INSTITUT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Diplomarbeit

Entwicklung eines Toolkonzeptes für die Unterstützung der ITIL Service Support Prozesse

Bearbeiter: Jens Jäger
Aufgabensteller: Prof. Dr. Heinz-Gerd Hegering
Betreuer: Michael Brenner
Vitalian Danciu

INSTITUT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Diplomarbeit

Entwicklung eines Toolkonzeptes für die Unterstützung der ITIL Service Support Prozesse

Bearbeiter:	Jens Jäger
Aufgabensteller:	Prof. Dr. Heinz-Gerd Hegering
Betreuer:	Michael Brenner Vitalian Danciu

Abgabetermin: 15. Januar 2005

Hiermit versichere ich, dass ich die vorliegende Diplomarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 15. Januar 2005

. . . .
(Unterschrift des Kandidaten)

Jeder erfolgreiche IT-Dienstanbieter muß durhgehend die mit seinen Kunden vereinbarten Dienstgütenmerkmale einhalten. Damit er diese Qualität auch wettbewerbsfähigen Preisen anbieten kann, müssen alle Arbeitsabläufe auf die Wünsche seiner Kunden optimiert werden. In dieser Diplomarbeit werden aus der ITIL Service Support Beschreibung die notwendigen Prozesse für die Administration eines Rechnerraums erarbeitet. Diese Prozesse werden dann die Abläufe zur Administration des Lehrstuhlrechnerraums angewandt.

Zusätzlich erfordern die Geschäftsprozesse eine effektive Verwaltung der anfallenden Informationen, so daß eine Unterstützung durch geeignete Softwareprodukte sinnvoll ist. Aus diesem Grund evaluiert und entwickelt diese Arbeit für das gewählte Szenario eine Reihe von Programmen, welche die gestellten Anforderungen im Szenario erfüllen können. Insgesamt zeigt damit diese Arbeit die Einführung von ITIL-Prozessen, welche auch nützliche Informationen für andere Szenarien bieten kann.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Motivation

In einer Informationsgesellschaft ist die Verarbeitung von Daten mit Hilfe moderner EDV eine Grundvoraussetzung, weil nur so die große Menge an Daten effektiv verarbeitet werden können. Jedoch stellt die moderne EDV auch sehr hohe Ansprüche an ihre Wartung, weil die Komplexität der Computerindustrie in den letzten Jahrzehnten stark zugenommen hat. Die stetige Zunahme der Komplexität und die Notwendigkeit, sich schnell an neue Gegebenheiten anzupassen, erfordert daher ein professionelles Management der IT-Infrastruktur.

Ein gutes Management baut auf den Ergebnissen der aktuellen Forschung auf und berücksichtigt die langjährigen Erfahrungen aus der Praxis. Ein moderner Ansatz für das Management ist die Prozessorientierung, welche alle Arbeitsschritte auf den Nutzen für den Kunden einer Unternehmung ausrichtet. Dieses zielorientierte Arbeiten kann dadurch die Produktivität des ganzen Unternehmens erhöhen, weil bei allen Tätigkeiten der Nutzen für den Kunden im Vordergrund steht.

Jedoch benötigt ein gutes Management auch viel Erfahrung, damit ihre Entscheidungen nicht nur theoretisch fundiert, sondern auch praxistauglich sind. Für das Management von IT-Systemen beschreiben die ITIL-Bücher Erfahrungswerte für das Management der IT-Infrastruktur. Die ITIL-Empfehlungen streben dabei eine allgemeine Beschreibung des IT-Managements an, damit so viele wie möglich die ausgesprochenen Empfehlungen auch Umsetzen können. Deshalb sind die Empfehlungen in ITIL sehr allgemein, so daß für die praktische Umsetzung nicht alle notwendigen Details beschrieben werden können.

In dieser Arbeit soll deshalb die Einführung von ITIL-Prozessen an einem überschaubaren Szenario vorgeführt werden. Das gewählte Szenario ist einerseits überschaubar, so daß der Schwerpunkt dieser Arbeit auf die ITIL-Prozesse möglich ist, und andererseits besitzt es auch die notwendige Komplexität, damit Probleme von größeren Organisationen auftauchen.

1.2 Aufgabenstellung

Im Speziellen werden deshalb in dieser Arbeit die ITIL Service Support Prozesse behandelt. Diese Prozesse sind für jedes Unternehmen notwendig, welche IT Dienstleistungen mit gleichbleibender hoher Qualität anbieten möchte. Dieses Ziel verfolgen aber viele Unternehmen und Organisationen, sobald sie ihre IT-Infrastruktur selber verwalten und diese Aufgabe nicht an Dritte ausgelagert haben. Und für die IT-Dienstanbieter stellt der Service Support sogar eine Kernkompetenz dar.

Jedoch erfordert die Einführung von ITIL-konformen Prozessen sehr viel Arbeit, weil die ITIL-Beschreibung keine spezifischen Prozesse vorgibt. Diese Arbeit soll deshalb die Lücke schließen, indem es die notwendigen Arbeitsschritte für die Einführung der Prozesse beschreibt und das Ergebniss an einem Sszenario anwendet.

Das gewählte Szenario stammt aus dem Lehrstuhl von Prof. Hegering, welches mehrere Praktika für die Informatikstudenten anbietet. Zwei Praktika werden wegen den begrenzten Resourcen innerhalb eines Rechnerraums durchgeführt. Die Administration dieses einen Rechnerraums bildet somit das ausgewählte Szenario für diese Arbeit. Die Zusammenlegung zweier Praktika in einer Rechnerraum erhöht offensichtlich die Komplexität der Administration und damit ergeben sich auch die Problemstellungen von größeren Unternehmenen, welche auch tägliche Änderungen an ihrer IT-Infrastruktur bewältigen müssen.

Für dieses Szenario sollen die ITIL-Service-Support-Prozesse erarbeitet und vollständig beschrieben werden. Dabei sind die spezifischen Eigenschaften des Szenarios zu berücksichtigen. Der Einsatz der Prozesse soll die Administration des Rechnerraums effektiver gestalten, so daß die verantwortlichen Administratoren effizienter arbeiten können.

Zusätzlich sollen die erarbeiteten Service Support Prozesse durch eine Software unterstützt werden, so daß die notwendigen und anfallenden Informationen für die Prozesse effektiv verarbeitet werden können. Der Charakter der Software ist daher eher ein Werkzeug, welche die Prozesse unterstützt, aber nicht selbstständig durchführt.

Aufgabe dieser Arbeit ist deshalb zu den Service Support Prozessen zusätzlich die Entwicklung eines Softwarekonzeptes, welche die beschriebenen Prozesse unterstützen kann. Die Einsatzfähigkeit dieses Konzeptes soll dann durch eine prototypische Teilimplementierung des Konzeptes demonstriert werden.

1.3 Vorgehensweise

Die Vorgehensweise dieser Arbeit beginnt mit einer allgemeinen Beschreibung von ITIL und endet mit spezifischen Vorschlägen für das Szenario. Damit konkretisieren sich in jedem Schritt die Geschäftsprozesse. Und für einen Überblick werden jetzt die einzelnen Kapitel kurz vorgestellt.

Beginnend mit dem zweiten Kapitel wird ein Überblick über die ITIL-Bücher mit dem Schwerpunkt auf die Support Prozesse gegegeben. Dabei wird auch der jeweilige Nutzen beim Einsatz der einzelnen Prozesse beschrieben.

Darauf folgt eine ausführliche Beschreibung des Szenarios im dritten Kapitel. Hier werden auch die Herausforderungen und die aktuellen Probleme der Administration beschrieben, welche durch den Einsatz der ITIL-Service-Support-Prozesse behoben werden sollen.

Im vierten Kapitel werden dann die spezifischen Prozesse für die Administration aus den ITIL Beschreibungen erarbeitet.

Dann folgt im fünften Kapitel die Anforderungen für die Software, welche die Prozesse, die im vierten Kapitel beschrieben sind, unterstützen sollen. Zusätzlich berücksichtigen die Anforderungen auch die spezifischen Eigenschaften des Szenarios.

Da es auch schon vorhandene Softwareprodukte für den Service Support auf dem Markt exisitieren, soll im Kapitel sechs diese kurz evaluiert werden, welche für den Einsatz am Lehrstuhl sinnvoll sind.

Und weil nicht alle Anforderungen durch vorhandene Programme erfüllt werden wird im siebten Kapitel die Entwicklung eines Prototyp beschrieben.

Und abschliessend wird im Kapitel acht eine kurze Zusammenfassung der Arbeit dargestellt und ein möglicher Ausblick gegeben.

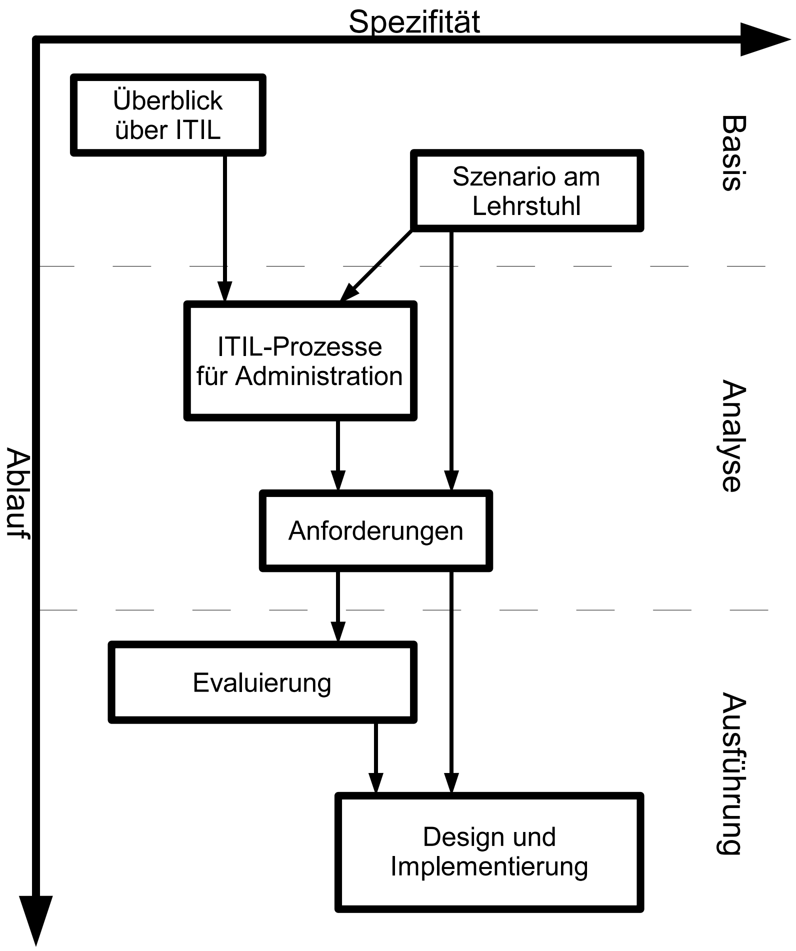

Abbildung 1.1: Vorgehensweise dieser Arbeit

2 Überblick über ITIL

Das zentrale Thema in dieser Arbeit ist der ITIL Standard und somit ist eine kurze Einführung in dieses Werk notwendig. Dazu soll in diesem Kapitel die Gründe für die Entwicklung und eine Übersicht über ITIL gegebenen werden. Der Schwerpunkt liegt aber auf der Beschreibung der Service-Support-Prozesse.

2.1 Einführung

Bevor die Einzelheiten von ITIL beschrieben werden, soll erstmal die Motivation und die Geschichte von ITIL aufgezeigt werden, um die Bedeutung von ITIL im IT Bereich besitzt. Dies ermöglicht dann später auch für ein besseres Verständnis, weil die Zusammenhänge schon erklärt worden sind.

2.1.1 Motivation

Der rasanten Entwicklung der Computerindustrie in den letzten Jahrzehnten begleitet einen ebenso schnelles Wachsen der Komplexität dieser Systeme. Jedoch muß für den produktiven Einsatz der IT-Systeme die Komplexität unter Kontrolle gebracht werden. Dies erfordert eine enge Zusammenarbeit vieler Spezialisten, welche unterschiedliche Aufgaben erledigen. Die Zusammenarbeit kann mit den Einsatz von modernen Managementmethoden gefördert und effizienter gestaltet werden. Ein weiterer Faktor für den Einsatz eines modernen Managements ist auch die wachsende Abhängigkeit der Unternehmen von ihrer IT-Infrastruktur, weil die EDV heute in viele geschäftskritische Bereiche Einzug gefunden hat. Daraus folgt, daß schon ein Teilausfall der IT-Infrastruktur sehr hohe Kosten für das Unternehmen erzeugen kann. Aus diesem Grund ist die ständige Kontrolle und ein schnelles Reagieren bei Problemen für das IT-Management sehr wichtig. Das IT-Management erfüllt dabei die Kriterien einer Dienstleistung, welche entweder vom Unternehmen selber oder von externen Dienstleistern angeboten werden muß. Um die Qualität dieser Dienstleistung (engl. service) für das Unternehmen kontrollierbar zu halten, werden diese häufig auch formal in Verträgen über Quality of Servce (kurz: QoS) Merkmale spezifiziert.

Jedoch reicht die formale Definition der Qualität nicht aus, sondern es müssen auch die Arbeitsabläufe an ein konkretes Ziel ausgerichtet werden. Ein möglicher Ansatz für diese Ausrichtung ist die Prozessorientierung der Arbeitsabläufe, welche später im Detail beschrieben wird.

2.1.2 Entstehung von ITIL

Diese Herausforderung der Unternehmen an die IT waren bekannt und sind auch der Grund, daß die “Central Computer and Telecommunications Agency” (CCTA) der britischen Regierung dieses Thema genauer untersucht hat. Das Ergebniss dieser Arbeit ist dann eine Reihe von Büchern, welche die “Best Practices” im IT-Bereich zusammenfassen. Diese Bücher sind dann unter dem gemeinsamen Namen “ITIL” für “IT Infrastructur Library” veröffentlicht worden. Später ist diese Organisation in das “Office of Government Commerce” (OGC) übergegangen. Bis heute werden die Vorschläge überarbeitet und erweitert, so daß sie auch neue Erkenntnisse und auch auf neue Entwicklungen reagieren können.

2.1.3 ITIL-Bücher

Die umfangreichen ITIL-Empfehlungen sind auf mehrere Bücher aufgeteilt. Für eine Übersicht aller beschriebenen Themen soll hier ein erster Wegweiser für ITIL vorgestellt werden. Jeder dieser Bücher beschreibt dabei ein Themenkomplex des IT-Managements, welches aber immer auch Bezüge auf alle anderen Bücher besitzt.

In der Ausgabe aus dem Jahr 2000 sind die Bücher nach folgenden Themen geordnet (vgl. [ITIL 00]):

- Deliver IT-Service
- Support IT-Service
- Manage the Infrastructur
- The Business Perspective
- Managing Applications

Wie man in Abbildung 2.1 erkennen kann, ist ITIL in 5 Bereiche aufgeteilt. Dabei soll in der Grafik mit der Überlappung und der engen Anordnung der Felder angedeutet werden, daß alle Bücher nicht für sich alleine stehen, sondern immer der Zusammenhang mit den anderen für ein umfassendes Verständnis notwendig ist. So ist keine klare Trennung zwischen den Bereichen möglich und die enthaltenen Überschneidungen sind wegen der Unschärfe der Themen notwendigerweise vorhanden. In dieser Arbeit wird aber nur der Teil zur Unterstützung von IT-Dienstleistungen genauer untersucht, weil diese einen direkten Einfluß auf die Verwaltung von Rechnern besitzen.

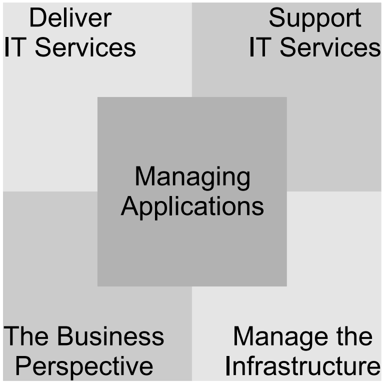

Abbildung 2.1: ITIL Aufbau

2.2 Grundlagen

Für ein besseres Verständnis der ITIL-Beschreibungen ist eine kurze Einführung in die Grundlagen der Geschäftsprozesse sinnvoll. Dabei soll hier nur das grundlegende Prinzip erörtert werden.

2.2.1 ITIL-Prozesse

Da ITIL das Management von IT-Systemen in Prozessen beschreibt, ist es notwendig eine einheitliche Definition eines Prozesses zu formulieren. Die Definition, die ITIL selber benutzt, soll hier kurz vorgestellt werden. Diese findet sich auch in der Beschreibung von ITIL-Service-Support-Prozesse[ITIL 00] oder im Buch von itSMF [BKP 02].

Ein Prozess wird in ITIL (entnommen aus [BKP 02]) wie folgt definiert:

Definition 1 (Prozess) *Ein Prozess ist eine logisch zusammenhängende Reihe von Aktivitäten zur Erreichung eines vorab definierten Ziels.*

Das Ziel eines Prozesses beschreibt die gewünschte Ausgabe (engl. Output) bei einer gegebenen Eingabe (engl. Input). Dabei unterliegen Input und Output vorbestimmten Qualitätsanforderungen oder Normen. Diese Umwandlung des Inputs in den gewünschten Output erfolgt durch die Ausführung der Prozessaktivitäten. Wenn alle Qualitätsanforderungen eines Prozesses erfüllt werden, spricht man von einem effektiven Prozess. Und wenn dies noch mit minimalen Kosten und minimalen Aufwand erfüllt wird, dann ist der Prozess auch noch effizient.

