Das Mapping zwischen Datenbankfeldern und XML CIRaum Feldern ist wie folgt:

XML Feld	Netzdoku Feld	Beispielswert	Bemerkung
RaumIntgID	Raum.ID	373772	
RaumID			Beim Export von Netzdoku leer, wird beim Export von OTRS besetzt
RaumName	Unterbezirk.Ort, Raum.Nummer	Lrz, Hauptgebäude, S450	Kombination von zwei Feldern
RaumNo	Raum.Nummer	S450	
Stockwerk	Raum.Stockwerk	4	
Rack	Raum.Rack	8k	
Telephone	Raum.Telefon	089-282828	
Fax			
LagePlan			
GebäudeName	Unterbezirk.Ort	Lrz Hauptgebäude	
Comments			

Tabelle 4.1: Mapping zwischen Netzdoku-DB und XML Feld

  Wir nehmen hier das Mapping zwischen XML CIRaum und den entsprechenden Datenbanktabellen als ein Beispiel. Die vollständigen Mappings zwischen den anderen XMLfeldern und Datebankfeldern befinden sich in der Abgabeversion dieser Ausarbeitung.

- CIOrganisation aus dem Feld Institut.Organisation
- CIInstitut aus Tabelle Institut
- CIGebäude aus Tabelle Unterbezirk
- CIPerson aus Tabelle Person

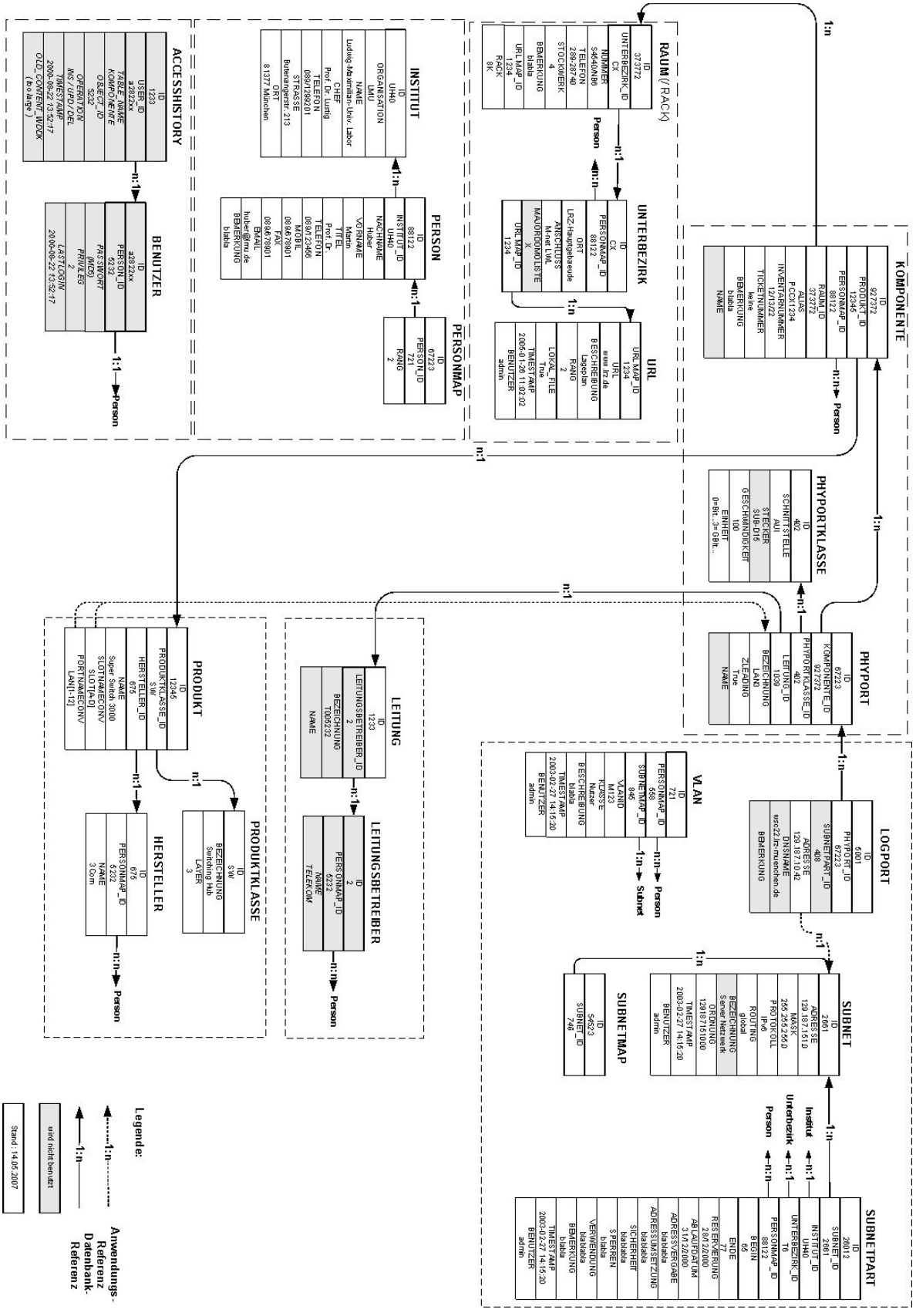


Abbildung 4.6: Netzdoku-Datenbankstruktur

- CIKomponente aus Tabelle Komponente
- CIPort aus Tabelle Phyport, Phyportklasse und Logport
- CIVlan aus Tabelle VLAN
- CISubNetz aus Tabelle Subnet
- CISubSubNetz aus Tabelle Subnetpart
- CIProdukt aus Tabelle Produkt, Produktklasse und Hersteller

Die Beziehungen zwischen CIs

- CIInstitut und CIPerson
  Eine Person arbeitet für ein Institut. Die Beziehung erfasst man durch den SQL Satz: select PERSON.INTSTITUT_ID, PERSON.ID from PERSON
- CIRaum und CIKomponente
  Eine Netzkomponente befindet sich in einem Raum.
  SQL Satz: select KOMPONENTE.ID, KOMPONENETE.RAUM_ID from KOMPONENTE
- CIGebäude und CIPerson
  Ein Mitarbeiter arbeitet in einem Gebäude.
  SQL Satz: select UNTERBEZIRK.ID, PERSONMAP.PERSON_ID from UNTERBEZIRK, PERSONMAP where UNTERBEZIRK.PERSONMAP_ID = PERSONMAP.ID
- CIKomponente und CIPort
  Ein Router besitzt mehrere Ports.
  SQL Satz: select PHYPORT.KOMPONENTE_ID, PHYPORT.ID from PHYPORT
- CIPort und CIPort
  Es gibt zwei Beziehungen zwischen CIPort und CIPort.
  Beziehung 1: MAPTO
  Mehrere logische Ports werden zu einem physischen Port abgebildet.
  SQL Satz: select LOGPORT.PHYPORT_ID, LOGPORT.ID from LOGPORT
  Beziehung 2: LINKTO
  Eine Verbindung wird zwischen zwei physischen Ports hergestellt. Zwei Ports, die die gleiche Leitung_ID haben, sind durch die Leitung verbunden.
  SQL Satz: select PHYPORT.LEITUNG_ID from PHYPORT group by LEITUNG_ID select PHYPORT.ID from PHYPORT where LEITUNG_ID = :LEITUNGID order by ID
- CIProdukt und CIKomponente
  An der Stelle einer Netzkomponente wird ein Produkt eingesetzt.
  SQL Satz: select KOMPONENTE.PRODUCKT_ID, KOMPONENTE.ID from KOMPONENTE
- CIPerson und CIKomponente
  Ein Mitarbeiter betreut mehrere Netzkomponenten. Es kann auch sein, dass eine Komponente von mehreren Mitarbeitern betreut wird.

SQL Satz: select KOMPONENTE.ID, PERSONMAP.PERSON_ID from KOMPONENTE, PERSONMAP where KOMPONENTE.PERSONMAP_ID = PERSONMAP.ID

- CIPerson und CIVlan
  Ein Mitarbeiter kümmert sich um die VLANs.
  SQL Satz: select VLAN.ID, PERSONMAP.PERSON_ID from VLAN, PERSONMAP where LVAN.PERSONMAP_ID = PERSONMAP.ID

- CIVlan und CISubnetz
  Ein großes VLAN kann man auf mehrere Subnetze aufteilen.
  SQL Satz: select VLAN.ID, SUBNETMAP.SUBNET_ID from VLAN, SUBNETMAP where VLAN.SUBNETMAP_ID = SUBNETMAP.ID

- CISubnetz und CISubsubnetz
  Ein Subnetz kann man auch auf mehrere Subnetzteile aufteilen.
  SQL Satz: select SUBNET.ID, SUBNETPART.ID from SUBNET, SUBNETPART where SUBNET.ID = SUBNETPART.SUBNET_ID

- CIInstitut und CISubsubnetz
  Ein Subnetzteil kann so groß sein, dass es mehrere Institute beinhaltet.
  SQL Satz: select SUBNETPART.INSTITUT_ID, SUBNETPART.ID from SUBNETPART

- CIGebäude und CISubsubnetz
  Das Gebäude ist ein geographischer Begriff. Ein Teil des Subnetzes lässt sich auf mehrere Gebäude verteilen.
  SQL Satz: select SUBNETPART.UNTERBEZIRK_ID, SUBNETPART.ID from SUBNETPART

- CIPerson und CISubsubnetz
  Ein Subnetzteil wird von mehreren Mitarbeitern betreut. Ein Mitarbeiter kann auch mehrere Subnetzteile betreuen.
  SQL Satz: select SUBNETPART.ID, PERSONMAP.PERSON_ID where SUBNETPART.PERSONMAP_ID = PERSONMAP.ID

- CIGebäude und CIRaum
  Die Räume liegen in einem Gebäude.
  SQL Satz: select RAUM.ID, RAUM.UNTERBEZIRK_ID from RAUM

- CIVlan und CIPhyPort
  In der Netzdoku-Datenbank gibt es Tabellen, in der die Zuordnungen von VLAN und physischen Ports gespeichert werden. Diese Tabelle wird jeden Tag automatisch über die entsprechenden Komponenten erneuert. Aus Sicherheitsgründen konnten die Daten von VLAN und Port in dieser Arbeit nicht erfasst werden. Sie kommen erst in der Systemerweiterungen vor.

Erweiterungen der Datenbanken

Um die Synchronisation zwischen Netzdoku und OTRS durchzuführen, muss man noch die Datenbankstruktur anpassen.

Netzdoku-Datenbank

Die Änderungen in der aktuellen Datenbank sind grundsätzlich die Zeiger, die besagen, ob die CIs schon mal erfasst und exportiert wurden. Wenn der Zeiger z.B. RAUM.SENDED true ist, bedeutet das, die Information von diesem Raum ist einmal exportiert worden, dadurch werden sie nicht nochmals vom Netzdoku-Exporter exportiert.
Teil 1 in der Abbildung 4.7: Zu dieser Änderung gehört noch ein Trigger in der Oracle-

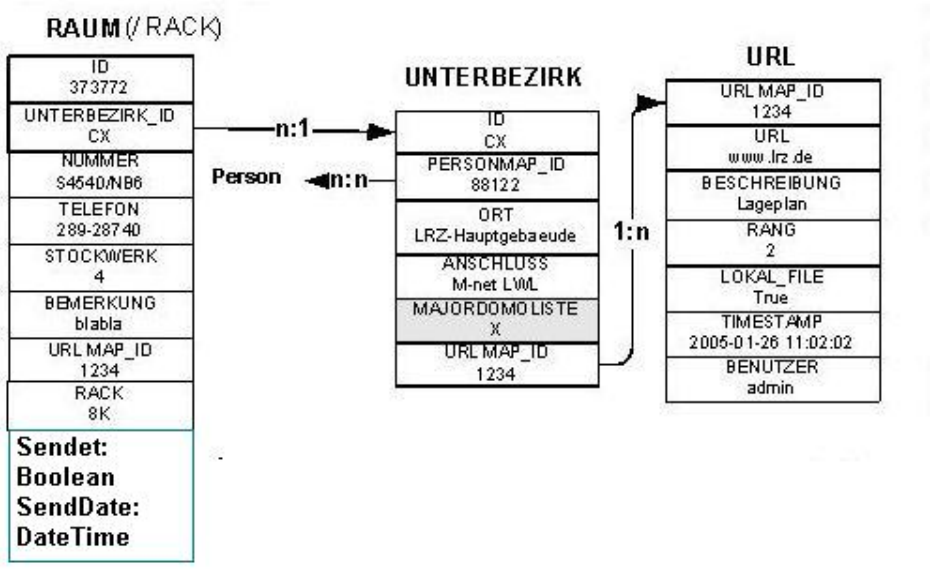


Abbildung 4.7: Teil 1: Änderung an der Netzdoku-Datenbankstruktur

Datenbank über die Tabelle RAUM. Wenn ein Raum ganz neu angelegt oder die Details von dem Raum geändert werden, ändert der Trigger das Feld RAUM.SENDED auf false. Der Raum wird vom Netzdoku-Exporter als nicht exportiert gefunden und dann wiederum exportiert. Das SENDDATE wird vom Exporter auf die Systemzeit gesetzt, wenn der Raum exportiert wird und ist sonst leer.

Teil 2 in der Abbildung 4.8: Ähnlich wie für Teil 1 werden noch 3 weitere Trigger angelegt werden. Trigger 1 ist über die Tabelle INSTITUT, die beim Anlegen oder Ändern der Institute den Zeiger INSITUT.SENDED als false setzt. Trigger 2 ist über die Tabelle PERSON, die beim Anlegen oder Ändern der Personinfos den Zeiger PERSON.SENDED als false setzt. Trigger 3 ist über die Tabelle PERSON, die bei der Änderung des Feldes PERSON.INSTITUT_ID den Zeiger RECONNECTED als true setzt. Der Zeiger RECONNECTED besagt, dass die Zugehörigkeit der Person zu Institut geändert wird und es soll eine Relationsänderung zwischen Institut und Person exportiert werden.

Teil 3 in Abbildung 4.9:

- Trigger 1 auf der Tabelle Komponente
  Wenn die KOMPONENTE.PRODUKT_ID geändert wird, d.h. ein neues Produkt eingesetzt wird, wird der PRODUKT-TRIGGER gleich true gesetzt. Der Netzdoku-Exporter muss die Beziehung zwischen Komponente und Produkt exportieren.

- Trigger 2 auf Tabelle Komponente
  Wenn PERSONMAP_ID geändert wird, wird PERSONTRIGGER gleich true gesetzt.