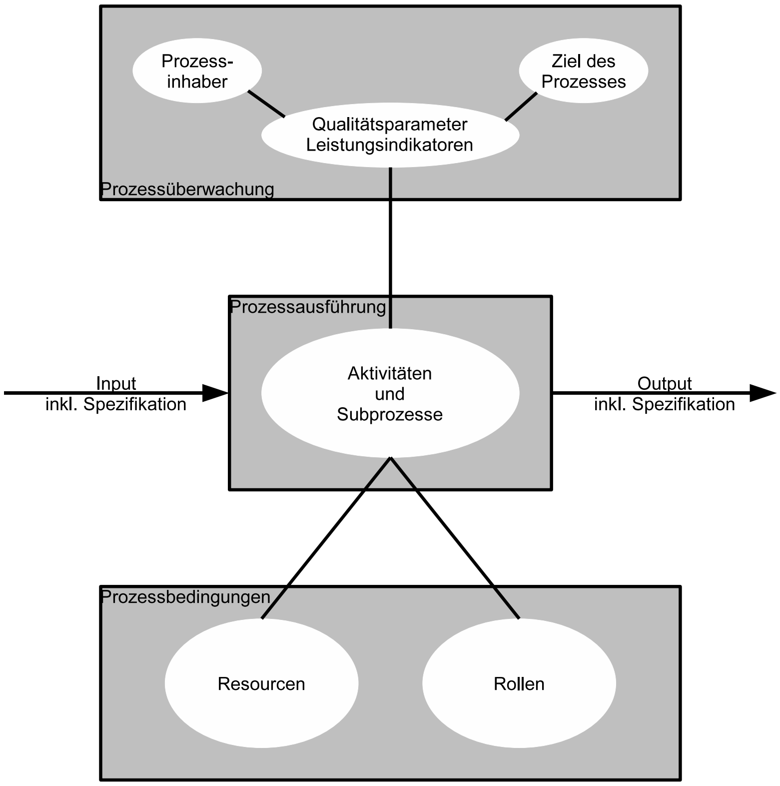

Abbildung 2.2: generisches ITIL Prozessmodell (Quelle: OGC)

Aufgaben innerhalb eines Prozesses Für die Einrichtung und Kontrolle besitzt jeder Prozess einen Inhaber und einen Manager. Die Einhaltung der geforderten Qualitätsanforderungen verantwortet dabei der Prozessinhaber (engl. Process Owner). Dazu kontrolliert er den Input und den Output seines Prozess mit seinen Vorgaben (vgl. Abbildung 2.3 auf der nächsten Seite). Zusätzlich organisiert er die Zusammenarbeit mit den anderen Prozessen, um die reibungslose Zusammenarbeit zwischen den Prozessen zu gewährleisten. Für die Beurteilung eines Prozesses benötigt aus diesem Grund der Prozessinhaber verschiedene Leistungsindikatoren, welche den Input und den Output bewerten. Der Vergleich dieser Indikatoren mit den Qualitätsanforderungen ermöglicht dem Prozessinhaber das Überprüfen und das Planen von Änderungen des Prozesses.

Hingegen verantwortet der Prozessverantwortliche (engl. Process Manager) das Einrichten und Überprüfen der Prozessaktiviäten (vgl. Abbildung 2.4 auf Seite 13). Dazu gehört auch das regelmäßige Überprüfen, ob die gewählten Strukturen innerhalb des Prozesses noch ein effektives Arbeiten ermöglichen.

Die eigentliche Durchführung der Aktivitäten eines Prozesses unterliegen den Prozessausführenden, die wiederum dem Prozessmanager regelmäßig berichten.

Leistungsmessung Damit also das Ziel eines effektiven und effizienten Prozesses auch realisiert werden können, benötigt der Prozessmanager eine Möglichkeit zur Kontrolle und Bewertung seines Prozesses. Eine bekannte Methode für Organisationen ist der Einsatz einer “Balanced Score Card” (kurz: BSC), welche auch für Einsatz der ITIL Prozesse empfohlen werden (siehe [BKP 02]). In diesem Konzept werden für die Prozesse kritische Erfolgsfaktoren (engl. Critical Success Factor, kurz: CSF) definiert, welche dann

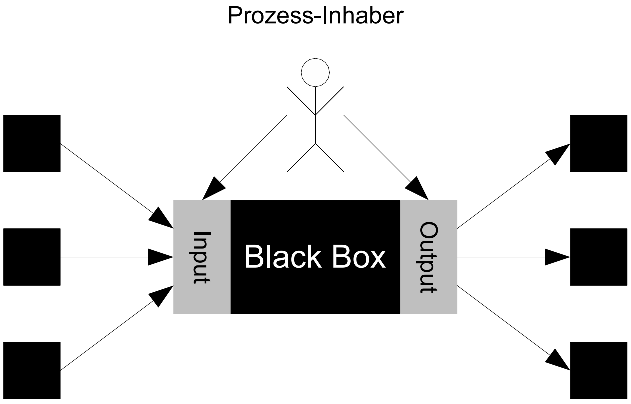

Abbildung 2.3: Aufgaben des Prozess-Inhabers

über regelmäßige Messungungen der Quantitativer Leistungsindiaktoren überprüft werden können.

Ein Quantitativer Leistungsindikator wird dabei wie folgt definiert (entnommen aus [BKP 02]):

Definition 2 (Quantitative Leistungsindikator) *Quantitative Leistungsindikator oder KPI (Key Performance Indicator) sind Variablen, anhand derer man den Fortschritt hinsichtlich wichtiger Zielsetzungen oder kritischer Erfolgsfaktoren innerhalb einer Organisation ermitteln kann.*

Prozessüberwachung Da aber kein Prozess auf Anhieb optimal arbeitet, müssen ständig die Leistungen und Ziele überprüft werden, ob diese auch der aktuelle Situation angepasst sind. Dazu benötigt der Prozessinhaber und der Manager aussagekräftige Berichte über die Leistungen über einen gewissen Zeitraum. Die Leistungen können in den Berichten über die Leistungsindikatoren gemessen und verwertet werden (siehe die Definition 2). Anhand dieser Berichte können sie dann Entscheidungen treffen, ob und welche Änderungen an den Prozessen durchgeführt werden müssen.

2.2.2 Service Support

Da diese Arbeit sich nur auf den Service Support beschränkt, soll hier einen kurzen Überblick vorgestellt werden. Die Beschreibung soll die Aufgaben und Zusammenhänge des Service Supports aufzeigen, damit später die detailierte Darstellung verständlicher ist.

Ziele Mit dem Einsatz der modernen IT ist auch die Nutzung der passenden Dienstleistungen erforderlich, damit die gesteckten Ziele vollständig erreicht werden können. Diese Dienstleistungen können dann entweder durch externe Anbieter oder selber erbracht werden. Für eine gleichbleibende hohe Qualität der Dienstleistungen, müssen die Merkmale der Diesntleistungen genau spezifiziert und regelmäßig überprüft werden.

Der Dienstanbieter braucht auf der anderen Seite geeignete Methoden, seine vereinbarten Leistungen sicher einzuhalten. Und genau diesen Zweck verfolgt ITIL bei der Beschreibung der Service Support Prozesse. Jedoch zeigt sich auch, daß die beschriebenen Dienstleistungen für alle Personen, welche eine moderne IT-Infrastruktur nutzen, sinnvoll ist. Die IT benötigt für gleichbleibenden Nutzen auch regelmäßige Wartung und Pflege, so daß viele Organisationen diese Dienstleistungen entweder selber übernehmen oder den Auftrag an Dritte übergeben. Daraus folgt, daß sehr viele die Leistungen des IT Service Support benötigen und nutzen.

Die Erfüllung der vereinbarten Leistungen ist der wichtigste Grund für die Einführung von Service Support, aber nicht der einzige. Ein weiterer Grund ist unternehmerische Notwendigkeit, sich ständig und sehr schnell an neue Situationen anpassen müssen. Jedoch erfordern regelmäßige Anpassungen an neue Gegebenheiten auch eine Weiterentwicklung der IT-Infrastruktur. Aber auch diese Herausforderung können mit dem Einsatz der Service Support Prozesse bewältigt werden.

Einordnung Für ein so komplexes Themengebiet wie den Service Support ist häufig eine einzige Kategorisierung nicht ausreichend, weil es häufig nicht nur eine Kategorisierung, sondern mehrere mögliche existieren. Deshalb soll anhand einer weiteren Grafik eine weitere Kategorisierung dargestellt werden, in der die engen Zusammenhänge zwischen den Prozessen verdeutlicht werden.

In der Abbildung 2.5 auf Seite 13 sind die für diese Arbeit wichtigen ITIL-Service-Support-Prozesse grau hinterlegt. Dieses Diagramm ist vom “British Standards Institute” in einer Publikation “A Code of Practice for IT Service Management (PD0005)” veröffentlicht worden. Dabei baut dieses Diagramm auf den ITIL-Büchern auf und wird deshalb in der aktuellen Ausgabe (vgl. [ITIL 00]) auch als Beispiel für die Verwendung von ITIL aufgeführt.

Der Grund für die Nennung dieser Grafik ist die sinnvolle Einordnung der einzelnen Service-Support-Prozesse in eigenen Kategorien. Diese Zusammenfassung der einzelnen Prozesse erfolgt aus den teilweise sehr engen Beziehungen oder aus den großen Gemeinsamkeiten im Ablauf und ihrer Ziele. Falls eine weitere Gliederung der Service-Support-Prozesse notwendig ist, so kann diese Grafik ein wertvoller Ansatzpunkt für die Organisation der einzelnen Prozesse sein.

2.3 Service Support Prozesse

Bis jetzt ist der Service Support sehr oberflächlich beschrieben worden und deshalb soll in diesem Abschnitt die einzelnen Prozesse vorgestellt werden. Die genauen Details über den Ablauf der Prozesse werden dann später im Kapitel 4 auf Seite 32 beschrieben.

2.3.1 Service Desk

Der Service Desk ist die zentrale Anlaufstelle für alle Probleme, Aufträge, Wünsche und Änderungen für die Kunden. Die Aufgabe für den Service Desk besteht dann auf die Aufnahme des Gesprächsinhaltes und die Weiterleitung an die korrekte Abteilung für eine weitere Bearbeitung. Zusätzlich soll auch der Service Desk die weitere Bearbeitung mit dem Kunden koordinieren, so daß ein fester Ansprechpartner für Fragen erhalten bleibt.

Jedoch ist dieser Prozess für diese Arbeit nicht relevant, weil das gewählte Szenario keinen eigenen Service Desk benötigt. Deshalb werden die für das Szenario sinnvollen Aufgaben des Service Desks innerhalb des Incident Management integriert. Daraus folgt auch, daß später der Service Desk nicht weiter erläutert wird.

2.3.2 Incident Management

Das Incident Management (deut. Störungsmanagement) sammelt und verwaltet alle Störungsmeldungen. Das Ziel des Incident Management ist eine schnelle Behebung aller Störungen, damit die Kunden oder die Mitarbeiter wieder ihre eigentliche Arbeit nachkommen können. Störungen können deshalb von allen Beteilgten der IT-Systeme gemeldet werden. Im Allgemeinen gibt es zwei Wege: Entweder ein Betroffener gibt die Störungsmeldung selber in das Incident Management ein, wenn dieser Betroffene dazu die Berechtigung und das Wissen besitzt. Oder man geht den Weg über den Service Desk. Dies ist in großen Organisationen der einzig sinvolle Weg, weil nur so eine konsistente und korrekte Erfassung von Störungsmeldungen möglich ist.

2.3.3 Problem Management

Das Ziel des Problem Managements ist die dauerhafte Lösung von Problemen. Im diesem Zusammenhang beschreibt ein Problem allgemein eine unerwünschte Situation [1]. Da das Ziel des Incident Management die schnelle Beseitigung der Störung ist, kann diese meistens nicht die Ursachen für eine Störungen erforschen. Im Gegensatz zum Incident Management kann sich das Problem Management die notwendige Zeit nehmen, so daß die Probleme grundlegend behoben werden können. Dies zeigt sehr deutlich die Unterschiede zwischen dem Incident und Problem Management.

Nachdem die Ursache zu einem Problem bekannt sind, wird in ITIL nicht mehr von einem Problem, sondern von einem bekannten Fehler gesprochen. Die Lösungs eines bekannten Fehlers ist die Ursache des Problems zu verhindern bzw. zu beheben. Die dabei notwendigen Änderungen werden in Form eines "Request for Change" beschrieben, welche im Change Management verwaltet werden.

2.3.4 Configuration Management

Das Configuration Management stellt Informationen über die IT-Infrastruktur den anderen Prozessen zur Verfügung. Dazu wird jedes Betriebsmittel in der IT-Infrastruktur als Configuration Items (kurz CI) erfaßt, verwaltet und kontrolliert. Weil der Begriff des Configuration Items eine zentrale Bedeutung für das Configuration Management besitzt, soll er nun formal definiert[2] werden:

Definition 3 (Configuration Item) *Configuration Items sind die Betriebsmittel für den IT-Services-Anbieter, welche für den Einsatz zugelassen sind. Über jedes Configuration Item muß seine Existenz und seinen aktuellen Zustand kontrolliert werden.*

Folgende Betriebsmittel fallen unter den Begriff des Configuration Items:

- *Computer, Hardwarekomponenten, andere IT-Geräte*
- *Software*
- *Dokumentation, Arbeitsanweisungen, Verträge*

Alle Informationen der Configuration Items werden dann in der Configuration Management Database (kurz: CMDB) abgespeichert. Die CMDB ist aber keine Inventur der vorhandenen Betriebsmittel, sondern der Schwerpunkt bei der CMDB liegt auf die Dokumentation der Konfiguration und der Abhängigkeiten zwischen den CIs. Als gute Ausgangsbasis kann aber eine Inventurdatenbank dienen, welche mit den Aspekten der Konfiguration und der Abhängigkeiten erweitert werden.

2.3.5 Change Management

Da die IT-Infrastruktur in einem Unternehmen ständigen Änderungen unterworfen ist, sollen diese Änderungen für eine bessere Kontrolle einem festgelegtem Prozess unterworfen werden. Dieser Prozess wird in ITIL im Change Management durchgeführt. Zusätzlich muß jede Änderung an der IT-Infrastruktur auch in der CMDB nachgezogen werden, damit die CMDB immer den aktuellen Stand wiederspiegelt. Änderungswünsche werden in ITIL als Request for Change (kurz RfC) bezeichnet und sind damit die Grundbausteine des Change Managementes.

Um ein reibungsloses Arbeiten nach der Implementierung einer RfC zu gewährleisten, muß jede RfC getestet und überprüft werden. Die Überprüfung der verschiedenen Aspekte einer Änderung erfolgt durch

[1] Diese Definition stammt aus dem Buch [BKP 02]

[2] Diese Definition basiert auf der Beschreibung des Configuration Managements in den Büchern [BKP 02] und [ITIL 00].

festgelegte Genehmigungen von den Experten. Diese Experten werden im Change Management zum Änderungsbeirat (engl. Change Advisory Board, kurz: CAB) zusammengefasst, welche dann jede Änderung genehmigen müssen. Aus diesem Grund ist der Änderungsbeirat mit Repräsentanten aus den verschiedenen Fachbereichen zusammengesetzt.

Falls eine Änderung doch zu neuen Problemen führt, kann das Incident oder Problem Management mit der genauen Dokumentation der durchgeführten Änderungen eine schnellere und korrekte Diagnose erarbeiten. Da es unterschiedliche, von trivialen bis zu sehr komplexen, Änderungen existieren, kann auch der Umfang und das Ausmaß der einzelnen Aktivitäten unterschiedlich stark ausfallen. Auch es soll auch möglich sein, einfache, aber dringend erforderliche Änderungen schnell und sicher umzusetzen.

2.3.6 Release Management

Damit die Administration vieler Rechner effizient durchgeführt werden kann, ist es notwendig, so weit wie möglich eine Vereinheitlichung der Rechner oder allgemein der IT-Infrastruktr. Diese Vereinheitlichung erfolgt mit einer Festlegung, welche Software und welche Hardware in einem Unternehmen zugelassen und unterstützt werden. Die Festlegung für die Software nennt man in ITIL die "Definitive Software Library" (kurz: DSL) und für die Hardware ist es der "Definitive Hardware Store" (kurz: DHS) .

Zusätzlich zu der Festlegung hat die DSL auch die Aufgabe als zentrale Verteilungsstelle der Software im Unternehmen. Analog ist der DHS das Zentrallager für die Hardware im Unternehmen.

Da diese Festlegung auch über die Zeit an die aktuelle Situation angepasst werden muß, existiert für die Anpassung des DSL und DHS der Prozess des Release Managements. Ein Release hat also die Aufgabe, eine Veränderung an der DSL oder DHS durchzuführen. Da die Auswirkungen einer solchen Änderung erheblich sind, weil diese Änderung potentiell das ganze Unternehmen betrifft, sind besonders hohe Qualitätsansprüche an ein Release zu stellen. Deshalb ist der Prozess des Release Managements strenger und ausführlicher als des Change Managements, weil im Change Management die Änderungen lokal begrenzt bleiben.

2.4 Kritik

Beim Studium der Service Support Prozesseschreibung in dem ITIL Buch erkennt man Inkonsistenzen. Diese Inkonsistenzen werden durch die Einschränkung, daß ITIL nur "Best Practices" dokumentiert, erklärt, aber genau diese Inkonsistenzen erfordern vor dem Einsatz von ITIL Prozessen eine weitere Analyse, welche die Strukturen für die Einsatzumgebung klar definiert.

2.4.1 Configuration Management

Durch den Einsatz des Configuration Management soll die Produktivität gesteigert werden, weil das Configuration Management zentral alle Informationen über die IT-Infrastruktur verwalten soll. Diese Aussage wird mit der Zielbeschreibung bestätigt (zitiert aus [ITIL 00]):

> provide accurate information on configuration and their documentation to support all the other Service Management processes

Jedoch ist es zweifelhaft, daß durch eine zentrale Informationsverwaltung die Effizienz erhöht werden kann.

Gründe für die Einführung Die Gründe für die Einführung eines Configuration Management ist folgende Überlegung: Viele Service Support Prozesse benötigen die selben Informationen über die IT-Infrastruktur. Weil ohne ein Configuration Management niemand diese Informationen anbieten kann, müssen alle Prozesse ihre notwendigen Information selber beschaffen und pflegen. Jedoch benötigen viele Prozesse die selben Informationen, so daß eine mehrfache Beschaffung und Pflege von mehreren Prozessen sehr teuer ist.

Dieser Grund spricht für eine Zentralisierung der Informationsbeschaffung aller Configuration Items, so daß alle Prozesse die benötigten Informationen nicht mehr selber verwalten müssen, sondern einfach beim Configuration Management erfragen können. Damit können die Kosten enorm gesenkt werden. Zusätzlich können die anfallenden Kosten für das Configuration Management auf alle Prozesse gleichmäßig verteilt werden, so daß die Einführung eines neuen Prozesses von den anderen Prozessen finanziert wird.

Dies ist deshalb so einfach möglich, weil die Kosten für die Reproduktion und Übermittlung von Informationen heute sehr gering sind. Dies bedeutet, daß die hohen Anfangskosten auf alle Informationsverbraucher gleichmäßig verteilt werden können und damit die Produktivität enorm gesteigert werden kann. Dieses Phänomen ist auch unter dem Begriff “First-Copy-Cost” bekannt.