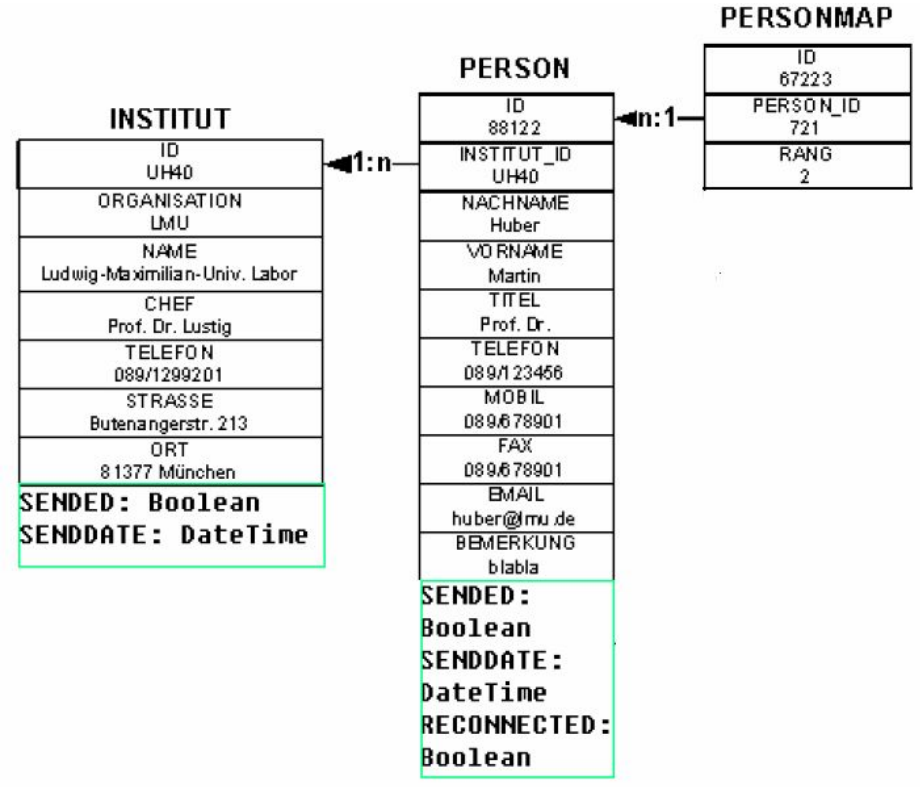


Abbildung 4.8: Teil 2: Änderung an der Netzdoku-Datenbankstruktur

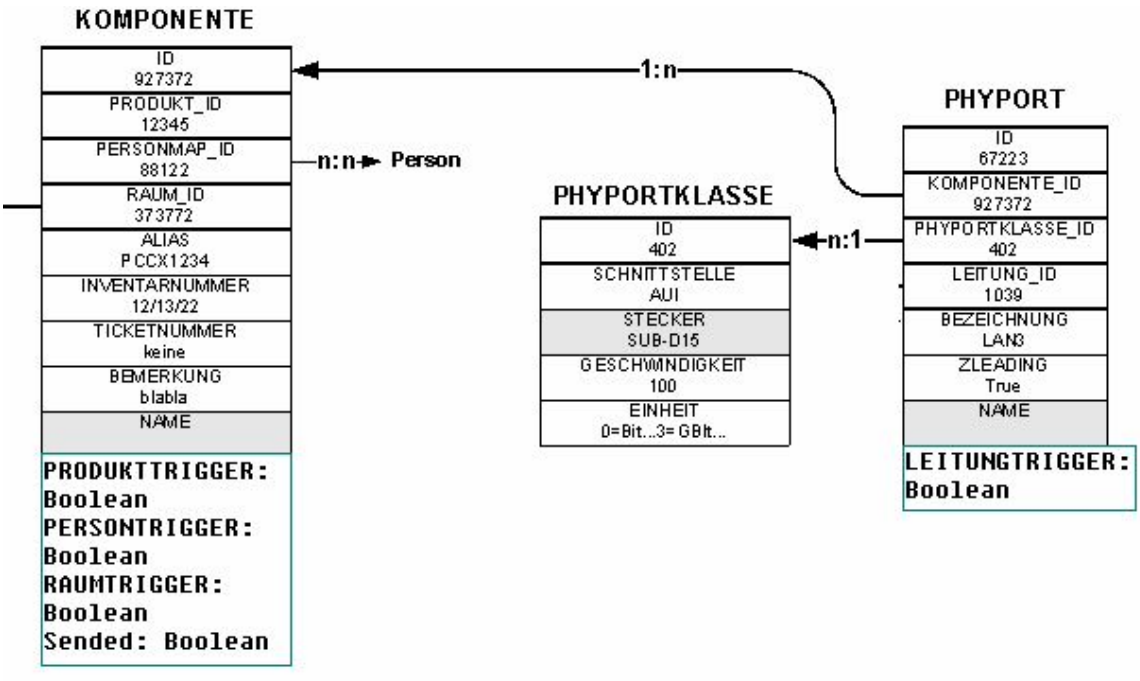


Abbildung 4.9: Teil 3: Änderung an der Netzdoku-Datenbankstruktur

- Trigger 3 auf Tabelle Komponente
  Wenn RAUM_ID geändert wird, wird Sended gleich false gesetzt.

- Trigger 4 auf Tabelle Phyport
  LEITUNG_ID wird geändert, wenn eine neue Verbindung herstellt wird. Dieser Zeiger dient der Anforderung, dass die Verbindung in OTRS transportiert wird.

Teil 4 4.10:

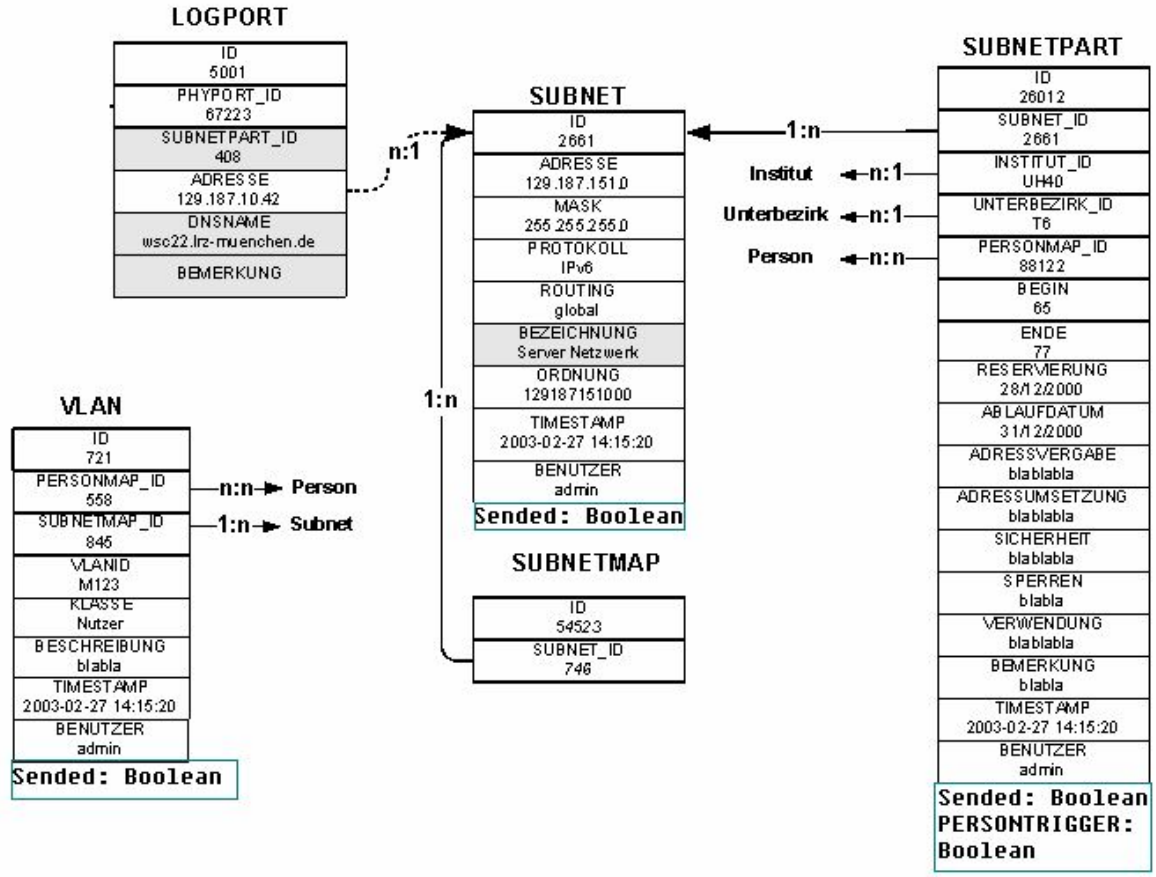


Abbildung 4.10: Teil 4: Änderung an der Netzdoku-Datenbankstruktur

- Trigger 1 auf Tabelle VLAN für den Zeiger Sended.