Problem Jedoch sind diese Ziele sehr schwer zu realisieren, weil man dazu schon im voraus wissen muß, welche Informationen häufig nachgefragt werden. Deshalb muß das Configuration Management zuerst feststellen, welche Informationen erfaßt und angeboten werden sollen (siehe dazu den Abschnitt 4.4.3 auf Seite 42). Je besser also die Prognose der zukünftigen Informationsnachfrage ist, desto effizienter kann der Prozess des Configuration Management arbeiten.

Ein weiteres Problem ist die Finanzierung des Prozesses. Wenn nämlich die Informationen kostenlos angeboten werden, dann werden die Kosten für die Informationsbeschaffung nur auf das Configuration Management umgewälzt. Dies ist für die Produktivität sehr gefährlich, weil dann jeder so viele Informationen wie möglich nachfragen wird und somit die Gesamtproduktivität gefährdet.

Deshalb muß ein effizienter Prozess für seine angebotenen Informationen auch einen Preis verlangen. Dann beteiligen sich alle Nutzer an der Finanzierung des Configuration Management. Die Beteiligung zwingt sie auch zu einem ökonomisch sinnvollen Einsatz der Informationen und nur damit kann der Produktivität gesteigert werden. Jedoch bleibt die Frage offen, wie der ideale Preis zu berechnen ist. Dieses Problem der Preisgestaltung soll nun anhand von zwei Fällen demonstriert werden.

Falls das Configuration Management die Informationen zu sehr günstigen Preisen anbietet, dann nutzen die anderen Prozesse diesen Informationsservice entsprechend intensiver. Diese intensive Nutzung kann aber das Configuration Management dazu verleiten, noch mehr Informationen anzubieten, weil sie die falsche Annahme besitzen, daß die angebotenen Informationen wirklich benötigt werden. Am Ende steigen für das Configuration Management die Kosten und der Nutzen dieser Arbeit bleibt eher fraglich. Dieser Fall zeigt dann, wie die Einführung des Configuration Management zum Gegenteil der gesteckten Ziele führen kann.

Der andere Fall wäre zu hohe Preise für die Informationen des Configuration Managements zu verlangen, weil dann die Prozesse diesen Service eher meiden und wieder zurückfallen auf die eigenständige Beschaffung der Informationen. In dieser Situation werden also schon vorhandene Informationen erneut beschafft. Dies bedeutet aber auch ein Rückschritt zu der Situation vor der Einführung des Configuration Managements.

Aus den beiden Fällen kann man folgern, daß für eine optimale Produktivitätssteigerung das Configuration Management nur kostendeckende Preise verlangen darf, weil nur dann die Produktivität sicher für die ganze Organisation gesteigert werden kann. Jedoch besitzt das Configuration Management nur unzureichende Mittel, um diesen Idealpreis zu bestimmen. Weil der Nutzen beim Einsatz einer Information nur sehr schwer bestimmbar ist, kann auch nur sehr schwer ein gerechter Preis festgelegt werden, so daß einerseits die Kosten gedeckt werden und andererseits auch die Gesamtproduktivität weiter gesteigert werden kann.

Diese Probleme bei der Produktivitätssteigerung über den Einsatz von mehr IT werden auch im sogenannten Produktivitätsparadoxon beschrieben (siehe dazu auch die Arbeit [Pill 97][3]). Und auch das Configuration Management besitzt die Eigenschaften, welche es anfällig für dieses Paradoxon macht. Das Paradoxon

[3]Die Arbeit [Pill 97] ist auch Online verfügbar: `http://www.aib.wiso.tu-muenchen.de/piller/prodpara.htm`

beschreibt die Beobachtung, daß die Unternehmen viel in moderne IT-Infrastruktur investieren, um ihre Produktivität weiter zu steigern. Aber in der Untersuchung zeigte sich für viele Unternehmen ein gegenteiliger Effekt, daß nämlich die Produktivität in dem Zeitraum gesunken ist.

2.4.2 Incident Management

Die allgemeine Prozessbeschreibung definiert als Prozessbedingungen die beiden Faktoren Resourcen und Rollen. Der Grund, weshalb der Begriff Rolle innerhalb von Prozessen verwendet wird, ist das Ziel, daß die Prozesse unabhängig von der Organisationsstruktur sein sollen. Diese Eigenschaft der Unabhängigkeit wird als großer Vorteil in ITIL erwähnt, weil somit die Einführung von ITIL in jedes Unternehmen ohne einer teuren Reorganisation möglich sein soll.

Aber diese Eigenschaft der Organisationsunabhängigkeit wird im Incident Management verletzt, weil dort der First-, Second- und Third-Level-Support eingeführt werden. Da aber alle Supportlinien die gleiche Aufgabe und somit nach der Definition auch die selbe Rolle besitzen, bedeutet die Verwendung von drei Supportlinien eine Vorgabe für die Organisationsstruktur.

Dieses Problem ist nicht schwerwiegend, weil einerseits die Organisationsstruktur sinnvoll und üblich ist und andererseits gegebenenfalls auch leicht an die eigene Organisation angepaßt werden kann. Aber es verdeutlicht eine offensichtliche Verletzung der Prozessbeschreibung in ITIL.

2.5 Zusammenfassung

Mit diesem Kapitel soll eine kurze Einführung in die ITIL-Welt vorgenommen werden. Dadurch fällt es leichter, die späteren Ausführungen im richtigen Kontext zu sehen. Auch kann der Überblick eine Orientierung über die Ziele und Aufgaben der ausgesuchten Prozesse bieten, welche in den späteren Detailbeschreibung nur erschwert möglich ist.

Im nächsten Kapitel folgt nun die Beschreibung des Szenarios, welches der zweite wichtige Grundstein für diese Arbeit ist. Mit der Einführung in ITIL und der Beschreibung des Szenarios sind dann alle notwendigen Grundlagen für die weiteren Ausführungen in dieser Arbeit gelegt.

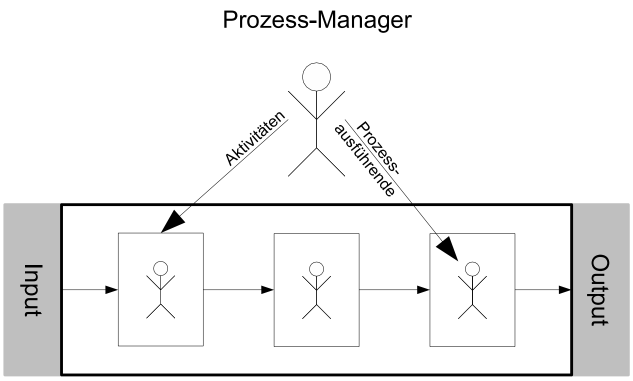

Abbildung 2.4: Aufgaben des Prozess-Managers

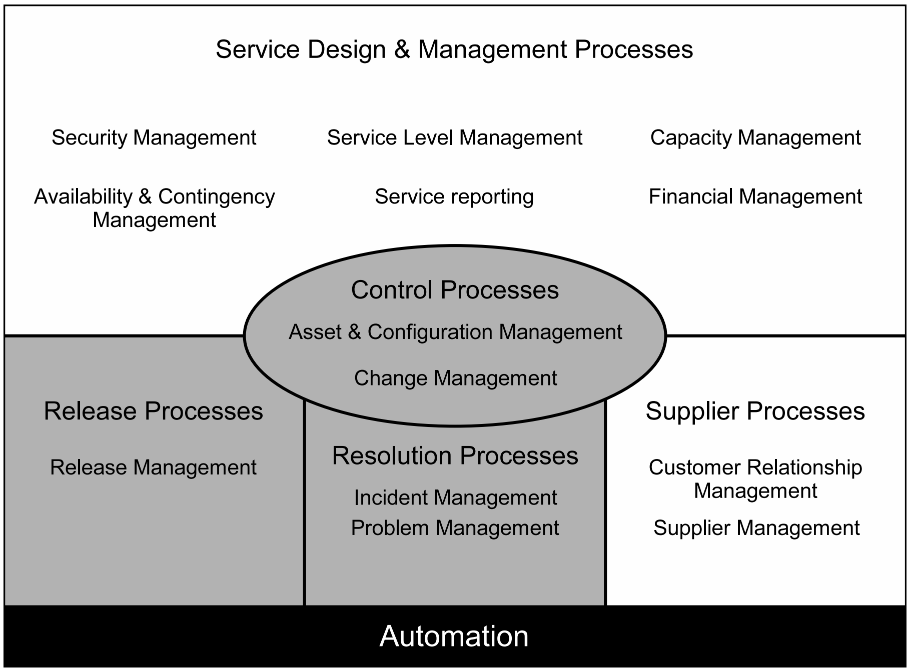

Abbildung 2.5: PD0005 Service Management

3 Szenario am Lehrstuhl

In diesem Kapitel wird ein Szenario vorgestellt, welche später als Beispiel für die Einführung von ITIL Service Support Prozessen dienen soll. Das Szenario dieser Arbeit ist im Bereich des Lehrstuhls angesiedelt, weil der Bedarf einerseits für eine Verbesserung der Administration vorhanden war und andererseits die Ergebnisse der Arbeit auch praktisch zur Anwendung kommen können.

Für einen besseren Verständnis wird zuerst ein kurzer Überblick über den Lehrstuhl gegeben, weil der Einsatzzweck des Rechnerraums ist die Durchführung von zwei Praktika. Daraus ergeben sich auch spezifische Anforderungen für die Administration, welche hier ebenfalls aufgedeckt werden sollen. Ein zentraler Punkt des Szenarios ist die detaillierte Beschreibung des Rechnerraums, weil dadurch die Notwendigkeit für die Einführung der Prozesse ersichtlich werden. Mit der Beschreibung des Rechnerraums soll auch die Komplexität vermittelt werden, welche durch die ITIL-Prozesse kontrolliert werden sollen.

Dann folgt ein Abschnitt über die Herausforderungen für die Administration, welche spezifisch für dieses Szenario sind. Und aus diesen Herausforderungen ergeben sich die aktuellen Probleme, die diese Arbeit versucht zu mindern.

3.1 Lehrstuhl

Der Lehrstuhl ist Teil der Ludwigs-Maximilians Universität München, welche von Univ.-Prof. Dr. Heinz Gerd Hegering geleitet wird. Zusätzlich zu den Aufgaben innerhalb der LMU besitzt der Lehrstuhl auch einen Lehrauftrag an der Technischen Universität München, so daß auch Studenten dieser Universität an den Veranstaltungen dieses Lehrstuhls teilnehmen dürfen. Die Schwerpunkte des Lehrstuhls liegen in folgenden Themenbereichen(Entnommen von der Lehrstuhl Homepage[1]):

- Kommunikationssysteme und neue netzbasierte Dienste
- Verteilte Systeme und Plattformen
- Internet-Anwendungen
- Planung und Betriebskonzepte für kooperative IT-Infrastruktur
- Konzepte und Methoden des IT-Managements
- Systemprogrammierung und Betriebssysteme
- Multimedia-unterstütztes E-Learning
- Telekommunikations- und Mobilkommunikationssysteme

Dabei fällt das Thema dieser Arbeit in den Schwerpunkt "Konzepte und Methoden des IT-Managements" und die Praktika sind ein Teil der Lehre über die "Kommunikationssysteme und neue netzbasierte Dienste". Und somit vereinigt diese Arbeit die beiden Themengebiete in sich.

Lehre Eine wichtige Aufgabe für jeden Lehrstuhl an einer Universität ist die Lehre. In ihr soll das aktuelle Wissen an die Studenten weitervermittelt werden. Wie in der Beschreibung des Lehrstuhls in Abschnitt 3.1 zu sehen ist, behandelt der Lehrstuhl viele Themen über Rechnernetze. Dieses Wissen über

[1]Lehrstuhl Homepage: `http://wwwhegering.informatik.tu-muenchen.de/`

das Thema Rechnernetze wird unter anderem in praktischen Versuchen im Rahmen des Rechnernetzpraktikums vermittelt. Ein wichtiger Aspekt bei Rechnernetzen ist auch die Sicherheit der Kommunikation, welches im Rahmen des IT-Sicherheitspraktikum behandelt wird. Diese beiden Praktika sind ein Teil des Szenarios, weil sie maßgeblich die Konfiguration des Rechnerraums beeinflusst haben.

Praktika Die beiden Praktika über die IT-Sicherheit und über Rechnernetze werden im Rahmen des Informatikstudiums an der Technischen Universität München (kurz: TU-München) und an der Ludwigs-Maximilians-Universität München (kurz: LMU-München) für das Hauptstudium vom Lehrstuhl des Prof. Hegerings angeboten. Die TU bietet die beiden Praktika im Bereich der "Technische Informatik" und die LMU im Bereich der "Systemnahen und Technischen Informatik" an.

Rechnernetze Auf der Homepage für das Rechnernetze Praktikum im Sommersemester 2004[2] wird das Ziel des Praktikums erklärt:

> Im Rahmen dieses Praktikums werden ausgewählte Fragestellungen aus dem Bereich der lokalen Netze und der Weitverkehrsnetze behandelt. Themen sind u.a.:
> - Codierungs- und Modulationsverfahren
> - Vielfachzugriffsverfahren und ihr Lastverhalten am Beispiel Ethernet
> - Ablauf typischer Protokolle der OSI-Schichten 2 und 3
> - Aufbau und Konfiguration eines ATM-Netzes
> - Konfiguration typischer Kommunikationskomponenten
> - Konfiguration von IP-Netzen
> - Sicherung von IP-Netzen
> - Komponentenmanagement am Beispiel von Ethernet-Switches
> - Netzmanagement am Beispiel von ATM
> - Netzmanagement-Plattformen

Damit liegen die Schwerpunkte dieses Praktikums auf die Einführung in die verschiedene Netzwerkprotokolle (IP, Ethernet und ATM) und das notwendige Management von Computernetzen.

IT Sicherheit Ein wichtiges Thema für alle Computernetze ist die sichere Kommunikation. Weil aber das Thema Sicherheit sehr komplex ist, wird dieses Thema auch in einem eigenen Praktikum vermittelt.

Auch auf der Homepage für das IT-Sicherheits Praktikum im Sommersemester 2004[3] wird das Ziel des Praktikums erklärt:

> Im Rahmen dieses Praktikums werden ausgewählte Probleme und Fragestellungen aus dem Bereich der Sicherheit von TCP/IP-basierten Kommunikationsssytemen behandelt. Die Themen und Aufgaben umfassen u.a.:
> - Grundlagen von TCP/IP Netzwerken
> - Netzwerk Dienste
> - Firewall I: Paketfilter
> - Firewall II: Application Level
> - Intrusion Detection
> - Hacking-Angriffe
> - VPN-Technik: IPSec

---

[2] Rechnernetze Praktikum im Sommersemester 2004: `http://wwwhegering.informatik.tu-muenchen.de/Praktika/ss04/rnp/`
[3] IT-Sicherheits Praktikum im Sommersemester 2004: `http://wwwhegering.informatik.tu-muenchen.de/Praktika/ss04/secp/`

Beteiligte Personen Da die Durchführung von praktischen Versuchen sehr viel Personal benötigt, sollen hier kurz die verschiedenen Aufgaben der Personen vorgestellt werden.

Den Bedarf und den Inhalt für die Praktika entscheidet natürlich Prof. Hegering. Weil die Durchführung jedes Praktikums enorme Resourcen beansprucht, ist eine Vollauslastung der Praktika und die erfolgreiche Durchführung eine Voraussetzung für das Weiterbestehen jedes Praktikums.

Für die eigentliche Durchführung und Betreuung der Teilnehmer sind die Praktikumsbetreuer zuständig. Die Betreuer erstellen dann die Vorträge und die Aufgaben für ihr Praktikum und sind während des Praktikums der Ansprechpartner für alle Belange bezüglich ihres Praktikums.

Natürlich braucht auch jedes Praktikum Teilnehmer, welche das Praktikumsangebot auch nutzen. Die Teilnahme an Praktika wird durch die jeweilige Prüfungsordnung an den Fakultäten vorgeschrieben. Als Beleg für die erfolgreiche Teilnahme an einem Praktikum erhalten die Teilnehmer einen Praktikumsschein.

Da alle angebotenen Praktika auch sehr komplexe Aufgaben für die Teilnehmer besitzen, unterstützen eine Menge von Tutoren die Betreuung der Teilnehmer. Die Tutoren sollen die erste Anlaufstelle für Fragen bei den praktischen Versuchen sein. Dadurch können die Tutoren einerseits den Teilnehmern helfen, weil die Teilnehmer eine zeitnahe Unterstützung erhalten, und andererseits entlasten sie auch die Betreuer erheblich, weil einfache Fragen schon von den Tutoren vollständig beantwortet werden können.

Die Tutoren sind auch wie die Teilnehmer Informatikstudenten, welche als wissenschaftliche Hilfskraft (kurz HiWi) am Lehrstuhl angestellt werden. Die notwendige Qualifikation der Tutoren erhalten sie meistens durch ihre erolfgreiche Teilnahme an dem Praktikum.

Für die Bereitstellung der notwendigen Computer und der dazugehörigen Infrastruktur ist auch der Lehrstuhl zuständig. Diese IT-Infrastruktur muß aber auch gewartet werden, damit die Teilnehmer sich auch auf die Aufgaben und nicht auf die Reparatur der Computer konzentrieren können. Für die Aufgabe der Wartung des Rechnerraums sind dann die Administratoren am Lehrstuhl zuständig. Dabei müssen die Administratoren mit den Betreuern sehr eng zusammenarbeiten, damit ein reibungsloser Betrieb der Praktika in den Rechnerräumen gewährleistet werden kann.

3.2 Rechnerraum

In diesem Abschnitt soll ein Überblick über die technische Ausstattung und der Konfigurationen des Rechnerraums gegebenen werden. Diese Beschreibung dient zwei Zielen, zum einen soll es die Komplexität demonstrieren, welche den Einsatz von ITIL Service Prozessen gerechtfertigt, und zum anderen soll es später die Datengrundlage für das Configuration Management System liefern.

3.2.1 Raumverteilung

Der Rechnerraum besteht streng genommen aus drei Zimmern im Keller des Informatikinstituts an der LMU, welche aber mit den offenen Türen zwischen den Zimmenern zusammenhängend erscheinen. Die beiden Abbildungen 3.2 auf Seite 21 und 3.5 auf Seite 25 zeigen die eingesetzten Computer für die beiden Praktika. Jedoch werden viele Computer für beide Praktika eingesetzt, so daß sie jeweils einen eigenen Namen für jedes Praktikum besitzen.