- Trigger 2 auf Tabelle SUBNET für den Zeiger Sended.

- Trigger 3 auf Tabelle SUBNETZPART für den Zeiger Sended.

- Trigger 4 auf Tabelle SUBNETZPART für den Zeiger PERSONTRIGGER. Wenn PERSONMAP_ID geändert wird, wird der Zeiger vom Trigger 4 auf true gesetzt. Die Änderungen über SUBNET_ID, INSTITUT_ID und UNTERBEZIRK_ID werden ignoriert, da solche Werte niemals geänert werden.

Der erste Export von der Netzdoku

Der erste Export aus der Netzdoku muss die kompletten CIs und Beziehungen auf einmal exportieren. Die entsprechenden Zeiger in der Datenbank werden auch initialisiert. Diese Aufgabe wird durch das Programm Netzdoku-Exporter durchgeführt. Der Sourcecode des Programms befinden sich in der Abgabeversion dieser Ausarbeitung. Das Programm exportiert die CIs und Beziehungen nach oben vorgesteller Logik und speichert die XML-Dateien unter verschiedenen Verzeichnissen. Wir nehmen die Export-Ergebnisse von CIKomponente und Beziehung zwischen CIKomponente und CIPhyPort an.
Der folgende XML-Abschnitt zeigt einen Abschnitt des Exportergebnis von CIKomponente:

```
<Relation>
  <RelationType>NORELATION</RelationType>
  <CIKomponent>
    <KomponentIntgID>KO_7</KomponentIntgID>
    <Alias>swh1-2qe</Alias>
    <TicketNo>513</TicketNo>
    <Comments>Bau 101/4 W.,m.15xTPs,100MB Server,VL961</Comments>
  </CIKomponent>
  <CIKomponent>
    <KomponentIntgID>KO_11</KomponentIntgID>
    <Alias>swn1-0sk</Alias>
    <TicketNo>1402</TicketNo>
  </CIKomponent><CIKomponent>
```

Der XML-Abschnitt zeigt uns drei Komponenten, die aus der Netzdoku erfasst wurden. Das XML-Feld RelationType wird als „NORELATION“ gesetzt, dieser Wert hilt dem OTRS-Importer, dass er dieses XML-Fragment als CI-Resourcen erkennt.

Der zweite XML-Abschnitt zeigt uns die Beziehungen zwischen CIKomponente und CI-PhyPort, die aus der Netzdoku erfasst wurden.

```
<Relation>
  <RelationIntgID>7-4648</RelationIntgID>
    <RelationName>KOMPONENT-PORT</RelationName>
    <RelationDir>TOPDOWN</RelationDir>
    <RelationType>HABEN</RelationType>
    <GegenRelationType>GEHABTVON</GegenRelationType>
    <CIKomponent>
      <KomponentIntgID>KO_7</KomponentIntgID>
    </CIKomponent>
    <CIPort>
      <PortIntgID>PP4648</PortIntgID>
    </CIPort>
</Relation>
<Relation>
  <RelationIntgID>11-4626</RelationIntgID>
  <RelationName>KOMPONENT-PORT</RelationName>
  <RelationDir>TOPDOWN</RelationDir>
```

```
  <RelationType>HABEN</RelationType>
  <GegenRelationType>GEHABTVON</GegenRelationType>
  <CIKomponent>
    <KomponentIntgID>KO_11</KomponentIntgID>
  </CIKomponent>
  <CIPort>
    <PortIntgID>PP4626</PortIntgID>
  </CIPort>
</Relation>
```

Die RelationIntgID wird von der Kombination von KomponenteID und PortID generiert. Dieser Abschnitt zeigt, dass die Komponente mit ID KO_7 ein Port besitzt. Die genaue Information von diesem Port muss man noch in der XML-Datei des Exports von CIPort untersuchen.

4.4.4 Synchronisationsvorgang: Export aus Physik

Die andere Datenquelle für OTRS ist die Datenbank der Physik. In diesem Schnitt schauen wir an, wie die Daten in der Physik-Datenbank erfasst und exportiert werden.

Datenbankstruktur der Physik-Datenbank

Die Datenquelle der Datenbank der Physik ist eine Tabelle wie in Abbildung 4.11 dargestellt. Wie die Abbildung 4.11 zeigt ist, ist das Feld patch_room der Raum, wo der Patch liegt; Die

Name	Type
Primary Index	Id
Id	int(11)
key	varchar(255)
tag	varchar(255)
patch_room	varchar(255)
patch_rack	varchar(255)
patch_panel	varchar(255)
patch_port	varchar(255)
room_number	varchar(255)
room_port	varchar(255)
vlan	varchar(255)
gruppe	varchar(255)
switch_module	varchar(255)
switch_port	varchar(255)
modified	timestamp
md5	varchar(255)
comment	varchar(255)

Abbildung 4.11: Physik Datenbankstruktur

Patches in einem Raum werden nummeriert und das Feld patch_rack ist die Nummer des Patches. Ein Patch kann mehrere Panels erhalten; Das Feld patch_port ist die Portnummer

auf einem Patchpanel. Der Patchport wird mit einer Dose in einem Raum verbunden, die von den Feldern room_number und room_port beschrieben wird. Das Feld vLAN beschreibt die IDs von VLAN, die auf dem Switchport eingerichtet werden. Die Felder switch_module und switch_port beschreiben die Ports von Switches, die mit dem Patchports verbunden werden. Die Gruppe ist die Arbeitsgruppe, die die Patchports benutzt.

Erfassung der CIs und Beziehungen der Physik Datenbank

Aus dieser Tabelle kann man folgende CIs erfassen:

- CIKomponente
  Hier gibt es drei Typen von Komponenten: Patch, Switch, Netzdose.
- CIRaum
  Die Räume werden aus Feldern patch_room und room_number erfasst.
- CIPort
  Ähnlich wie CIKomponente gibt es auch zwei Typen von Port: Patchport und Switchport.
- CIVlan aus dem Feld VLAN
- CIArbeitsgruppe aus dem Feld gruppe

Die Beziehung zwischen CIs:

- CIPatch und CIPort (1:n Beziehung)
- CIPatch und CIRaum (n:1 Beziehung)
- CIPort und CIPort (Switchport und Patchport, 1:1 Beziehung)
- CISwitch und CIPort (1:n Beziehung)
- CIVlan und CIPort (1:n Beziehung)
- CIArbeitsgruppe und CIRaum (n:m Beziehung)
- CIKomponente und CIPort (Dose und Patchport, 1:1 Beziehung)

Export aus der Physik-Datenbank

Ein Programm PHY-Exporter wurde entwickelt. Dieses Programm exportiert die CIs und Beziehungen nach der oben vorgestellten Logik. Ein Beispiel des XML-Ausschnitts ist in Abbildung 4.12 dargestellt. Die Abbildung zeigt die Information von einer Netzdose in der Physik-Fakultät und die Verbindungsinfo zwischen dieser Netzdose und einem lokalen Switchport.