Aus den Namen der Rechner kann man die Rechnerarchitektur und die Verwendung herauslesen. Die ersten beiden Buchstaben deuten die Rechnerarchitektur des betreffenden Gerätes an. Dabei steht “PC” für IBM-Kompatiblen PCs, welche *Linux* oder *MS-Windows* benutzen. Der Präfix “hp” steht für die Hewlett-Packard Workstations, auf denen das mitgelieferte *HP-Unix* läuft.

Obwohl die Switches in den beiden Abbildungen für die Raumverteilung eingetragen sind, soll hier kurz das Namensschema vervollständigt werden. Ihr Name enthält entweder das Wort “switch” (vgl dazu “vlanswitch1") oder beginnen mit dem Präfix "sw". Fast alle Ethernet-Switches kommen vom Hersteller 3Com und sind managebar. Für Übungszwecke existieren im Raum-D9 noch zwei HP-Switches.

3.2.2 Einsatz von Profilen

Eine wesentliche Eigenschaft des Szenarios ist die Mehrfachverwendung der Rechner für zwei unterschiedliche Praktika. Dies bedeutet, daß jeder Computer abhängig vom Praktikum eine andere Systemkonfiguration auf der Softwarebene besitzt. Diese unterschiedlichen Konfiguration sind aber notwendig für die unterschiedlichen Aufgaben in den Praktika, weil beide Praktika erstens sehr systemnah angelegt sind und zweitens auch sehr unterschiedliche Themen behandeln.

Um diese Mehrfachverwendung der Rechner besser zu verwalten, soll hier der Begriff eines Konfigurationsprofiles eingeführt werden.

Definition 4 (Profil) *Ein Profil umfasst die komplette und von den Administratoren vorgesehene Konfiguration des Rechnerraums zu einem Zeitpunkt.*

Das Vorhandensein von zwei Profilen für die beiden Praktika ist offensichtlich, jedoch ergeben sich bei genauerer Betrachtung noch mehr Profile, weil die unterschiedlichen Aufgaben auch leicht veränderte Konfigurationen benötigen. Und diese veränderte Konfigurationen erfüllen damit die Anforderungen für ein neues Profil, so daß potentiell für jede Aufgabe aus den beiden Praktika ein eigenes Profil definiert werden kann.

Jedoch soll zur Wahrung einer besseren Übersicht der Profilbegriff nicht zu häufig eingesetzt werden. Dies bedeutet, daß eine kleine Änderung an der Konfiguration für eine spezielle Aufgabe noch kein Grund für die Erstellung eines eigenen Profils ist. Für den Anfang kann jeweiles ein Profil pro Praktikum ausreichen. Dann können später innerhalb der Praktika jeweils neue Profile angelegt werden, falls die verwendeten Konfiguration sich erheblich unterscheiden.

3.2.3 Bootkonzept

Für die Unterstützung der verschiedenen Profile sind mehrere Betriebssystem auf jedem Rechner vorhanden. Damit die Administration dieser unterschiedlichen Betriebssysteme handhabbar bleiben, werden die Betriebssystem über das vorhandene Netzwerk von einem zentralen Server geholt und dann gestartet. Dieses Konzept ist in einem Systementwicklungsprojekt (siehe [AlCh 03]) für das IT-Sicherheitspraktikum erarbeitet worden und dann später für das Rechnernetzpraktikum erweitert worden.

3.2.4 Rechnernamen

Die beiden Praktika benutzen die selben Rechner, aber mit jeweils einer anderen Konfiguration. Aus diesem Grund besitzen viele Rechner mindestens zwei unterschiedliche Namen. Zusätzlich besitzen viele Rechner auch mehrere Netzwerkkarten und jede dieser Netzwerkkarten besitzt eine eigene IP-Adresse und damit auch einen eigenen Namen. Dadurch besitzen viele Rechner sogar mehr als zwei Namen. Um diese Komplexität der Namen zu beherrschen, kann man mehrere Profile einsetzen (siehe dazu den Abschnitt 3.2.2).

Damit in der weiteren Beschreibung dieses Szenarios die Rechner auch mit unterschiedlichen Namen immer zu identifizieren, werden in der Tabelle 3.1 auf der nächsten Seite die unterschiedlichen Namen pro Rechner aufgezählt. Diese Tabelle muß zeilenweise gelesen werden und in jeder Zeile stehen alle Namen eines Rechners.

Dabei fällt der der Rechner pcrnp10 besonders auf, weil er sechs Namen besitzt. Diese Menge an Namen ist durch den Einsatz als Server für das Rechnernetzpraktikum zu erklären und deshalb benötigt er für jedes Subnetz eine eigene IP-Adresse und einen dazugehörigen Namen. Dieser besondere Rechner wird deshalb später (siehe 3.3.4 auf Seite 22) genauer untersucht.

pcrnp9	pcfwsub				
pcrnp10	pcrnp10fw	pcrnp10atm	pcrnp10nm1	pcrnp10nm2	pcrnp10boot
swatm4	swlane4	swclip4			
swatm5	swlane5	swclip5			
hprnp1	hplane1	hpclip1			
hprnp2	hplane2	hpclip2			
pcatm3	pclane3	pcclip3	pcsec1		
pcatm4	pclane4	pcclip4	pcsec2		
hprnp5	hplane5	hpclip5			
hprnp6	hplane6	hpclip6			
pcrt1sub1	pcsec3				
pcrt1sub2	pcsec4				
pcrt2sub1	pcsec5				
pcrt2sub2	pcsec6				
pcnm1prot	pcsec7				
pcnm1ov	pcsec8				
pcnm2prot	pcsec9				
pcnm2ov	pcsec10				

Tabelle 3.1: Rechnernamen und Aliase

3.2.5 Verkabelung

Da beide Praktika zum Thema Computernetzwerke gehören, ist die Vernetzung aller Praktikumsrechner offensichtlich notwendig. Dabei ist die Komplexität des Netzwerks erheblich höher als für diese kleine Anzahl von Rechnern üblich ist, weil diese Komplexität als Übungsobjekt für das Rechnernetzepraktikum benötigt wird.

Eine zentrale Stelle für die Vernetzung in diesem Rechnerraum sind die beiden VLAN-Switches von 3Com (mit dem Namen “vlanswitch1” und “vlanswitch2”), die mit weiteren Hubs, Patchfeldern und mit den beiden Servern “pcrnp10” und “secserver” zusammen in einem 19-Zoll-Rackgehäuse stehen. Dieses Gehäuse steht im Raum D-9. Dabei arbeiten die beiden VLAN-Switches zusammen, welche über ein proprietäres Kabel verbunden sind.

Für das eingesetzte Bootkonzept (siehe Abschnitt 3.2.3 auf der vorherigen Seite) im Rechnerraum besitzt jeder Rechner mindestens eine eigene Ethernet-Netzwerkkarte, über die er booten kann. Und da viele Rechner mehr als zwei Ethernet-Netzwerkkarten besitzen, ist die Ethernet-Verkabelung in diesem Rechnerraum sehr komplex.

Jedoch ist es bei einer komplexen Verkabelung immer sinnvoll, alle Kabel zu beschriften und für alle Hubs und Switches ein Port-Belegungsplan zu erstellen. Für diesen Rechnerraum waren verschiedene Port-Belegungspläne vorhanden, die mehrmals korrigiert und überarbeitet worden sind. Aus diesem Grund ist für diese Arbeit ein neuer Plan anhand der Beschriftungen der Kabel erstellt worden (siehe Abbildung 3.1 auf der nächsten Seite).

Weil viele Rechner mehrere Netzwerkkarten besitzen, reicht die Beschriftung des Kabels mit dem Namen des Rechners jedoch nicht aus. Aus diesem Grund erhält jeder Rechner für seine Netzwerkkarten Kabel mit unterschiedlichen Farben, so daß die Identfizierung der Kabel vom Switch zu einem Rechner einfacher erfolgen kann. Aus diesem Grund steht in der Abbildung die Farbe des Netzwerkkabels hinter dem Rechnernamen.

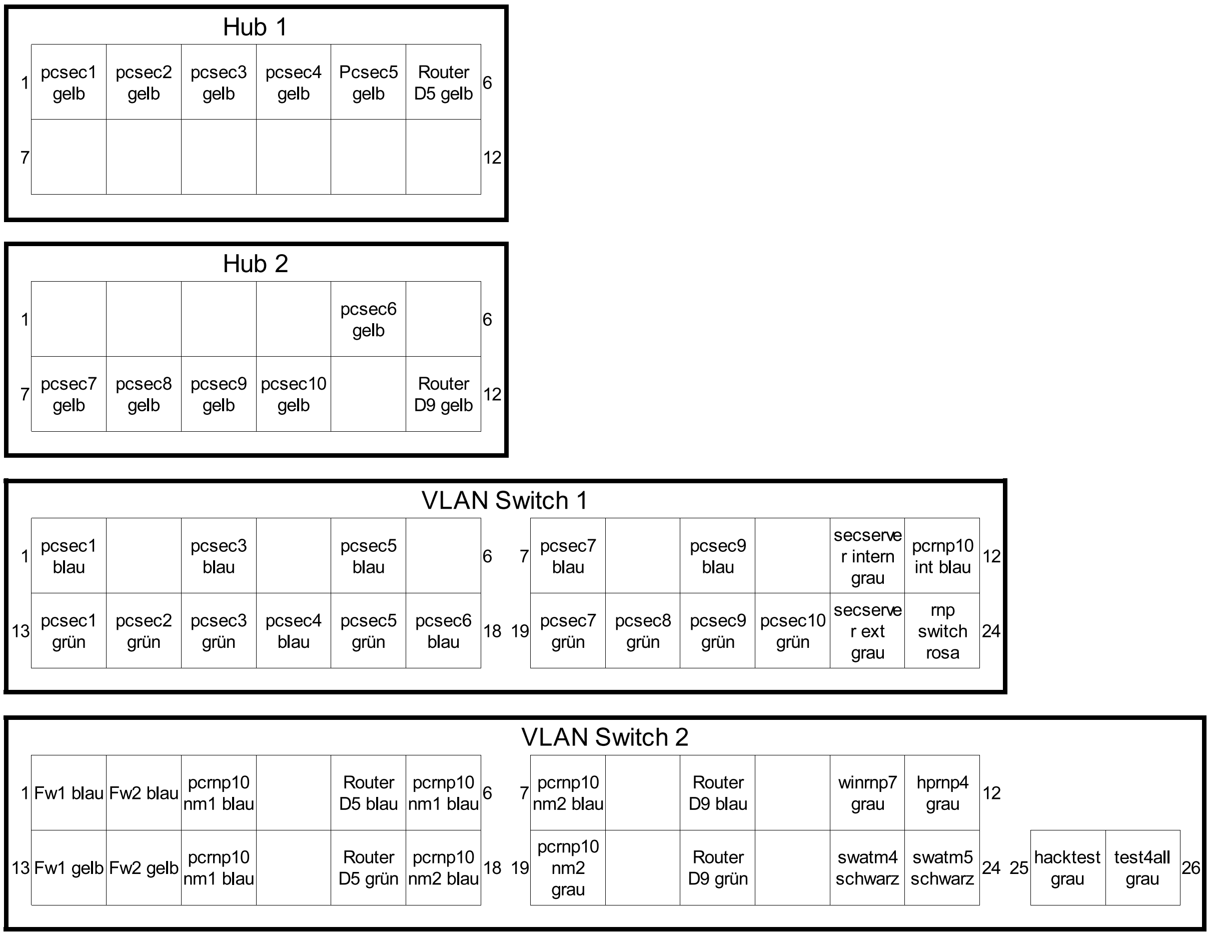

Abbildung 3.1: Port-Belegung der Switches und der Hubs

3.2.6 Netzwerkkarten

Schon mehrmals ist die Verwendung mehrerer Netzwerkkarten pro Rechner in dieser Arbeit beschrieben worden und weil diese Eigenschaft des Szenarios ein erheblicher Teil der Komplexität darstellt, soll dies genauer untersucht werden.

Um eine Netzwerkkarte unabhängig von dem Betriebssystem oder einer anderen Softwarekonfiguration eindeutig zu identifizieren, benutzt man die eindeutige Hardwareadresse, welche auch häufig als MAC-Adresse genannt wird.

Da die MAC-Adressen für den reibungslosen Ablauf eines Ethernetnetzwerkes eindeutig sein müssen, werden ganze Adressintervalle an die verschiedenen Netzwerkkartenhersteller zugewiesen. Somit kann man auch aus der MAC-Adresse auf den Hardwarehersteller der Netzwerkkarte zurückschliesen.

In der Tabelle A.3 auf Seite 109 stehen nun die MAC-Adressen bei den jeweiligen Rechnern, in denen sie eingebaut sind. Zusätzlich steht in der dritten Spalte der Hersteller der Netzwerkkarte, falls sich dieser eindeutig identifizieren lies.

Diese Tabelle A.3 auf Seite 109 ist eine sehr wichtige Grundlage, um die komplexe Netzwerkkonfiguration dieses Szenarios zu verstehen. Anhand der MAC-Adresse kann immer eindeutig der Rechner und die Netzwerkkarte unabhängig von der Softwarekonfiguration ermittelt werden. Diese Fähigkeit ist besonders in diesem Szenario wichtig, weil mit den zwei Praktika sehr unterschiedliche Netzwerkkonfigurationen bezüglich Software und Betriebssystem gefahren werden.

Um beispielsweise die Netzwerkkarte und den dazugehörigen Rechner zu einer IP-Adresse oder zu einem DNS-Namen zu identifizieren, schaut man im Betriebssystem im ARP-Cache nach, welche MAC-Adresse zu der IP-Adresse zugeordnet ist.

Aber man kann auch die Verkabelung der Rechner mit den managebaren Switches ermitteln, weil die Switches zu jedem Port eine Liste von MAC-Adressen besitzen, welche mit diesem Port physikalisch verbunden sind. Mit diesem Verfahren kann man dann über die Software die physikalische Netzwerktopologie erfahren.

3.3 Rechnernetzpraktikum

In diesem Abschnitt soll die spezifische Konfiguration für das Rechnernetzpraktikum beschrieben werden. Weil das Praktikum vorallem über Computernetzwerke handelt, liegt auch der Schwerpunkt dieser Beschreibung auf der Netzwerkkonfiguration. Und da die Rechnernetzkonfiguration sich sehr stark von der Konfiguration des IT-Sicherheits-Praktikum unterscheiden, sollte die Konfiguration für das Rechnernetzpraktikum zu einem eigenen Profil zusammengefasst werden.

3.3.1 Aufteilung der Themen

Das Rechnernetzpraktikum ist logisch in drei Themenbereiche aufgeteilt, welche unabhängig voneinander behandelt werden können. Dies ermöglicht die Aufteilung der Teilnehmer auf mehrere Gruppen mit jeweils zwei Teilnehmern. Die gleichzeitige Bearbeitung mehrerer Themenbereiche erfordert auch eine ausreichende Anzahl von Computern, damit die Gruppen ohne Probleme nebeneinander arbeiten können. Zur besseren Trennung stehen die Computer in Gruppen zusammen, welches man auch sehr gut in der Abbildung 3.2 auf der nächsten Seite erkennen kann.

IP Netzwerk In dem Abschnitt über IP Netzwerke werden die Ethernetgrundlagen mit Versuchen den Teilnehmern näher gebracht. Dazu besitzt der Rechnerraum auch die notwendigen Messgeräte, um auch die elektrischen Signale bei einer Ethernetübertragung zu beobachten.

Dann folgen verschiedene Versuche über das Internetprotokoll IP und ihrer Konfigurationsmöglichkeiten. Für diese Versuche sind auch die teilweise komplexe Netzwerktopologien notwendig, um den Einsatz der unterschiedlichen Konfigurationen auch zu demonstrieren.

Und am Ende dieses Abschnitts über die IP Netzwerke erfolgen einige Versuche zum Thema Sicherheit, in dem verschiedene Firewallkonzepte in praktischen Versuchen untersucht werden.

ATM Netzwerk Für das weniger bekannte ATM Netzwerk benötigt das Praktikum erst eine allgemeine Einführung in diese Netzwerktechnologie. Nach dieser Einführung wird dann zuerst der Einsatz des “ATM Adaption Layer” (kurz: AAL) und von “Classical IP and ARP over ATM” (kurz: CLIP) in praktischen Versuchen demonstriert.

Und am Ende wird die andere Einsatzmöglichkeit von ATM mit der “LAN Emulation” (kurz: LANE) vorgeführt.

Netzmanagement In diesem Abschnitt werden zuerst die bekannten Netzmanagementwerkzeuge vorgestellt, welche dann später in den Versuchen auch benutzt werden sollen.

Die Integration der unterschiedlichen Werkzeuge wird wegen der hohen Komplexität auf Basis einer Netzmanagement-Plattform durchgeführt. Einer dieser Plattformen wird deshalb in dem Praktikum vorgestellt. Die Teilnehmer müssen dann diese Plattform zur Lösung der gestellten Aufgaben nutzen.

Am Ende werden verschiedene Konzepte zum Management der Netzwerkkomponenten vorgestellt. Dazu gehört die Beschreibung der unterschiedlichen Komponenten. Für die praktische Versuche zum Komponentenmanagement wird ebenfalls eine gängige Software benutzt.

Folgen der Aufteilung Weil die unterschiedlichen Themenbereiche im Rechnernetzpraktikum auch eine unterschiedliche Netzinfrastruktur benötigen, besitzen diese drei Bereich auch jeweils eine eigene Rechnergruppe, welche in der Abbildung 3.2 dargestellt werden. In dieser Abbildung stehen zum einen die Rechnernamen für dieses Praktikum und zum anderen sind die Rechner entsprechend ihrer Aufgabe gruppiert. Für die Ethernetversuche ist die notwendige Hardware in den angegebenen Bereich vorhanden, aber diese Rechner besitzen keine oder nur eine eingeschränkte Netzverbindung zu den anderen Rechnern, damit die Ethernetversuche nicht das restliche Netzwerk beeinträchtigen können.

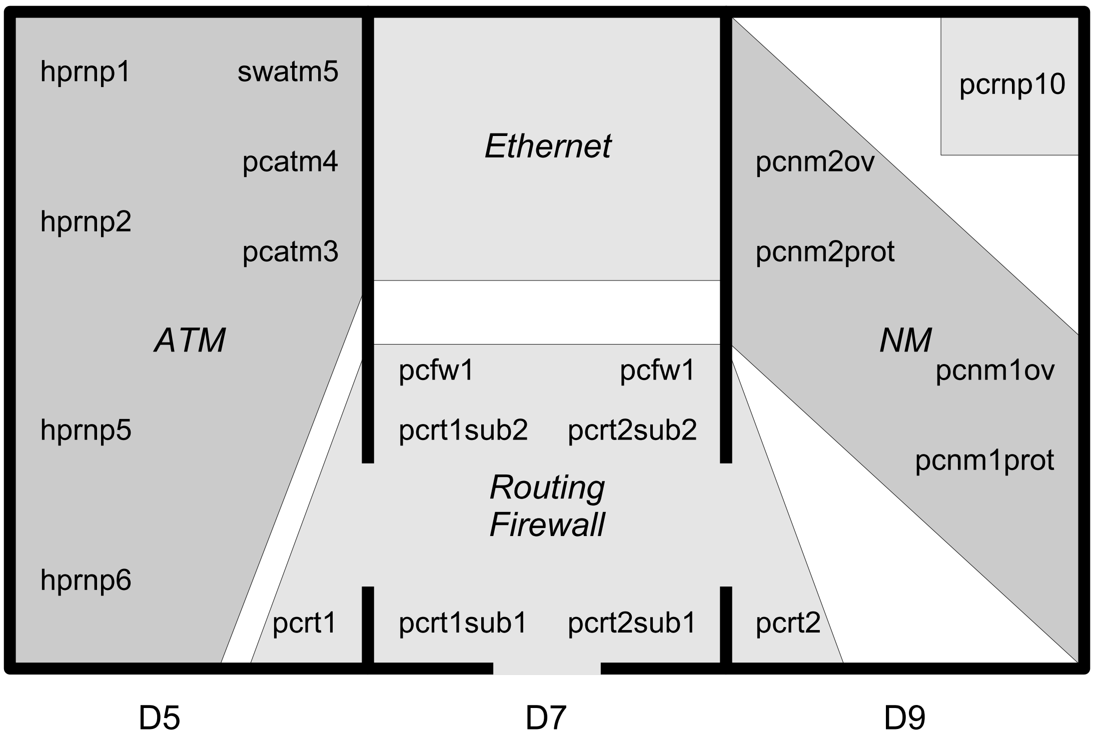

Abbildung 3.2: Raumaufteilung für das Rechnernetzepraktikum

3.3.2 Adressen zu den Namen

Da die Rechnernamen auch einen Hinweis auf die Verwendung und ihrer Aufgabe geben, sollen zuerst alle Namen für die Rechner aufgezählt werden. Aus dieser Aufzählung können dann weitere Rückschlüsse gezogen werden.

Dann wird der Zusammenhang zwischen den Namen und der verwendeten Hardware hergestellt, so daß auch die Verteilung der Namen auf die vorhandenen Rechner klar ist.

DNS Konfiguration Für einen vollständigen Überblick über alle Rechnernamen ist ein Blick auf die DNS-Server sehr hilfreich. Dabei stellt man fest, daß jedes Praktikum eine eigene DNS-Domäne besitzt. Für das Rechnernetzpraktikum zeigt die Tabelle A.2 auf Seite 108 alle eingetragenen DNS-Namen der Domäne “rnp.nm.informatik.uni-muenchen.de”. Die Daten für diese Tabelle stammen aus einem DNS-Zonetransfer (beschrieben in [PACL98]) aus dem zuständigen DNS-Server “pcrnp10”.

Dabei sind die die Rechner nach ihren Subnetzen zusammengefasst. Der erste Name in jedem Subnetz entspricht dem Namen des ganzen Subnetzes und wird deshalb durch Kapitälchen besonders hervorgehoben.

DHCP Konfiguration Mit dem Wissen über die IP-Adressen kann aber noch kein Zusammenhang mit der vorhandenen Hardware hergestellt werden, weil die IP-Adressen unabhängig von der darunterliegenden Hardwardeschicht ist. Aus diesem Grund ist der Zusammenhang zwischen IP-Adresse und der Netzwerkkarte erforderlich. Da jede Netzwerkkarte eine eindeutige Hardwareadresse für das Ethernetprotokoll benötigt, soll nun die Abbildung der MAC-Adresse auf eine IP-Adresse verdeutlicht werden.

Diese Zuordnung kann über drei unterschiedliche Verfahren geschehen: Statische Konfiguration, Konfiguration über DHCP oder manuelle Konfiguration.

In der statischen Konfiguration trägt der Administrator die IP-Adresse zu einer Netzwerkkarte in die Konfiguration des Betriebssystems ein. Somit bleibt die IP-Adresse während der Nutzung des Betriebssystems immer erhalten.

Da jedoch die statische Konfiguration für den Administrator sehr arbeitsaufwendig ist, weil er jeden Rechner einzeln konfigurieren muß, existiert die zentrale Netzwerkkonfiguration über einen DHCP-Server. Dann frägt das Betriebssystem über das Netzwerk den DHCP-Server, welche IP-Adresse es benutzen soll.

Und die dritte Möglichkeit ist die manuelle Konfiguration, in dem man dem Betriebssystem zur Laufzeit die zu verwendeten IP-Adresse mitteilt. Da diese Konfiguration nicht auf der Festplatte gespeichert wird, ist bei einem Neustart die Konfiguration verloren.

Einen kurzen Überblick über die DHCP-Konfiguration für das Rechnernetzpraktikum erhält man in der Tabelle A.4 auf Seite 110, welche die Zuordnung der MAC-Adresse zu den jeweiligen IP-Adressen abbildet. Da im Praktikum die Grundlagen über Computernetzwerke vermittelt werden sollen, werden nicht alle Netzwerkkarten über DHCP konfiguriert. Die Konfiguration der restlichen Netzwerkkarten ist dann Bestandteil der Praktikumsaufgaben, welche die Teilnehmer lösen müssen.

3.3.3 VLAN-Konfiguration

Das Rechnernetzpraktikum benötigt wie schon beschrieben auch Beispiele für komplexere Netzstrukturen. Diese komplexe Netzstrukturen werden im Praktikum durch Virtuelle Netzwerke (auch VLAN genannt) an den beiden Switches simuliert. Dabei kann die VLAN-Konfiguration am Switch automatisch per Software geändert werden. Für die automatische Änderung der VLAN-Konfiguration bieten die Administratoren Shellskripte an, die die gewünschte Konfiguration am Switch einstellt.

Anhand des Skriptes ist dann die Tabelle 3.2 auf der nächsten Seite entstanden, welche eine kurze Übersicht über die vorhandenen VLANs geben soll.

3.3.4 Konfiguration des Servers pcrnp10

Am Beispiel des Computers pcrnp10 soll die Komplixität der vorhandenen Netzkonfiguration eines Rechners demonstriert werden. Ein Grund für die Wahl des Rechners ist sein Verwendungszweck als Server, und damit bleibt seine Konfiguration über den Zeitraum eines Praktikums unverändert.

ID	Name	Switch	Portnummer
6	PCSec3zuRouterD5	1	15
		2	17
7	PCSec5zuRouterD9	1	17
		2	21
8	nm1	2	3, 6, 15
9	nm2	2	7, 18, 19
10	nm1pcsec7pcnm1prot	1	7
		2	4
11	nm2pcsec9pcnm1ov	1	20
		2	16
12	nm2pcsec9pcnm2prot	1	9
		2	8
13	nm2pcsec10pcnm2ov	1	22
		2	20
14	secpMINI	1	11
		2	25, 26

Tabelle 3.2: VLAN Konfiguration für das Rechnernetzpraktikum

Allgemeines Der Computer pcrnp10 steht im Rackgehäuse und ist über die angeschlossene Tastatur und Monitor einfach zu bedienen. Weil dieser Computer eine zentrale Rolle einnimmt, können aus seiner Konfiguration sehr viele Hinweise auf die Konfiguration der anderen Rechner gezogen werden.

Routing Ein zentraler Punkt im IP-Netzen ist das Routing, welches über die "IP routing table" gesteuert wird. Aus diesem Grund ist in der Tabelle 3.3 auf der nächsten Seite eine aufbereitete routing table des Computers pcrnp10, welche nach Netzwerkkarten zusammengefasst und nach Adressen geordnet ist.

Auf den ersten Blick erscheint es sehr viele Routingeinträge für die Netzwerkkarte eth0 zu existieren. Diese Häufung kann aber durch die nächste Abbildung 3.3 auf der nächsten Seite erklärt werden, weil dort die Erklärung für diese Eintrage stehen. Für die Netzwerkkarte eth0 sind nämlich drei Aliase definiert worden, damit der Server auch aus allen Subnetzen direkt erreichbar ist.

Am besten liest man diese Abbildung von unten nach oben. Auf dem Namen des untersuchten Computers stehen die eingebauten Netzwerkkarten mit ihren Namen. Dabei deutet der Name eth0:0 mit dem Doppelpunkt auf einen Alias hin. Auf jedem Netzwerkgerät ist ein Quadrat mit der zugewiesenen IP-Adresse, bei denen die Netzadresse aus Platzgründen entfernt worden ist. Die Pfeile von der IP-Adresse zeigen auf das passende Subnetz, in welche die IP-Adresse liegt. Für ein einfacheres Verständnis ist das Subnetz als Intervall von IP-Adressen eingetragen. Die schwarzen Balken deuten auf die Lücken zwischen Intervallen hin.

Um einen besseren Überblick zu verschaffen, ist in Abbildung 3.4 auf Seite 25 das Routing über die Gateways graphisch dargestellt.

Auch diese Abbildung liest man am besten von unten nach oben. Diese Abbildung beschränkt sich auf die Netzwerkkarten, welche ein Routing über einen Gateway besitzen (eth0 und eth1). Auf diesen Netzwerkkarten liegen dann die Rechtecke mit den Namen der Gateways. Die erreichbaren Subnetze über diese Gateways liegen dann darüber. Dabei spielt die Netzwerkkarte eth1 eine besondere Rolle, weil diese Karte einen Default-Gateway-Routingeintrag besitzt, d.h. alle IP-Pakete, welche nicht über die anderen vorhandenen Routingeinträgen weitervermittelt werden, werden an den Default-Gateway übermittelt.

Destination	Name	Gateway	Name	Netmask	Device
192.168.215.0	RNPNETZ	*		255.255.255.240	eth0
192.168.215.32	ATMETHER	*		255.255.255.240	eth0
192.168.215.64	ATMCLIP	192.168.215.41	hprnp1	255.255.255.240	eth0
192.168.215.80	RT1NET	192.168.215.131	pcfw1ext	255.255.255.240	eth0
192.168.215.96	RT1SUB1	192.168.215.131	pcfw1ext	255.255.255.240	eth0
192.168.215.112	RT1SUB2	192.168.215.131	pcfw1ext	255.255.255.240	eth0
192.168.215.128	FIREWALLNETZ	*		255.255.255.240	eth0
192.168.215.144	RT2NET	192.168.215.132	pcfw2ext	255.255.255.240	eth0
192.168.215.160	RT2SUB1	192.168.215.132	pcfw2ext	255.255.255.240	eth0
192.168.215.176	RT2SUB2	192.168.215.132	pcfw2ext	255.255.255.240	eth0
192.168.215.224	BOOT	*		255.255.255.240	eth0
0.0.0.0	DEFAULT	141.84.218.129	gw0	0.0.0.0	eth1
141.84.218.129	gw0	*		255.255.255.255	eth1
192.168.215.16	EXTERNKOPPEL	*		255.255.255.240	eth1
192.168.215.208	NM2	*		255.255.255.240	eth2
192.168.215.192	NM1	*		255.255.255.240	eth3

Tabelle 3.3: Routing Table des Computers pcrnp10

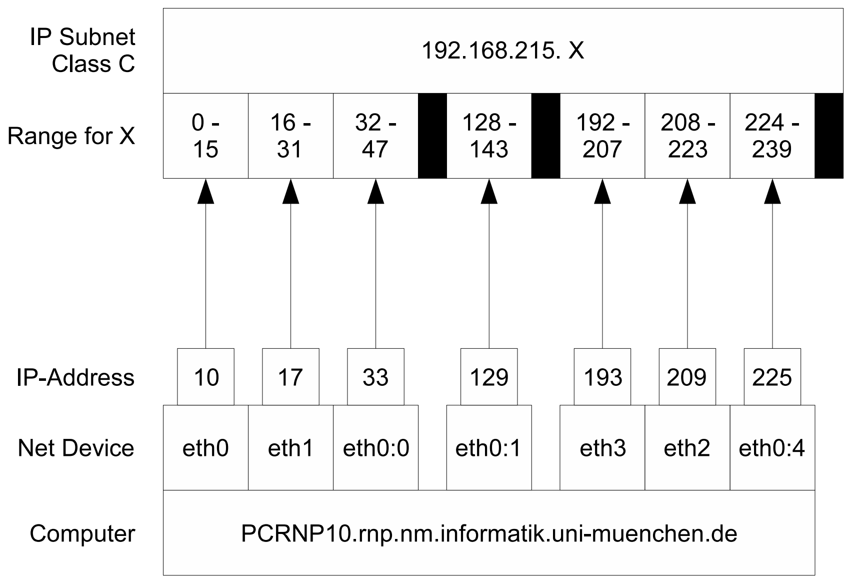

Abbildung 3.3: Netzkonfiguration des Computers pcrnp10

3.4 IT-Sicherheits-Praktikum

Nachdem das Rechnernetzpraktikum im Detail besprochen ist, soll jetzt das IT-Sicherheitspraktikum genauer untersucht werden. Da das IT-Sicherheitspraktikum sich den Rechnerraum mit dem Rechnernetzpraktikum teilen muß, verwendet es auch die selbe Hardwareinfrastruktur.

Aus diesem Grund beschreiben die folgenden Abschnitte auch nur die wesentlichen Unterschiede zum Rechnernetzpraktikum, welche vorallem auf der Softwareseite zu finden sind. Darunter gehören vorallem eine andere Netzkonfiguration, um die spezifischen Anforderungen des IT-Sicherheitspraktikum zu erfüllen.

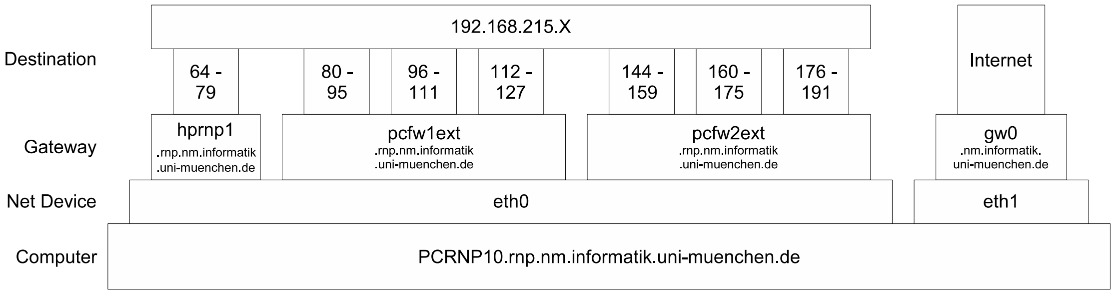

Abbildung 3.4: Routing des Computers pcrnp10

3.4.1 Adressen zu den Namen

Wie auch im Abschnitt über die Adressen und Namen im Rechnernetzpraktikum (siehe Abschnitt 3.3.2 auf Seite 21) sollen hier die verwendeten Rechnernamen und ihre zugeordneten Adressen vorgestellt werden.

Für einen ersten Überblick zeigt die Abbildung 3.5 alle Rechner für das IT-Sicherheits-Praktikum. In diesem Diagramm stehen die Rechnernamen für dieses Praktikum. Offensichtlich werden für das Praktikum nicht alle Rechner vom Rechnernetzpraktikum benutzt. Desweiteren sind die Rechner auch gruppiert (Details für die Gruppierung sind in der Tabelle A.1 auf Seite 107 vorhanden). Die beiden großen Polygone stellen dabei die Rechnergruppierung für die Hubkonfiguration dar. In der Switchkonfiguration gibt es fünf Gruppen, die im selben Diagramm mit abgerundeten Rechtecken visualisiert werden.

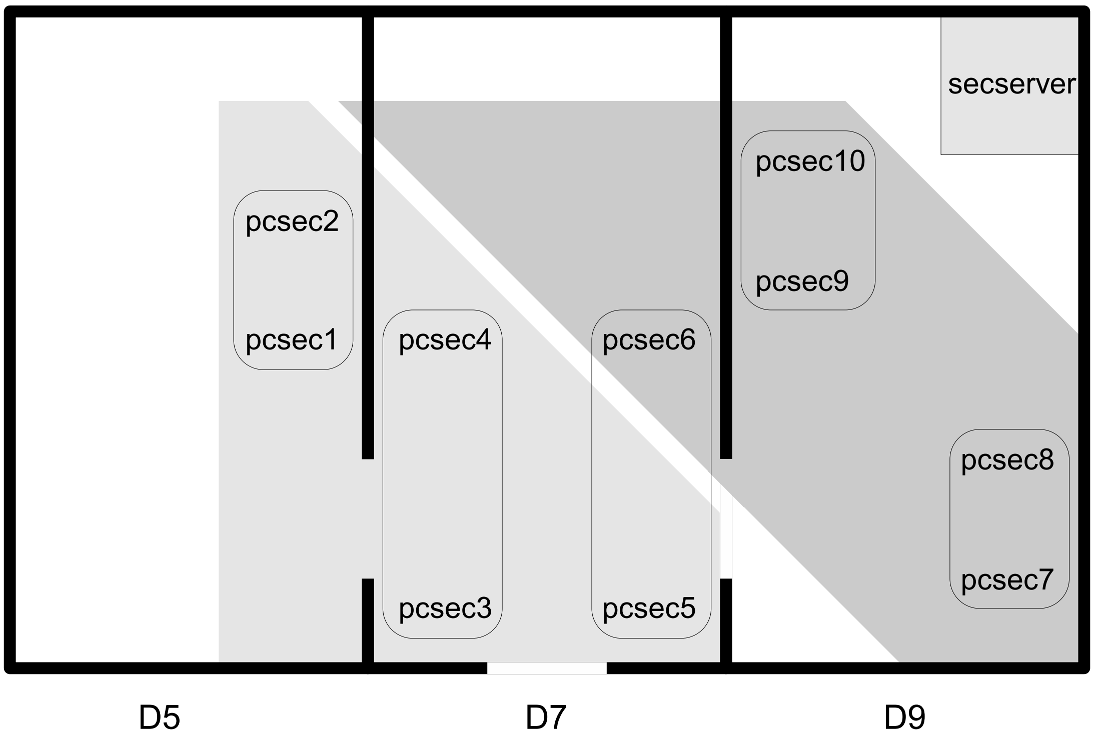

Abbildung 3.5: Raumaufteilung für das IT-Sicherheitspraktikum

DNS Konfiguration Für das Praktikum IT-Sicherheit gibt die Tabelle A.1 auf Seite 107 einen Überblick über die DNS-Konfiguration und damit über die Abbildung der IP-Adressen auf Namen. Da im IT-Sicherheitspraktikum das Netzwerk einmal mit einem Switch und ein anderes mal mit einem Hub verbunden ist, existieren zwei verschiedene DNS-Konfigurationen für die DNS-Domäne "secp.nm.informatik.uni-muenchen.de". Die beiden Konfigurationen können über ein Skript umgestellt werden, so daß immer zu der aktuellen Netzkonfiguration auch die passende DNS-Konfiguration vorhanden ist. Der zuständige DNS-Server läuft auf dem Server "secserver".