4.4.5 Synchronisationsvorgang: Import ins OTRS

In den obigen zwei Abschnitten haben wir alle CIs und Beziehungen aus beiden Datebanken erfasst und exportiert. In diesem Abschnitt importieren wir die exportierten XML-Dateien in OTRS, damit in OTRS diese Daten auch verwaltet werden können. Zuerst schauen wir an, wie die OTRS-CMDB strukturiert wird und welche Änderungen durchgeführt werden müssen.

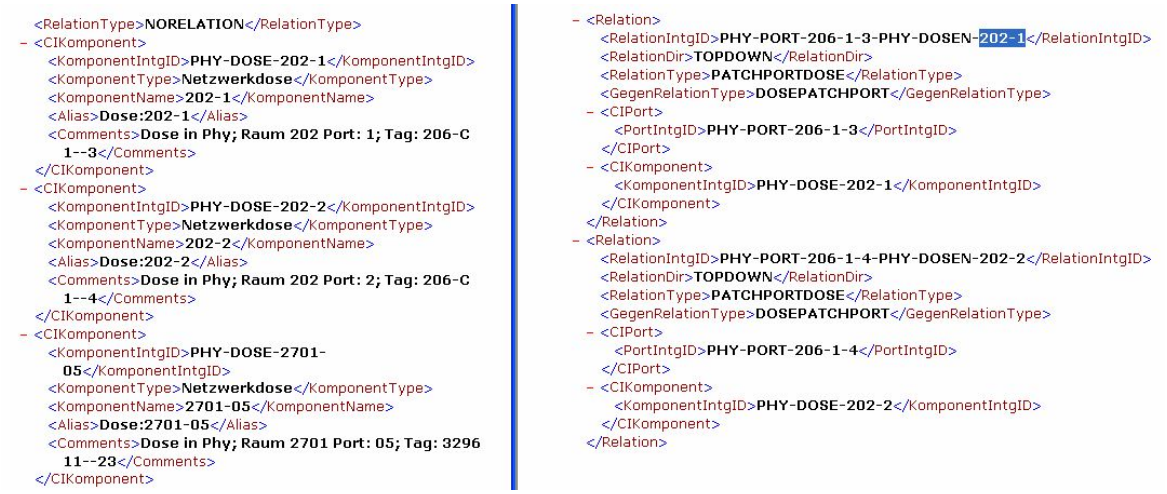


Abbildung 4.12: Export aus Physik-Datenbank

OTRS Datenbank

Für das CM in OTRS sind insgesammt 5 Tabellen relevant: Tabelle ConfigItem Abbildung 4.13:

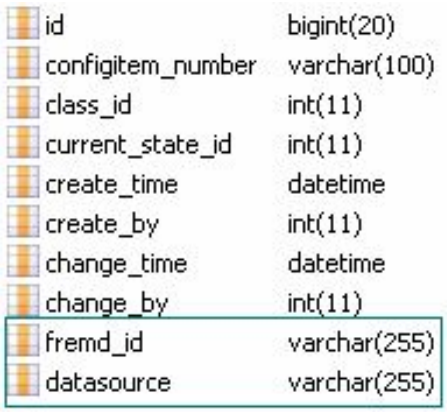


Abbildung 4.13: Änderung an OTRS Datenbankstruktur

Die Felder id und configitem_number bestimmen eindeutig das CI in der CMDB. Ein Wert von configitem_number ist normalerweise 11 Zeichen lang. Die ersten beiden Zeichen sind „10“. Die folgenden vier sind gleich wie class_id. Die übrigen bilden den Counter von CIs dieses Typs. Das Feld class_id kennzeichnet die CI-Typen. Jeder Typ bekommt eine eindeutige class_id. Falls wir im OTRS Interface manuell einen Typ definieren, wird die class_id vom System intern gesetzt. Deshalb ist es vorteilhaft, dass wir per vordefinierten Skript das CM installieren (siehe OTRS Installation Methode 1 in Abschnitt 4.2). In diesem Fall bleiben die class_ids permanet und der OTRS Importprozess ist leichter zu implementieren. Wir fügen zu dieser Tabelle zwei neue Felder hinzu: fremd_id und datasource. Die fremd_id ist die ID des Zielsystems (z.B. ID von Netzdoku). Jedes mal wenn ein CI importiert wird, wird diese ID geprüft. Falls die ID existiert, wird der Import des CIs ignoriert. Die Datasource zeigt,

woher das CI kommt. Momentan ist die Datasource entweder „Netzdoku“ oder „Physik“.

Tabelle configitem_counter 4.14: Für jeden CI-Typ wird ein Counter eingesetzt. Wenn ein

class_id	counter_type	counter
106	AutoIncrement	1
108	AutoIncrement	1
109	AutoIncrement	1601
110	AutoIncrement	19
111	AutoIncrement	1
112	AutoIncrement	477
113	AutoIncrement	2250
231	AutoIncrement	904
232	AutoIncrement	4347
233	AutoIncrement	1059
234	AutoIncrement	7176
235	AutoIncrement	2561
242	AutoIncrement	1848
243	AutoIncrement	3622
244	AutoIncrement	180

Abbildung 4.14: Änderung an OTRS Datenbankstruktur

CI hinzugefügt wird, wird der Counter von diesem Typ um 1 hochgezählt.

Tabelle configitem_definition 4.15: Die Definitionen von CI-Eigenschaften werden in dieser

id	int(11)	No
class_id	int(11)	No
configitem_definition	longblob	No
version	int(11)	No
create_time	datetime	No
create_by	int(11)	No

Abbildung 4.15: Änderung an der OTRS Datenbankstruktur, Tabelle configitem-definition

Tabelle gespeichert. Wenn die Definition geändert wird, wird sie aber nicht gelöscht, sondern es wird die Versionsid aktuallisiert.

id	bigint(20)	No
configitem_id	bigint(20)	No
name	varchar(250)	No
definition_id	int(11)	No
state_id	int(11)	No
create_time	datetime	No
create_by	int(11)	No

Abbildung 4.16: Änderung an der OTRS Datenbankstruktur, Tabelle configitem-version

Der Rekord von dieser Tabelle wird durch das Feld configitem_id mit der Tabelle configi-

tem verknüpft. Das Feld name ist eine allgemeine Eigenschaft von CIs. Es liefert die textuelle Beschreibung von CIs.

xml_type	xml_key	xml_content_key	xml_content_v...
ITSM::ConfigItem::Archiv::111	4	[1]{'Version'}[1]{'Bezirk'}[1]{'TagKey'}	<BLOB>
ITSM::ConfigItem::Archiv::111	4	[1]{'Version'}[1]{'Lageplan'}[1]{'Content'}	<BLOB>
ITSM::ConfigItem::Archiv::111	4	[1]{'Version'}[1]{'Lageplan'}[1]{'TagKey'}	<BLOB>
ITSM::ConfigItem::Archiv::111	4	[1]{'Version'}[1]{'Link'}[1]{'Content'}	<BLOB>
ITSM::ConfigItem::Archiv::111	4	[1]{'Version'}[1]{'Link'}[1]{'TagKey'}	<BLOB>
ITSM::ConfigItem::Archiv::111	4	[1]{'Version'}[1]{'TagKey'}	<BLOB>

Abbildung 4.17: Änderung an der OTRS Datenbankstruktur, Tabelle xml-storage

Da die Definitionen der CI-Typen halb-dynamisch sind, werden die Eigenschaften von CIs auch in der Tabelle XMl_Storage dynamisch gespeichert. Der xml_type hat die Form „ITSM::Configitem::Archiv::“ plus state_id. Der xml_key ist die ID in der Tabelle configitem_version. Der xml_content_key bezeichnet die Namen von Eigenschaften und xml_content_value beinhaltet die Werte.