In der ersten Spalte der Tabelle A.1 auf Seite 107 stehen die verwendeten IP-Adressen. Dann folgt in der zweiten Spalte die DNS-Namen für die Hubkonfiguration und in der dritten Spalte die Namen für die Switchkonfiguration. Falls in einer der Konfigurationen kein Namen für die IP-Adresse existiert, bleibt an dieser Stelle die Tabelle einfach leer.

DHCP Konfiguration für das IT-Sicherheitspraktikum Auch für das IT-Sicherheitspraktikum ist ein DHCP-Server konfiguriert, jedoch wird das DHCP nur zum Booten des Betriebssystems über Netzwerk benötigt (siehe Abschnitt 3.2.3 auf Seite 17). Die Namen in der Tabelle A.5 auf Seite 110 sind deswegen nur zum Booten aktiv. Sobald das Betriebssystem läuft wird die Netzwerkkonfiguration aus dem DHCP verworfen. Die Konfiguration für das laufende Betriebssystem ist dann Teil der Praktikumsaufgaben.

3.4.2 Netzkonfigurationen

Normalerweise ist für ein Ethernetnetzwerk eine Verbindung über einem Switch optimal, weil damit jeder beteiligte Rechner seine volle Bandbreite ausnutzen kann. Diese hohe Bandbreite kann dazu im Gegensatz nicht mit einem Ethernethub oder einer Busverkabelung erreicht werden, weil dann das Transportmedium von allen beteiligten Rechnern gemeinsam benutzt wird.

Jedoch besitzt die Verwendung eines Hubs in einem Sicherheitspraktikum einen wesentlichen Vorteil: Da jeder Rechner den kompletten Netzwerkverkehr mithören kann, kann im Praktikum das Mitlesen einer Kommunikationsverbindung zwischen zwei anderen Rechnern ausprobiert werden. Aus diesem Grund braucht das IT-Sicherheitspraktikum auch eine Netzkonfiguration über einen Hub. Die genaue Netzkonfiguration der Rechner erkennt man in der Abbildung 3.6 auf der nächsten Seite.

Jedoch ermöglicht der Einsatz eines managebaren Switches innerhalb des Sicherheitspraktikum die einfache Realisierung verschiedener Netztopologien. Die Netztopologien können mit einer einfachen Änderung der VLAN-Konfigurationen am Switch simuliert werden. Die Änderung der VLANs ist dabei erheblich einfacher und sicherer als das Umstecken der Kabel an einem Patchfeld. Diesen Vorteil kann man in der Tabelle 3.4 auf Seite 29 erkennen, weil dort zwei Switchkonfiguration existieren ("Stern" und "Doppelstern"). Einen Überblick über die genaue Aufteilung der Rechner für die Sternkonfiguration erhält man in der Abbildung 3.7 auf Seite 28.

Die beiden Abbildungen 3.6 und 3.7 auf Seite 28 sind aus den Praktikumsunterlagen[4] entnommen. Freundlicherweise stellte Dr. Helmut Reise diese beiden Abbildung für diese Arbeit zur Verfügung.

3.4.3 VLAN-Konfiguration

Für die Unterstützung der unterschiedlichen Netzwerktopologien für das IT-Sicherheitspraktikum existieren drei unterschiedliche VLAN-Profile mit den Namen "Hub", "Stern" und "Doppelstern". Die genaue VLAN-Konfiguration der Switches ist in der Tabelle 3.4 auf Seite 29 erkennbar. Wie auch für das Rechnernetzpraktikum (siehe Abschnitt 3.3.3 auf Seite 22) werden diese Profile automatisch über den Aufruf eines entsprechenden Skriptes erzeugt.

[4]Praktikumsunterlagen: `http://wwwhegering.informatik.tu-muenchen.de/Praktika/ss04/secp/`

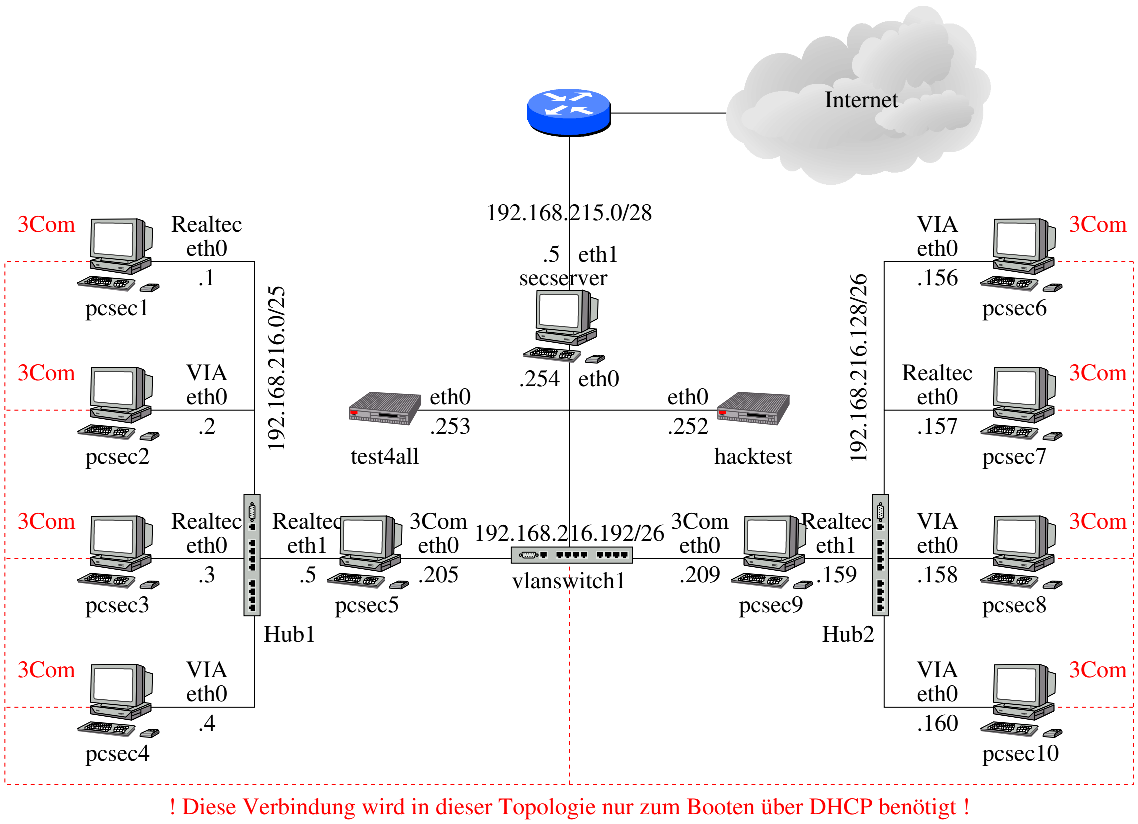

Abbildung 3.6: Hubkonfiguration für das IT-Sicherheitspraktikum

3.5 Administration

Im vorhergehenden Abschnitt sind sehr viele technische Details beschrieben, aber nicht die Gründe für die Einführung von ITIL-Service-Support-Prozessen für die Administration. Es ist offensichtlich, daß die hohe Komplexität eine gut organisierte Administration benötigt, damit die beiden Praktika auch regelmäßig auf einem hohen Niveau abgehalten werden können.

In diesem Abschnitt sollen vorallem die spezifischen Herausforderungen dieses Szenarios beschrieben werden und weshalb die Einführung von ITIL-Prozessen diese Herausforderungen unterstützen kann. Weil die aktuelle Situation für die Administratoren nicht befriedigend ist, werden auch die aktuelle Probleme beschrieben. Diese Probleme stehen auch im engen Zusammenhang mit den Herausforderungen und die Lösung dieser Probleme ist ein wesentlicher Antrieb für die Umsetzung der ITIL-Prozesse.

3.5.1 Herausforderungen

In diesem Abschnitt sollen die Herausforderungen an die Administration des Rechnerraums hervorgehoben werden. Diese Herausforderungen sind ein wesentlicher Grund für die Notwendigkeit der ITIL Service Support Prozesse.

Verfügbarkeit Um zwei Praktika innerhalb des selben Semesters in einem Praktikumsraum abzuhalten, ist eine genaue Zeiteinteilung der Teilnehmergruppen beider Praktika notwendig. Aber genau diese enge Zeiteinteilung stellt auch gleich sehr hohe Anforderungen an die Verfügbarkeit der Rechner, weil ein

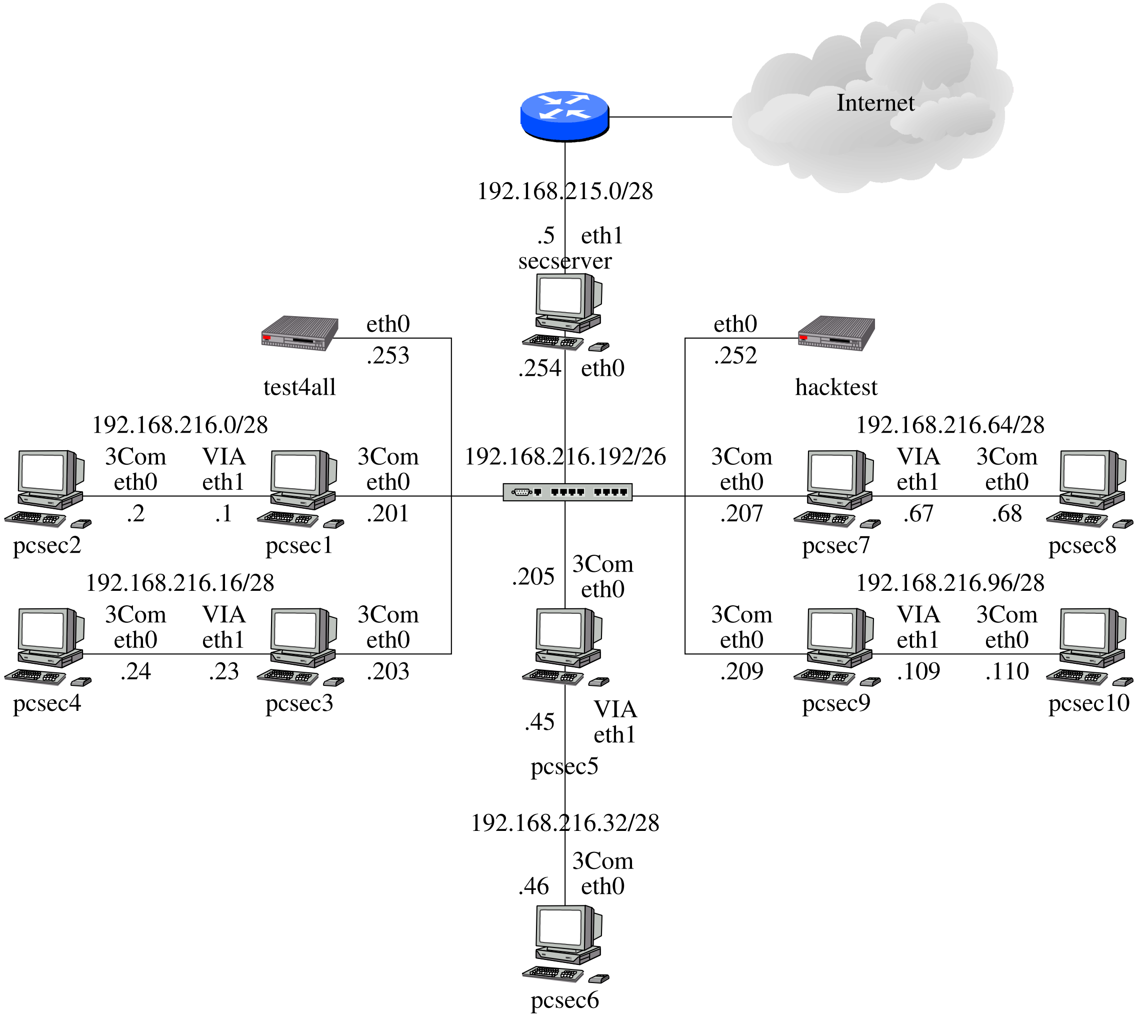

Abbildung 3.7: Switchkonfiguration für das IT-Sicherheitspraktikum

Ausfall den kompletten Zeitplan durcheinander bringen. Damit die geforderete Verfügbarkeit auch gewährleistet werden kann, müssen die Rechner professionell administriert werden.

Mehrere Konfigurationsprofile Eine spezielle Herausforderung ist die Unterstützung mehrerer Konfigurationsprofile für einen Rechnerraum (siehe Abschnitt 3.2.2 auf Seite 17). Diese Profile erhöhen die Komplexität der Administration erheblich, weil nicht nur eine Konfiguration fehlerfrei arbeiten muß, sondern es müssen immer alle Konfigurationen aus jedem Profil perfekt funktionieren. Damit die Komplexität unter Kontrolle gebracht werden kann, benötigen die Administratoren eine ausführliche und aktuelle Dokumentation des Rechnerraums.

Änderungen Eine weitere Herausforderung folgt aus den unterschiedlichen Profilen. Diese Profile werden täglich zweimal gewechselt, so daß auch jedem Arbeitstag beiden Praktika der Rechnerraum zur Verfügung steht. Dieser Wechsel erschwert aber auch die Administration, weil jeder Wechsel fehlerfrei arbeiten muß, damit die resultierende Konfiguration auch funktionstüchtig ist.

ID	Name	Switch	Portnummer
Hub			
4	PcSec5_PcSec9_	1	1, 3, 5, 7, 9, 11, 14, 16, 18, 20, 22
	SecServer_und_boot	2	3, 15, 25, 26
Stern			
5	Stern	1	1, 3, 5, 7, 9, 11, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22
		2	3, 25, 26
Doppelstern			
2	ZweiPortPCSec1PCSec2	1	13, 14
4	ZweiPortPCSec3PCSec4	1	15, 16
6	ZweiPortPCSec5PCSec6	1	17, 18
8	ZweiPortPCSec7PCSec8	1	19, 20
10	ZweiPortPCSec9PCSec10	1	21, 22
15	UngeradeSecPCs	1	1, 3, 5, 7, 9, 11
16	ZweiPortSecServerRestDerWelt	1	23, 24

Tabelle 3.4: VLAN Konfiguration für das IT-Sicherheitspraktikum

Keine Einschränkung der Nutzer Eine dritte Herausforderung stellt sich aus dem Themengebiet der Praktika, weil beide Praktika ein sehr systemnahes Thema behandeln. Aus diesem Grund erhalten alle Teilnehmer vollständigen Zugriff auf die Rechner im Rechnerraum. In der Natur eines Praktikums liegt dann auch das Ausprobieren verschiedener Konfigurationen durch die Teilnehmer. Da jedoch die Teilnehmer die Themen erst durch das Praktikum lernen, können beim Ausprobieren natürlicherweise auch Fehler passieren. Diese Fehler müssen deshalb auch von den Administratoren berücksichtigt werden. Und damit erhöht der vollständige Zugriff nochmals die Schwierigkeit und die Komplexität zur Bereitstellung der Verfügbarkeit.

Personenwechsel Ein weiterere Herausforderung für die Administration sind die häufigen Personenwechsel. Dies kann mit der Übertragung der Administrationsaufgabe auf Doktoranden erklärt werden. Aber nach der Beendigung ihrer Doktorarbeit endet häufig auch die Administrationsaufgabe und diese Aufgabe wird an den Nachfolger übergeben. Dann muß sich der Nachfolger erst wieder in die Materie einarbeiten und in dieser Zeit ist es sehr schwer, notwendige oder große Änderungen durchzuführen. Weitere Problemen ergeben sich dann, wenn keine vollständige und korrekte Dokumentation des Rechnerraums vorhanden ist.

Fazit Die aufgeführten Herausforderungen erschweren die Administration eines kleinen Rechnerraums erheblich. Für die Bewältigung dieser Herausforderungen können die Erfahrungen aus großen Unternehmen sehr nützlich sein, weil dort auch sehr komplexe Konfigurationen behandelt werden müssen. Deshalb ist die Einführung einiger ITIL Prozesse auch für dieses kleine Szenario sinnvoll. Weil aber aktuell nicht diese Prozesse benutzt werden, ergeben sich momentan eine Menge von Problemen.

3.5.2 Probleme

Die Herausforderungen aus dem letzten Abschnitt bilden unter anderem die Ursachen für die Probleme, die aktuell bei der Administration auftreten. Aus diesem Grund soll in diesem Abschnitt diese Probleme einmal genauer untersucht werden.

Komplexe Fehlersuche Die beiden Herausforderungen der schnellen Konfigurationswechsel und der volle Zugriff der Teilnehmer auf die Rechner erschwert die Fehlerbehandlung erheblich. Einerseits ist die

Ursachenanalyse sehr komplex, weil die auslösende Fehlkonfiguration von vielen Faktoren, beispielsweise vom eingesetzten Konfigurationsprofil und den Tätigkeiten der Teilnehmer, abhängt.

Und andererseits muß jede Änderung zur Problembehebung in verschiedenen Konfiguration stabil laufen. Dies kann bedeuten, daß ein Fehlerbehebung für das eine Praktikum einen neuen Fehler im anderen Praktikum erzeugen kann. Aus diesem Grund müssen alle Änderungen in mehreren Konfigurationen ausführlich getestet werden, bevor sie übernommen werden.

Schlechte Kommunikation Die beteiligten Personen kommunizieren momentan entweder direkt (im engl. "face to face") oder über eMail und Telefon. Dabei entscheidet jeder selbst, wem er welche Informationen weitergibt. Diese Freiwilligkeit kann aber zu Situationen führen, in der eine Person nicht eine wichtige Information enthält, weil der Besitzer der Information nicht alle rechtzeitig informiert hat.