Tabelle link_object2:
Diese Tabelle besitzt eine einfache Struktur. Die Beziehungen zwischen CIs werden hier gespeichert.

Import in OTRS

Ein Programm OTRS-Importer wurde entwickelt, um die XML-Dateien importieren zu können. Die Haupteaufgabe dieses Programms ist es, CIs und Beziehungen aus den XML-Dateien auszulesen und die fünf entsprechenden OTRS-Tabellen zu aktualisieren. Das Programm findet man in der Abgabeversion dieser Ausarbeitung. Wir nehmen den Router csr2-2wr als Beispiel und schauen genau an, wie die CIs und Beziehungen ins OTRS importiert werden.

- Schritt 1, in der Tabelle configitem.
  Die Werte werden in folgender Abbildung 4.2 gezeigt:

Feld	**Wert**
ID	die größte ID plus 1
configitem_number	10 232(class id) 000001(select counter from config_item where class_id = 232)
class_id	232 (vom System gegeben, eindeutige ID von CI-Typ)
current_state_id	123 (vom System gegeben, eindeutige ID von CI-Status)
create_time	Systemzeit
create_by	ID des Systembenutzers
change_time	Systemzeit
change_by	ID des Systembenutzers
fremd_id	eindeutige ID von CI aus XML
datasource	Netzdoku

Tabelle 4.2: Tabelle configitem nach einem Import

- Schritt 2, in der Tabelle configitem_count.
  Die Tabelle wird aktualisiert, indem der Counter um 1 hochgezählt wird.
  Sql-Satz: update configitem_count set counter = :counter where class_id = 232

- Schritt 3, in der Tabelle configitem_version. (siehe folgende Abbildung 4.3)

Feld	**Wert**
ID	die größte ID plus 1
configitem_id	configitem.ID
name	csr1-2wr (Aliasname aus XML)
definition_id	configitem_definition.ID (hier werden die Eigenschaften des CI-Typs definiert)
state_id	ID von CI-Status
create_time	Systemzeit
create_by	ID des Systembenutzers

Tabelle 4.3: Tabelle configversion

- Schritt 4, in der Tabelle xml_storage.
  In dieser Tabelle werden mehrere Rekords hinzugefügt.
  Das Feld xml_type ist immer gleich „ITSM::ConfigItem::232“, wobei 232 die Typ-ID bedeutet.
  Das Feld xml_key wird mit der ID in configitem_version befüllt. Die Werte von xml_content_key und xml_content_value sind laut der CI-Definition unterschiedlich. Die folgende Tabelle 4.4 beschreibt einen Teil der hinzufügenden Rekords:

xml ocntent key	**xml content value**
(1)(Version)(1)(Aliasname)(1)(TagKey)	(1)(Version)(1)(Aliasname)(1)
(1)(Version)(1)(Aliasname)(1)(Content)	Alias aus der XML-Datei
(1)(Version)(1)(Inventarnummer)(1)(TagKey)	(1)(Version)(1)(Inventarnummer)(1)
(1)(Version)(1)(Inventarnummer)(1)(Content)	InventarNo aus der XML-Datei
(1)(Version)(1)(Hersteller)(1)(TagKey)	(1)(Version)(1)(Hersteller)(1)
(1)(Version)(1)(Hersteller)(1)(Content)	Hersteller aus der XML-Datei

Tabelle 4.4: Tabelle xmlstorage

- Schritt 5, in der Tabelle link_object2.
  In dieser Tabelle werden die Beziehungen abgebildet. Wenn wir beispielsweise die Beziehung zwischen einer CI-Komponente und einem CI-Produkt hinzufügen, müssen wir folgende Schritte verfolgen:
  1. Wir prüfen in der Tabelle configitem, ob die IDs der CIKomponente und CIProdukt aus der XML-Datein erhalten sind. Wenn wir die IDs nicht finden, muss der Prozess unterbrechen. 2. Wir suchen in der Tabelle configitem die configitem.ID nach der fremd_id. Die zwei configitem.id werden in link_object2 gespeichert. 3. Ein Rekord wird in die Tabelle link_object2 hinzugefügt (siehe 4.5).

Feld	Wert
id	die größte ID plus 1
link_type	Typ der Beziehung
source_class	„ITSMConfigItem“
source_key	die ID des Komponente-CIs in Tabelle configitem
target_class	„ITSMConfigItem“
target_key	die ID des Produkt-CIs in Tabelle configitem
create_time	Systemzeit
create_by	die ID des Systembenutzers
link_id	die eindeutige ID der Beziehung

Tabelle 4.5: Tabelle link_object

Integrierte Daten im OTRS

Die Informationen des Routers csr1-2wr wurden schon in die OTRS-CMDB importiert. Das OTRS repräsentiert dieses CI wie die folgende Abbildung 4.18.

4.5 Zusammenfassung

In diesem Kapital haben wir die ITSM-Lösung von OTRS angeschaut. Es wurde vorgestellt, wie man die CM-Pakete in OTRS installiert und wie man die CIs in OTRS definiert. Anschießend haben wir die CMDB-Struktur der Netzdoku und des Werkzeuges der Fakultät der Physik analysiert und eine Methodik entwickelt, wie die Daten aus der Netzdoku und der Physiks in die OTRS-CMDB integriert werden. Diese Methodik wird durch XML realisiert und verwirklicht eine Schnittstelle, durch die die Daten auch aus verschiedenen anderen Quellen in OTRS integriert werden können. Am Ende haben wir eine vollständige OTRS-CMDB mit Daten befüllt und die Daten via OTRS angeschaut.

Name: csr1-2wr
Status: Komponent
Aliasname: csr1-2wr
Inventarnumme: -
Komponenttyp: Router
Ticketnummer: 8448
Bemerkung:
Manufacturedate: 03.05.2009
Hersteller:
Modell:
Serienummer:
SKU:
Status:
Tag:
Version:
Bezeichung:
Layer:

Abhängige Objekte: Config Item Port

Nummer	Name	Status	Letzte Änderung	Verknüpfungstyp
10234003258	10GBase-LR 10 GB/s	Port	18.02.2009 23:56:14	
10234003259	10GBase-SR 10 GB/s	Port	18.02.2009 23:56:14	
10234003260	10GBase-SR 10 GB/s	Port	18.02.2009 23:56:14	
10234003270	10GBase-SR 10 GB/s	Port	18.02.2009 23:56:14	
10234003272	10GBase-SR 10 GB/s	Port	18.02.2009 23:56:14	
10234003356	1000BaseSX 1 GB/s	Port	18.02.2009 23:56:14	
10234003393	10GBase-SR 10 GB/s	Port	18.02.2009 23:56:14	
10234003554	1000BaseSX 1 GB/s	Port	18.02.2009 23:56:14	
10234003652	10GBase-LR 10 GB/s	Port	18.02.2009 23:56:14	
10234003821	10GBase-SR 10 GB/s	Port	18.02.2009 23:56:14	
10234004017	10GBase-LR 10 GB/s	Port	18.02.2009 23:56:14	