Dabei ist es offensichtlich, daß ausergewöhnliche Informationen sich schneller verbreitern, als regelmäßige Statusmeldungen. Jedoch können diese regelmäßigen Meldungen über übliche Tätigkeiten in vielen Situationen die Arbeit der Administration erleichtern, weil sie vielleicht Hinweise auf neue Probleme oder Störungen enthalten können.

Durch die Einführung der ITIL Prozesse soll die Kommunikation zwischen den Personen formalisiert werden, so daß jeder seine notwendigen Informationen auch erhält.

Lückenhafte Dokumentation Mit einer guten Dokumentation kann die Administration eines Rechnerraums erheblich erleichtert werden, weil schon in der Planungsphase alle notwendige Daten sehr leicht zu beschaffen sind und nicht erst durch aufwendige Tests selber erzeugt werden müssen. Aber durch den häufigen Personenwechsel existieren verschiedene Formate der Dokumentation, weil jeder neue Administrator seinen eigenen Stil im Bereich der Dokumentation pflegt. Dadurch entstehen in der Dokumentation auch Lücken, weil nicht sofort erkennbar ist, was schon dokumentiert ist, und welche Information noch in der vorhandenen Dokumentation fehlen.

Zusätzlich erzeugen die verschiedenen Konfigurationsprofile für die beiden Praktika mehr Dokumentationsaufwand als für einen üblichen Rechnerraum in dieser Größe.

Auch dieses Problem erfordert die Einführung der ITIL-Service-Support-Prozesse, weil durch das Configuration und Change Management eine gute Grundlage für eine konsistente Dokumentation geschaffen wird. Jedoch entsteht eine gute Dokumentation nicht durch die Einführung der ITIL-Prozesse, sondern durch eine ständige Verbesserung der aktuellen Dokumentation. Der Nutzen einer guten Dokumenation sind die Erleichterungen bei der Administrationsarbeit, weil fehlende Informationen einfach nachgeschlagen werden können und nicht mühsam erst selbst erarbeitet werden müssen.

3.6 Fazit

Das hier vorgestellte Szenario zeigt die Gründe für die Einführung von ITIL-Service-Support-Prozessen, weil die Ansprüche an den Rechnerraum wegen der hohen Komplexität momentan nicht befriedigend erfüllt werden können. Deshalb wird im nächsten Kapitel die ITIL Prozesse im Detail vorgestellt, welches dann der erste Schritt für die Verbesserung der Administrationstätigkeit darstellt.

Bei der Zusammenstellung der technischen Informationen über den Rechnerraum hat sich auch herausgestellt, daß nur sehr wenig oder lückenhafte Dokumentation über den Rechnerraums existieren. Dies ist dann auch als Problem des Szenarios beschrieben worden (siehe dazu den Abschnitt 3.5.2). Deshalb sind die meisten Daten über diesen Rechnerraum durch manuelle Nachforschungen entstanden und sind nicht aus der vorhandenen Dokumentation entnommen worden. Damit stellt auch schon die Beschreibung des Szenarios eine gute Informationsquelle für die Administratoren dar, weil hier der Rechnerraum einheitlich auf dem aktuellen Stand dokumentiert wird. Zusätzlich dienen die gesammelten Daten auch für die

Erstellung der Ausgangskonfiguration im Configuration Management (siehe dazu den Abschnitt 7.2.2 auf Seite 91).

4 ITIL-Prozesse für Administration

In diesem Kapitel sollen die spezifischen Prozesse für die Administration des Praktikumrechnerraums beschrieben werden. Diese Prozesse folgen dabei den ITIL-Empfehlungen und sollen damit eine effektive Arbeit der Administratoren ermöglichen. Um diese Empfehlungen am Besten umzusetzen, basieren die Prozesse auf direkt auf den ITIL Empfehlungen (siehe dazu das Buch [ITIL 00]).

Da die Empfehlungen sehr allgemein gehalten sind, sind die hier beschriebenen Prozesse an das Szenario angepasst worden. Jedoch enthalten Sie keinen direkten Bezüge auf das Szenario, damit die Prozesse auch für andere einsetzbar bleiben.

4.1 Aufbau

Der Aufbau der Prozessbeschreibung orientiert sich an der Struktur, welche bei der Einführung in ITIL beschrieben ist (siehe dazu den Abschnitt 2.2.1 auf Seite 5). Dazu wird jeder Prozess in einem eigenen Abschnitt vollständig beschrieben. Um den engen Zusammenhang zwischen der Einführung und diesem Kapitel hervorzuheben, gibt es für jeden Prozess eine kurze Übersicht in einer Tabelle, welche sich an der Abbildung 2.2 auf Seite 6 orientiert.

Nach einer kurzen Einführung wird das Ziel des Prozesses nochmal vorgestellt. Dabei werden in der Zielbeschreibung auch die möglichen Leistungsindikatoren zur Überprüfung der gesteckten Ziele aufgeführt. Anhand der Zieldefinition kann der Prozessmanager seinen Prozess weiter optimieren.

Danach folgt eine Liste der Rollen mit ihrer Beschreibung, welche die Prozesstätigkeiten ausführen sollen. Diese Rollen sind sehr allgemein gehalten, damit in der Prozessbeschreibung keine Organisationsstruktur implizit vorgegeben wird. Zu den aufgezählten Rollen gehören auch immer der Prozessinhaber und der Manager dazu. Die Aufgaben dieser beiden Rollen ist ebenfalls in der Einführung beschrieben (siehe dazu 2.2.1 auf Seite 6).

Nach der Vorstellung der Rollen beginnt die ausführliche Erklärung der einzelnen Tätigkeiten. Die Tätigkeiten beschreiben dann den genauen Ablauf innerhalb des Prozesses und stellen somit den Kern jedes Prozesses dar. Um die Zusammenhänge und die Abläufe innerhalb eines Prozesses besser zu visualisieren, besitzt jeder Prozess auch eine Abbildung mit dem Flowchart[1].

In diesen Flowcharts wird jede Tätigkeit mit einem Rechteck dargestellt und die Abläufe innerhalb eines Prozesses werden mit den eingezeichneten Pfeilen symbolisiert. Manchmal kann aber der Prozessablauf sich variieren. Dann zeigen die Rauten das Kriterium für die Wahl der möglichen Alternativen. Eine besondere Bedeutung besitzen die Rechtecke mit den Doppelrändern an der linken und rechten Seite: Mit diesen Symbolen werden die anderen ITIL-Prozesse dargestellt.

Schon in den Flowcharts erkennt man auch die Zusammenarbeit der einzelnen Prozesse. Und diese Zusammenarbeit wird dann jeweils für jeden Prozess am Ende genauer erläutert. Dadurch ergibt sich ein Gesamtbild aller Tätigkeiten für die Administration, welches auf den einzelnen Prozessen aufbaut.

[1] Die Flowcharts entsprechen dabei dem Standard ISO Norm 5807

Incident Management	
Ziel:	schnelle Behebung aller Störungen
Qualitätsparameter:	Störungen sind nachweisbar behoben
Leistungsindikatoren:	Dauer zur Behebung der Störungen
Input:	Störungsmeldungen vom Service Desk, Mitarbeiter oder Kunden
Output:	Änderungen in Form von RfC für das Change Management
Resourcen:	Informationen über die Konfigurationen der betroffenen Configuration Items
Rollen:	Störungsmelder, First-Level-Support, Second-Level-Support, Third-Level-Support

Tabelle 4.1: Prozessübersicht: Incident Management

4.2 Incident Management

Im Incident Management werden die Störungen bearbeitet und behoben. Dazu gehört vorallem die Aufzeichnung und schnellen Behebung aller Störungen. Da in dieser Arbeit kein Service Desk umgesetzt wird, übernimmt das Incident Management auch die Erfassung aller Störungen. Diese Vereinfachung ist wegen der kleinen Personenanzahl im Szenario möglich und am effektivsten (siehe dazu den Abschnitt 3.1 auf Seite 16).

4.2.1 Ziel

Das Ziel des Incident Management ist die schnelle Behebung aller Störungen, damit der reibungslose Arbeitsablauf der Organisation bzw. des Unternehmens gewährleistet werden kann.

Der Erfolg des Incident Management kann über die Anzahl aller Störungen und die jeweilige Zeitdauer für die Behebung einer Störungen gemessen werden. Die Effizienz kann dann zusätzlich über einer Senkung der Kosten weiter optimiert werden.

4.2.2 Rollen

Eine Besonderheit bei den Rollen ist der First-, Second-, und Third-Level Support, weil alle drei Supportlinien die selbe Aufgabe besitzen und damit streng genommen auch nur durch eine Rolle representiert werden kann. Jedoch besizen diese drei Linien unterschiedliche Kompetenzen, so daß eine organisatorische Trennung gerechtfertigt ist (siehe dazu auch die Kritik im Abschnitt 2.4.2 auf Seite 12).

Störungsmelder Die Rolle des Störungsmelders beschreibt alle Personen, die eine Störung dem Incident Management melden. Ohne einen Service Desk müssen die Störungsmelder entweder selber eine Störungsmeldung verfassen oder jemanden die Störung erklären, der dann für sie die Störungsmeldung eingibt. In diesem Fall sind beide in der Rolle des Störungsmelders. Auch muß der Störungsmelder nicht immer von der Störung selbst betroffen sein, sondern er kann beispielsweise auch nur die Störung beobachtet haben.

First-Level-Support Damit das Incident Management viele Anfragen annehmen kann, existiert der First-Level-Support, der alle Störungen als erstes annehmen muß. Falls die Störung zu komplex für eine schnelle Bearbeitung innerhalb des First-Level-Support ist, kann die Meldung an den Second-Level-

Support weitergereicht werden. Im First-Level-Support sollen vorallem einfache und regelmäßig auftauchende Störungen bearbeitet werden, für die es schon Workarounds oder fertige Lösungen vorhanden sind.

Second-Level-Support Für eine ausführlichere Bearbeitung einer Störung ist der Second-Level-Support zuständig. Dort werden die Störungen behandelt, für die keine offensichtliche Lösung schon vorhanden sind. Falls jedoch für eine Störung spezielles Fachwissen benötigt wird, welche nicht im Second-Level-Support verfügbar ist, dann wird die Störung an den Third-Level-Support weitergereicht.

Third-Level-Support Im Third-Level-Support sollen dann die Störungen angenommen werden, die nur noch von Experten bearbeitet werden können. Die Experten arbeiten häufig in anderen Bereichen und durch diese Arbeit erhalten sie das spezifische Expertenwissen, welches im Incident Management benötigt wird. Und darum soll der Third-Level-Support immer eine Ausnahme sein, damit die Experten ihrer Hauptbeschäftigung nachgehen können.

4.2.3 Aktivitäten

Die Abbildung 4.1 auf der nächsten Seite gibt eine grobe Übersicht über die Aktivitäten des Incident Managements. Eine Besonderheit in dieser Arbeit ist die Erfassung der Störung und die erste Unterstützung, weil diese Tätigkeiten implizit die Aufgaben des nicht vorhandenen Service Desk übernehmen. Als weiterer Input ist das Configuration Management eingezeichnet, damit auch die erfaßten Störungen auf einer gemeinsamen Datenbasis stehen.

Der Output des Incident Managements ist eine RfC für das Change Management, welche ebenfalls in der Abbildung 4.1 auf der nächsten Seite eingezeichnet ist.

Erfassung Beim Auftreten einer Störung soll als erstes überprüft werden, ob die Störung reproduzierbar ist. Falls es sich um eine reproduzierbare Störung handelt, kann später die Störung leichter behoben werden. Idealerweise wird vor der Eingabe geprüft, ob eine ähnliche Störung in letzter Zeit schon gemeldet worden ist, damit keine mehrfache Meldung der selben Störung vorgenommen wird.

Wenn alle Überprüfungen plausibel auf eine neue Störung schliessen lässt, dann muß eine Störungsmeldung für das Incident Management erstellt werden. Dabei sollen folgende Informationen in einer Störungsmeldung enthalten sein:

- Name und Adresse für Rückfragen des Störungsmelders
- Liste von betroffenen Personen
- genaue Beschreibung der Störung mit Soll- und Istverhalten
- betroffene Hardware bzw. Software aufzählen
- genaue Uhrzeit und Datum des ersten Auftretens
- Ablauf zum Reproduzieren der Störung
- mögliche und eingetretenen Auswirkungen
- versuchte Behebungsversuche
- bekannte Workarounds

Nach der Eingabe der Störung erhält man eine eindeutige Störungsidentifikationsnummer unter der man die eingegebene Störung jederzeit wiederfinden kann. Diese Nummer kann bei der weiteren Bearbeitung der Störung sehr hilfreich sein, weil damit immer eindeutig eine Störungsmeldung identifiziert werden kann.

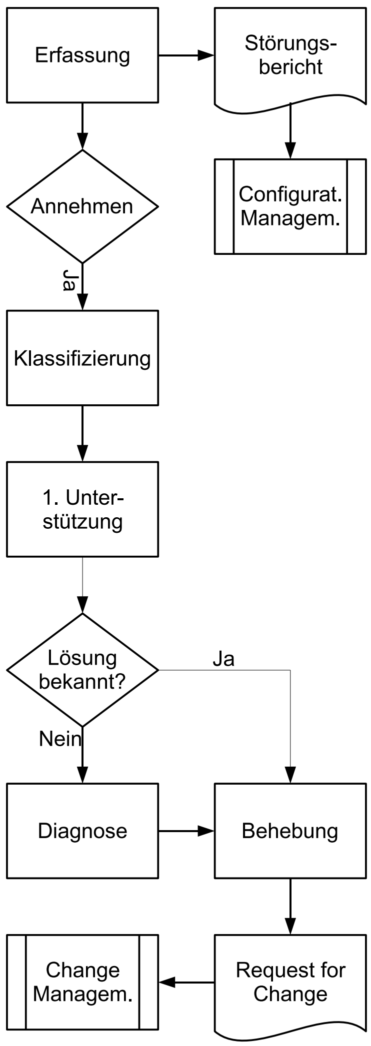

Abbildung 4.1: Übersicht über die Aktivitäten des Incident Managements

Annehmen Das Annehmen einer Störung ist ein Entscheidungsprozess, ob die Störungsmeldung eine echte Störung beschreibt. Keine Störungen sind Auswirkungen von erwünschten Verhalten der IT-Infrastruktur. Beispielsweise können Einschränkungen aus Sicherheitsgründen als eine Störung missverstanden werden. Diese Entscheidung kann deshalb erheblich vereinfacht werden, wenn eine vollständige Dokumentation der IT-Infrastruktur vorliegt

Eine Störung soll nur abgelehnt werden, wenn klar feststeht, daß die Störung ein erwünschtes Verhalten ist. Damit soll das Ablehnen einer echten Störung so weit wie möglich vermieden werden.

Klassifizierung Da jeder Betroffene eine sofortige Behebung der Störung wünscht, aber dies wegen den begrenzten Resourcen nicht sofort möglich ist, müssen alle Störungen klassifiziert werden. Mit der Klassifizierung können dann wichtige Störungen erkannt werden und dann auch schneller behoben werden. Dazu teilt die Klassifizierung die Störungen in unterschiedliche Kategorien nach einer vorgegebenen Richtlinie ein.

Unter folgende Gesichtspunkte kann Klassifizierung durchgeführt werden:

- Größe der Auswirkung
- Dringlichkeit
- Komplexität

Aus diesen Gesichtspunkten kann dann eine Priorität für die Störung ermittelt werden. Die Priorität kann dann die Reihenfolge der Bearbeitung und später auch die verfügbaren Resourcen zur Behebung beeinflussen.

Erste Unterstützung Als erste Unterstützung für die Betroffenen soll ihnen eine provisorische Lösung angeboten werden. Diese besteht häufig aus einer Beschreibung eines Workarounds und soll den Betroffenen kurzfristig helfen, damit sie ihre Arbeit weiter fortsetzen können. Jedoch kann die erste Unterstützung nicht garantiert werden, weil nicht für alle Störungen ein Workaround vorhanden ist.

Lösung bekannt? Hier soll recherchiert werden, ob schon eine Lösung für die Störung schon vorhanden ist. Dabei ist die erste Quelle die Beschreibungen von alten Störungsmeldungen, weil dort die jeweiligen Vorgänge zur Behebung beschrieben ist. Diese Recherche nach einer Lösung kann sich auch auf das Problem Management ausweiten, weil eventuell die Ursache einer Störung ein bekanntes Problem ist, welches gerade im Problem Management behandelt wird. Jede mögliche Lösung muß dann in der Störungsmeldung dokumentiert werden. Falls schon eine Lösung vorhanden ist, kann die Diagnose für diese Störung ausbleiben.

Diagnose Die Erforschung der Ursachen einer Störung ist in komplexen Systemen sehr aufwendig, weil die Symptome einer Störung von vielen Ursachen abhängen können und damit die Lokalisierung erheblich erschwert wird.

Falls die Störung reproduzierbar ist, kann man bei der Diagnose sehr systematisch vorgehen und somit den Aufwand überschaubar halten. Bei diesem Vorgehen grenzt man die möglichen Ursachen ein, indem man durch eine gezielte Veränderung, zum Beispiel den Austausch einer möglichen defekten Komponente, eine Menge von Ursachen ausschliest. Durch wiederholtes Eingrenzen der Ursachen findet man irgendwann die Menge der Ursachen, welche die Störung auslöst.

Bei nicht-reproduzierbaren Störungen ist die Diagnose jedoch erheblich schwieriger, weil keine sofortige Überprüfung einer Ursachenhypothese möglich ist. Dann soll versucht werden, soviel wie möglich an Informationen beim Auftreten der Störung zu sammlen, damit man bei der Diagnose den Ablauf bis zum Auftreten der Störung nachvollziehen kann. Das Sammeln der Informationen erfolgt meistens durch die ausführliche Protokolierung aller Aktivitäten des Systems. Am Ende muß die Diagnose die Ursachen für eine Störung eindeutig bestimmt haben, so daß im nächsten Schritt sie auch behoben werden können.

Behebung Nach der Diagnose stehen die Ursachen für eine Störung fest und somit ergeben sich daraus automatisch mögliche Behebungsstrategien. Die Störung soll dann so schnell und so weitgehend wie möglich behoben werden. Dazu werden die notwendigen Änderungen in einer RfC dokumentiert, damit diese Änderung vom Change Management überwacht werden kann.