Abhängige Objekte: Config Item Produkt

Nummer	Name	Status	Letzte Änderung	Verknüpfungstyp
10244000027	Catalyst 6509 Router	Komponent	18.02.2009 23:56:14	

Abbildung 4.18: Repräsentation der CIs im OTRS

5 Zusammenfassung

5.1 Evaluation

Wir bewerten die OTRS-CMDB nach im Kapitel 3 vorgestellten Kriterien. Jedes Kriterium wird mit einer der folgenden vier Stufen (+: vorhanden und positiv; 0: vorhanden aber verbesserbar; -: nicht vorhanden; ?: nicht bewertbar) bewertet:

Tabelle 5.1: Evaluation von OTRS

Index	**Kriterium**	**Istzustand**	**Bewertung**
(1)	Die CMDB soll alle Informationen von relevanten CIs und CI-Beziehungen enthalten.	Durch das hier entwickelte Import/Export Interface kann man die Daten von CIs aus verschiendenen Quelldatenbanken erfassen und in OTRS importieren.	+
(2)	Die Personen, die keine entsprechenden Berechtigung haben, sollen vom CM-System isoliert werden. Deshalb soll die Authentifizierungsinformationen auch in der CMDB gespeichert werden.	Istzustand: Das OTRS bietet eine Userauthentifikation an. Nur der Benutzer mit der entsprechenden Berechtigung kann via OTRS CIs bearbeiten. Aber die Rechte kann man nicht CI-spezifisch vergeben	+
(3)	Das CM soll möglicherweise die hierarchische CI-Struktur unterstützen. Beispielsweise haben wir in unserer IT-Umgebung einen CI-Typ Software. Unter Software gibt es weitere Subtypen wie Betriebssysteme, Kommunikationssoftware, Officeserie usw. Die CMDB soll auch für die Abbildung der hierarchischen CI-Struktur geeignet sein.	Das OTRS kann die hierarchische CI-Struktur nicht gut unterstützen. Eine hierarchische Verwaltung ist momentan nicht realisierbar.	-
(4)	Die Informationen von CIs können einfach in die CMDB eingefügt und aus der CMDB entfernt werden.	Istzustand: durch das OTRS ist es machbar.	+

(5)	Da CM normalerweise die Suchefunktion nach CIs bzw. Beziehungen anbieten soll, soll die effziente Suche von der CMDB unterstützt werden.	Suchfunktion ist vom OTRS angeboten, aber nicht effizient genug. Eine Wildcard-Suche ist von OTRS nicht unterstützt. Eine weitere Funktion muss wie OTRS Searcher (siehe 4.3.2) entwickelt werden. Diese Funktion muss in OTRS integriert werden	0
(6)	Das CM-System soll jedes CI eindeutig identifizieren. Dieses CI soll auch in der CMDB eindeutig dargestellt werden.	Die CIs sind im OTRS eindeutig identifizierbar.	+
(7)	Das CM-System verwaltet zentral die Informationen von CIs. Die Struktur von CI-Definitionen, wie ein CI im System dargestellt werden soll, ist in der CMDB enthalten.	Die Konfigurationsinfos werden im OTRS gespeichert und sind dynamisch konfigurierbar.	+
(8)	Das CM-System soll historische Informationen von CIs anliefern, deshalb sollen solche Informationen auch in der CMDB enthalten sein. Wenn die Definition von einem CI-Typ geändert wird, sollen die CIs an die neue Definition angepasst werden und nicht verloren werden. Wir definieren z.B. den CI-Typ Computer mit einer Eigenschaft IP-Adresse und mehrere CIs werden dokumentiert. Wir wollen dann den CI-Typ Computer in zwei Typen splitten: Computer und Netzwerkkarte. Die Eigenschaft IP-Adresse wird vom Typ Computer entfernt und gehört zun neuen Typ Netzwerkkarte. In der CMDB sollen die historischen IP-Adressen von Computer aber nicht gelöscht werden, da sie für die Systemwiederhestellung sehr wichtig sind.	Die historische Informationen werden im OTRS gespeichert.	+

(9)	Die CMDB ist die Basis für andere Prozesse. Falls andere Datenbanken wie z.B. eine Problem-Management-Datenbank bestehen, soll sie ein Interface für andere Datenbank anbieten.	Das OTRS bietet momentan noch keine Interfaces für andere ITIL-Prozesse (außer Incident-, Problemmanagement und Service Level Management) an. Die CMDB muss noch erweitert werden und neue Funktionen müssen noch entwickelt werden. Die Erweiterungen sind möglich.	0
(10)	Die CMDB soll die Änderungen von Daten dokumentieren. Sie sind besonders wichtig für das Change-Management.	Die Änderungen sind dokumentiert.	+
(11)	Die CMDB soll zu bestimmten Zeitpunkten die aktuellen Daten sichern und im Notfall alle CIs in einer gesicherten Version wiederherstellen.	Die Daten sind gesichert aber eine Wiederhestellung der Daten ist nur durch Wiederhersellung der Datenbank möglich.	+

Von diesen 11 Kriterien hat das OTRS achtmal +, zweimal 0 und einmal - bekommen. Wir können jetzt das Ergebniss bekommen, dass das OTRS für unseres CM geeignet ist.

5.2 Weiterentwicklungen

Wir haben die OTRS-Lösung schon bewertet. Jetzt schauen wir an, welche Möglichkeiten es noch gibt, diese Lösung zu verbessern.

- Prozessvervollständigen
  Aus Sicherheitsgründen konnten wir im Rahmen dieser Arbeit die Struktur der Netzdoku-Datenbank nicht ändern. Deshalb konnten wir die Trigger in der Netzdoku-CMDB auch nicht implementieren. Wir hatten noch keine Berechtigung, auf die Ist-Daten der VLANs zuzugreifen. Deshalb ist der Import/Export Prozess noch nicht vollständig implementiert. In der Weiterentwicklungen sollen die fehlenden Daten und Prozessschritte vervollständigt werden, damit das OTRS-CM sinnvoller arbeiten kann.

- Weiterentwicklungen von OTRS-Adapter
  Der OTRS-Adapter wurde spezifisch für die Integration der Daten aus der Netzdoku und des Physik Werkzeuges entwickelt. Aber erwartet wird eine allgemeine Version des Adapters, der auch für die Integration der Daten aus anderen Quellen geeignet ist. In dieser Version sollen die Definitionen von CIs im Adapter konfigurierbar sein. Ein dynamisches Mapping zwischen Datenbankfeldern und XML-Feldern ist auch nützlich. Einige weitere Funktionen wie z.B Protokollierung der Aktionen, die der Adapter durchführt, sollten auch implementiert werden. Dies hilft uns zu analysieren, wenn Import/Export Probleme auftauchen.

- Weiterentwicklung des OTRS
  Die Schwäche von OTRS bzgl. Suchmöglichkeiten sollten verbessert werden. Das OTRS unterstützt z.B. keine Wildcard-Suche. In dieser Arbeit habe wir ein lokal laufendes Programm zur Wildcard-Suche entwickelt. Sinnvoller soll eine in OTRS integrierte Webseite entwickelt werden, um die Funktion zu unterstützen.

5.3 Zusammenfassung

An der TUM wird das Incident-Management durch das Werkzeug Open source Ticket Request System (OTRS) unterstützt. Jetzt gibt es von diesem Werkzeug eine ITSM-Lösung für das Configuration-Management. In dieser Arbeit haben wir untersucht, ob diese Lösung für unsere IT-Umgebung geeignet ist.
Am Anfang dieser Arbeit haben wir die IT-Umgebung angeschaut, wo die CM-Lösung von OTRS eingesetzt wurde. Hauptsächlich wird OTRS in LRZ und TUM zur Verwaltung des MWNs eingesetzt. In LRZ wurde die Netzanwendung NetzDoku entwickelt, um Netzressourcen des MWNs zu verwalten. An der TUM schauen wir das Werkzeug der Physik an, die das lokale Netz verwaltet. Basierend auf diese zwei Informationsquellen implementieren wir die ITSM-Lösung.
In dieser Arbeit haben wir anschließend die Theorie von ITIL, insbesondere des ITIL-Prozesses CM angeschaut. Nach der ITIL-Theorie haben wir die Anforderungen erfasst, die unsere CMDB erfüllen soll. Diese Anforderungen sind auch die wichtigen Kriterien, mit denen wir die ITSM-Lösung evaluieren können.
Mit diesen Anforderungen und den bestehenden IT-Umgebungen haben wir dann die CIs, die zu verwalten sind, und die Beziehungen zwischen CIs erfasst. Wir haben ein UML-Diagramm entwickelt, durch die wir alle CIs mit Eigenschaften und die Beziehungen dazwischen veranschaulichen können. Dieses Diagramm bietet die Basis der idealen CMDB an.
Wir haben dann eine Methodik entwickelt, um eine ideale CMDB aus dem UML-Diagramm aufzubauen. Wir haben anschließend versucht, diese ideale CMDB-Struktur in OTRS abzubilden. Da OTRS bereits eine eigene CMDB-Struktur hat und sich diese CMDB nicht ändern lässt, konnten wir die ideale Struktur nicht direkt im OTRS realisieren. Wir haben jetzt das Problem: das OTRS, die betrachteten CM-Systeme von LRZ und TUM haben eigene Datenbankstrukturen. Wie konnten wir dann die Daten aus verschiedenen Quellen in das OTRS integrieren? Wir schafften das durch die Technik XML und das Programm OTRS-Adapter. Wir haben zuerst ein XML-Schema definiert. Dieses Schema bietet ein Standardinterface an. Alle Systeme, die in OTRS integriert werden wollen, können durch dieses Interface die Daten exportieren. Beispielsweise wollen wir das OTRS in anderen Fakultäten außer der Physik erweitern, so können wir einfach die Daten aus den lokalen Quellen in diesem Format exportieren und in OTRS importieren. Die Import/Export Aufgabe nimmt das Programm OTRS-Adapter. Mit diesem Programm haben wir als Beispiel die Daten von Netzdoku und dem Physik Werkzeug exportiert und in OTRS integriert.
Mit diesen vollständigen Daten konnten wir die ITSM-Lösung von OTRS bewerten. Die Bewertungskriterien kommen aus den Anforderungen, die am Anfang dieser Arbeit in Abschnitt 3.1 eingeführt werden. Insgesammt haben wir elf Kriterien erfasst und jedes Kriterium mit einer der vier Stufen bewertet. Wir haben am Ende das Ergebnis bekommen, dass das OTRS für unsere IT-Umgebung geeignet ist. Zusätzlich haben wir noch abgeschätzt, welche Weiterentwicklung noch durchgeführt werden sollen. Die Weiterentwicklungen können im

Systemerweiterungen durchgeführt werden.

Abbildungsverzeichnis

1.1 MWN Backbone 4
1.2 Beispiel 1 aus der Netzdoku 6
1.3 Beispiel von Physik Webanwendung 7
1.4 Beispiel Anzeige Router in OTRS 9

2.1 ITIL Lebenszyklus nach [AC07a] 12
2.2 ITIL Hauptprozesse nach [AC07a] 14
2.3 CM Beziehung zu anderen Prozessen nach [AC07b] 16
2.4 Federated-CMDB Struktur nach [DC07] 20

3.1 CI Typ Deklaration 28
3.2 CI Typ Deklaration 32

4.1 CI-Typ Deklaration 35
4.2 CI-Typ Definition in OTRS 36
4.3 CI-Subtyp Definition 36
4.4 CI Beispiel 37
4.5 OTRS Suche, Erweiterung mit Wildcard-Suche 38
4.6 Netzdoku-Datenbankstruktur 46
4.7 Teil 1: Änderung an der Netzdoku-Datenbankstruktur 49
4.8 Teil 2: Änderung an der Netzdoku-Datenbankstruktur 50
4.9 Teil 3: Änderung an der Netzdoku-Datenbankstruktur 50
4.10 Teil 4: Änderung an der Netzdoku-Datenbankstruktur 51
4.11 Physik Datenbankstruktur 53
4.12 Export aus Physik-Datenbank 55
4.13 Änderung an OTRS Datenbankstruktur 55
4.14 Änderung an OTRS Datenbankstruktur 56
4.15 Änderung an der OTRS Datenbankstruktur, Tabelle configitem-definition . . 56
4.16 Änderung an der OTRS Datenbankstruktur, Tabelle configitem-version . . . 56
4.17 Änderung an der OTRS Datenbankstruktur, Tabelle xml-storage 57
4.18 Repräsentation der CIs im OTRS 60

Literaturverzeichnis

[AC07a] Alison Cartlidge, Ashley Hanna, Colin Rudd Ivor Macfarlane John Windebank Stuart Rance: *An Introductory Overview of ITIL® V3*. The UK Chapter of the itSMF, 2007.

[AC07b] Alison Cartlidge, Ashley Hanna, Colin Rudd Ivor Macfarlane John Windebank Stuart Rance: *ITIL Service Support*. The UK Chapter of the itSMF, 2007.

[AC07c] Alison Cartlidge, Ashley Hanna, Colin Rudd Ivor Macfarlane John Windebank Stuart Rance: *ITIL Version 3 Service Design*. The UK Chapter of the itSMF, 2007.

[AC07d] Alison Cartlidge, Ashley Hanna, Colin Rudd Ivor Macfarlane John Windebank Stuart Rance: *ITIL Version 3 Service Operation*. The UK Chapter of the itSMF, 2007.

[AC07e] Alison Cartlidge, Ashley Hanna, Colin Rudd Ivor Macfarlane John Windebank Stuart Rance: *ITIL Version 3 Service Strategy*. The UK Chapter of the itSMF, 2007.

[AC07f] Alison Cartlidge, Ashley Hanna, Colin Rudd Ivor Macfarlane John Windebank Stuart Rance: *ITIL Version 3 Service Translation*. The UK Chapter of the itSMF, 2007.

[Bila] `http://www.lrz-muenchen.de/services/netz/mhn-ueberblick/index.html`.

[Bilb] `https://netzdok.lrz-muenchen.de:8443/Netzdoku`.

[DC07] Dale Clark, Pratul Dublish, Mark Johnson Vincent Kowalski Yannis Labrou Stefan Negritoiu William Vambenepe Marv Waschke Van Wiles Klaus Wurster: *The Federated CMDB Vision*, Januar 2007.

[Här07] Härtl, M.: *Konzeption und Realisierung der technischen Unterstützung eines zentralen IT-Service-Desk mit OTRS an der TUM*. Diplomarbeit, Ludwig-Maximilians-Universität München, 2007.

[OTR08a] OTRS: *Admin-Handbuch*. OTRS AG, 2.2 Auflage, 2008.

[OTR08b] OTRS: *Developer Manual*. OTRS AG, 2.2 Auflage, 2008.

[OTR08c] OTRS: *ITSM Grundlagen*. OTRS AG, 1.2 Auflage, 2008.