Der Erfolg der Änderung muß auf jeden Fall überprüft werden, um festzustellen, ob eine Störung auch korrekt und vollständig behoben worden ist. Sobald jemand die Lösung einer Störung bestätigen kann, wird die Störungsmeldung erfolgreich abgeschlossen. Dazu müssen auch alle Betroffenen einer Störung informiert werden, so daß sie wieder ihre normale Arbeit fortführen können.

Problem Management	
Ziel:	Umfassende Lösung von Problemen
Qualitätsparameter:	Wettbewerbvorteile durch die Behebung der Probleme
Leistungsindikatoren:	erzielte Produktivitätssteigerung
Input:	Problembeschreibungen
Output:	Änderungen in Form von RfCs
Resourcen:	Problemanalyse
Rollen:	Problemmelder, Problemanalyst, Problemkoordinator

Tabelle 4.2: Prozessübersicht: Problem Management

4.2.4 Zusammenarbeit

Das Incident Management benötigt für aussagekräftige Störungsmeldungen die Informationen aus dem Configuration Management. Und für die Behebung der Störungen benötigt es auch das Change Management.

Configuration Management Damit die betroffenen Geräte oder Programme einer Störung eindeutig identifiziert werden können, benötigt das Incident Management die Informationen der Configuration Items aus dem Configuration Management.

Später können die Beziehungen zwischen den Configuration Items die Diagnose vereinfachen, weil damit die Ursachen einfacher identifiziert werden können. Jedoch reichen häufig diese Informationen nicht aus, so daß auch eine manuelle Nachforschung der Ursachen nicht ausgeschlossen werden kann.

Change Management Das Ziel jeder Störungsmeldung ist die Beseitigung der Störung und dies erfordert häufig eine Änderung an der IT-Infrastruktur. Damit diese Änderungen auch kontrolliert durchgeführt werden können, werden die notwendigen Änderung zur Behebung einer Störung als RfC für das Change Management verfaßt. Die Ergebnisse einer durchgeführten RfC muß dann das Change Management dem Incident Management melden, damit die Störungsmeldung erfolgreich abgeschlossen werden kann.

4.3 Problem Management

Das Incident Management versucht so schnell wie möglich alle Störungen zu beheben. Aus diesem Grund kann das Incident Management ein wiederholtes Auftreten einer Störung nicht verhindern. Diese Aufgabe besitzt aber das Problem Management, weil es die Probleme grundlegend beheben soll.

4.3.1 Ziel

Das Ziel des Problem Managements ist eine umfassende Behebung der Probleme. Eine wichtige Grundlage dazu ist die Identifzierung aller Probleme, weil nur wenn die Probleme bekannt sind, können sie auch behoben werden. Neue Probleme können am Besten aus der Analyse der Störungsmeldungen erkannt werden, damit die Problemlösung die Anzahl der Störungen verringern kann und damit auch die Kosten für das Incident Management senkt.

Die Behebung jedes Problems soll dem Unternehmen in der Zukunft einen Wettbewerbsvorteil oder eine Produktivitätssteigerung ermöglichen. Daraus kann dann auch die Effektivität des Prozesses gemessen werden.

4.3.2 Rollen

Die Rollen spiegeln die drei Aufgaben des Prozesses wieder: Identifizieren, Untersuchen und Beheben. Damit auch komplexe Probleme effizient bearbeitet werden können, erhält der Problemkoordinator die Verantwortung für die Bearbeitung seiner Probleme.

Problemmelder Die Identifizierung neuer Probleme ist die Hauptaufgabe des Problemmelders. Immer wenn er neue Probleme gefunden hat, erstellt er dann für das Problem Management einen Problembericht. Eine häufige Quelle für neue Probleme sind die Störungsmeldungen im Incident Management.

Problemanalyst Die Aufgabe des Problemanalysten ist die Untersuchung der Ursachen für ein beschriebenes Problem. Für diese Tätigkeit ist das spezifische Expertenwissen für den jeweiligen Bereich des Problemes notwendig, damit die Ursachen und Auswirkungen eines Problems korrekt ermittelt werden können.

Problemkoordinator Der Problemkoordinator organisiert dann die Bearbeitung der Probleme. Und damit ist er für die erfolgreiche Bearbeitung seiner Probleme verantwortlich. Dabei kann er einzelne Tätigkeiten, wie die Diagnose oder die Implementierung der Lösung, an andere Personen delegieren. Dabei muß er auch kontrollieren, daß der Aufwand für die Behebung eines Problems dem möglichen Nutzen nicht übersteigt.

4.3.3 Aktivitäten

Die Abbildung 4.2 auf der nächsten Seite gibt eine Übersicht über die Tätigkeiten des Problem Managements. In dieser Abbildung ist nur eine Quelle, nämlich das Incident Management, für neue Probleme eingezeichnet, obwohl neue Probleme auch aus allen anderen Prozessen herkommen können. Ein weiterer Input für Informationen ist wie beim Incident Management das Configuration Management, damit die Ursachen und Auswirkungen eines Problems eindeutig beschrieben werden können. Am Ende werden die notwendigen Änderungen für ein Problem in Form einer RfC für das Change Management formuliert.

Eine Besonderheit besitzt das Problem Management in ITIL, weil es zwischen einem Problem und einem bekannten Fehler unterscheidet (siehe dazu auch den Abschnitt 2.3.3 auf Seite 9). Ein Problem ist eine unerwünschte Situation, dessen Ursache noch nicht bekannt ist. Sobald die Ursache eines Problems in der Diagnose festgestellt worden ist, erhält es in den ITIL-Empfehlungen den Namen “bekannter Fehler”.

Identifizierung Das Ziel für die Identifizierung ist das Entdecken von neuen Problemen, welche durch ihre Behebung einen Wettbewerbsvorteil bringen können. Die wichtigste Informationsquelle für das Finden von neuen Problemen sind die Störungsmeldungen aus dem Incident Management. Beispielsweise kann ein häufiges Auftreten einer Störung auf ein grundlegendes Problem hinweisen, welches dann innerhalb des Problem Managements behandelt werden muß.

Jedoch können potentielle Probleme durch eine Prognosen frühzeitig entdeckt werden, damit diese auch rechtzeitig behandelt werden können. Weil neue Probleme überall auftauchen können, müssen die Problemmelder für alle Hinweise offen sein.

Erfassung Nach der erfolgreichen Identifizierung neuer Probleme erstellt der Problemmelder dann ein passenden Problembericht. Ein Problembericht soll dann mindestens folgende Informationen beinhalten:

- Allgemeine Beschreibung des Problems
- Beschreibung des aktuellen Zustandes

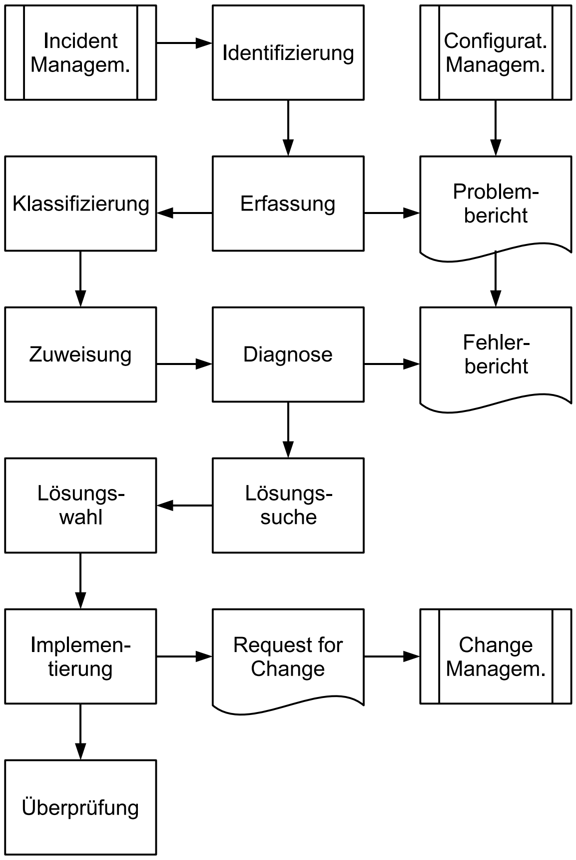

Abbildung 4.2: Übersicht über die Aktivitäten des Problem Managements

- Beschreibung des erwünschten Zustandes
- Vorteile nach der Behebung des Problems
- Auswirkungen bei keiner Behandlung des Problems
- Liste der betroffenen Personen
- Quellen für die Problemidentifizierung

Klassifizierung Da bei der ersten Erfassung eines Problemes es nur beschrieben wird, muß das Problem jetzt intensiver untersucht werden. Zuerst muß die Tragweite des Problems bestimmt werden, so daß die Auswirkungen des Problems besser sichtbar werden. Der nächste Punkt bei der Klassifizierung ist die Bestimmung der Dringlichkeit. Darin sollen die gegebenenfalls vorhandenen zeitlichen Beschränkungen zusammengefasst werden. Dann folgt aus den Beschreibungen der Auswirkungen und der Dringlichkeit die Festlegung einer Priorität, welche die weitere Bearbeitung erheblich beeinflussen kann.

Zuweisung Wenn der Umfang des Problems feststeht, müssen die notwendigen Resourcen für dieses Problem zugewiesen werden. Aus diesem Grund benötigt jedes Problem eine verantwortlichen Person, welche die weitere Koordination übernimmt. Diese Person steckt dann in der Rolle des Problemkoordinators.

Zu seinen Aufgaben gehört bei größeren Problemen das Aufstellen eines Teams, welches die weitere Bearbeitung des Problems übernimmt, und das Allokieren der notwendigen Resourcen für die Lösung des

Problems. Und sobald er sich einen groben Überblick über das Problem verschafft hat, muß für die weitere Planung ein Termin festlegen, bis wann das Problem wahrscheinlich behoben ist.

Diagnose Die erste Aufgabe des aufgestellten Teams ist die Diagnose des Problems, damit alle Ursachen für die weitere Bearbeitung bekannt sind. Dabei ist es wichtig, daß alle möglichen Ursachen untersucht werden, damit später das Problem vollständig behoben werden kann.

Das Vorgehen dabei ähnelt der Diagnose einer Störung (vgl. Kapitel 4.2.3 auf Seite 36). Weil aber dem Problem Management mehr Zeit zur Verfügung steht, soll auch die Diagnose ausführlicher und umfassender als beim Incident Management ausfallen.

Am Ende muß aber die Diagnose alle Ursachen des Problems identifiziert haben. Dieses Ergebnis muß nach einer Überprüfung dann im Problembericht dokumentiert werden. Und mit der Kenntniss der Ursachen sprechen die ITIL-Empfehlungen nicht mehr von einem Problem, sondern von einem bekannten Fehler, welche nur noch behoben werden muß.

Lösungssuche Nachdem die Ursachen bekannt sind, müssen mögliche Lösungsvorschläge für den bekannten Fehler erarbeitet werden. Es sollen dabei mehrere Vorschläge zusammengestellt werden, damit später nicht nur die offensichtliche Lösung realisiert wird, sondern die beste Lösung. Aus diesem Grund sollen sich die Lösungen in Umfang und Vorgehensweise unterscheiden.

Alle Lösungsmöglichkeiten müssen im Problembericht so dokumentiert werden, daß die Realisierung in der Beschreibung klar erkennbar ist.

Lösungswahl Damit das Ziel des Problem Managements zur Verbesserung der Produktivität auch erreicht werden kann, muß bei der Lösung eines Problems der mögliche Nutzen größer als die anfallenden Kosten der Änderung sein. Dazu benötigt der Problemkoordinator eine Schätzung der Kosten und des Nutzens für jeden angebotenen Lösungsvorschlag. Diese Daten bilden mit der Einschätzung über die Realisierbarkeit die Grundlage für die Wahl einer Lösung, welche dann im nächsten Schritt realisiert werden soll.

Damit die Lösungswahl nachvollziebar ist, soll der ganze Ablauf der Lösungswahl mit allen Fakten ausführlich dokumentiert werden. Falls sich später herausstellt, daß die Lösung doch nicht optimale war, kann man durch eine erneute Lösungssuche eine bessere Lösung finden.

Implementierung Nachdem ein Lösungsvorschlag ausgewählt worden ist, müssen die Details zur Behebung des Problems in Form einer RfC erarbeitet werden. Die Erstellung einer RfC kann dabei in enger Zusammenarbeit mit dem Change Management erfolgen, damit die Änderungen wie vorgesehen umgesetzt werden können. Die Implementierung der RfC wird dann vom Change Management überwacht.

Überprüfung Nach der erfolgreichen Implementierung müssen die durchgeführten Änderungen überprüft werden, ob das Problem vollständig und wie vorhergesehen behoben worden ist. Die Informationen für eine Überprüfung liefert die Problembeschreibung, welche am Anfang nach der Identifizierung verfaßt worden ist. Diese Beschreibung soll dann mit dem Ergebnis der Implementierung verglichen werden, um eventuelle Abweichungen zu erkennen. Zusätzlich sollen die entstandenen Kosten für die Behebung ermittelt werden, damit später die Kosten mit den eingetretenen Nutzen verglichen werden kann.

4.3.4 Zusammenarbeit

Ähnlich wie das Incident Management benötigt das Problem Management die beiden Prozesse Configuration und Change Management. Zusätzlich werden neue Probleme häufig aus der Analyse von Störungs-

	Configuration Management
Ziel:	Bereitstellung aller notwendigen Informationen über die IT-Infrastruktur
Qualitätsparameter:	Korrekte und vollständige Informationen über alle angefragten Configuration Items
Leistungsindikatoren:	Anteil der erfolgreichen Anfragen
Input:	Informationsanfragen und Änderungen der Configuration Items
Output:	Informationen über die Configuration Items
Resourcen:	vollständige CMDB
Rollen:	Architekt, Kontrolleur

Tabelle 4.3: Prozessübersicht: Configuration Management

meldungen identifiziert, so daß das Problem Management auch eng mit dem Incident Management zusammenarbeiten muß.

Incident Management Eine wichtige Quelle für neue Probleme sind die Störungsmeldungen des Incident Managements. Wenn nämlich viele ähnliche Störungen häufiger auftreten, kann die Ursache für diese Störungen auch ein neues Problem sein. Falls ein Problem durch die Analyse von Störungsmeldungen identifiziert worden ist, müssen die betreffenden Störungen auch im Problem dokumentiert werden.

Aber auch den Erfolg einer Problembehandlung kann im Incident Management beobachtet werden, wenn beispielsweise die Anzahl der Störungen abnimmt.

Configuration Management Auch das Problem Management benötigt für eine klare Beschreibung eines Problems oder der Änderung die Informationen aus dem Configuration Management. Zusätzlich ermöglichen die Informationen über die Beziehungen der Configuration Items eine einfachere Diagnose der Probleme.

Change Management Als Ergebnis liefert das Problem Management wie das Incident Management eine RfC, damit die untersuchten Probleme auch sicher behoben werden können. Zusätzlich kann es als eine weitere Informationsquelle für die Diagnose dienen, weil eventuell die Ursache mit einer Änderung des Change Managements zusammenhängt.

4.4 Configuration Management

Das Configuration Management liefert anderen Prozessen Informationen über die Konfiguration der IT-Infrastruktur. Dazu werden alle Informationen über die Configuration Items gesammelt und aufbereitet abgelegt, damit später andere bequem und effizient auf diese Informationen zurückgreifen können.

4.4.1 Ziel

Das Ziel des Configuration Management ist die Bereitstelung aktueller und korrekter Informationen über die IT-Infrastruktur.

Damit die anderen Prozesse auch einfach die Informationen nutzen können müssen sie auch komfortabel abgefragt werden können. Um damit der Prozess auch effizient arbeiten kann, dürfen nur die Daten gesammelt werden, welche auch von den anderen Prozessen benötigt werden. (siehe dazu den Abschnitt 2.4.1 auf Seite 10). Zur weiteren Effizienzsteigerung kann die Datenerfassung automatisiert werden.

4.4.2 Rollen

Die beiden Rollen befassen sich vorallem mit der Einrichtung und dem Betrieb der CMDB.

Architekt Der Architekt im Configuration Management entwirft und plant das Modell der CMDB. Seine Arbeit wird dabei von den Anfragen der anderen Prozessen, von den verfügbaren Resourcen und von der vorhandenen IT-Infrastruktur beeinflusst.

Kontrolleur Die Aufgabe des Kontrolleurs ist die Überwachung der CMDB, so daß der Inhalt der CMDB immer auf dem aktuellen Stand bleibt. Zu seinen Aufgabenbereichen gehört die Statusüberwachung, die Kontrolle und die Verifizierung der CMDB.

4.4.3 Aktivitäten

Alle Aktivitäten des Configuration Managements werden in der Abbildung 4.3 dargestellt. Man erkennt, daß alle Aktivitäten zum Betrieb der CMDB benötigt werden.

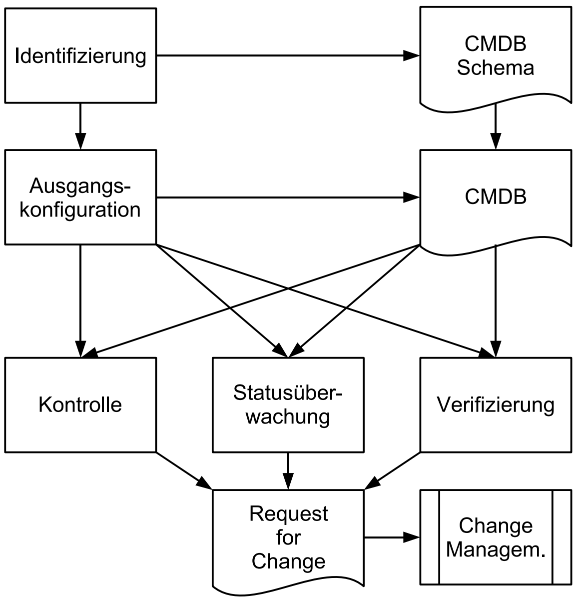

Abbildung 4.3: Übersicht über die Aktivitäten des Configuration Managements

Identifizierung Bevor die ersten Configuration Items in die CMDB aufgenommen werden können, müssen zuerst der Umfang und die Strukturen festgelegt werden.

Zuerst wird der Umfang der CMDB festgelegt, in dem eine Liste aller Betriebsmittel zusammengestellt wird, welche vom Configuration Management überwacht werden sollen.

CI spezifizieren Die erfaßten Betriebsmittel müssen dann als Configuration Items beschrieben werden. Damit die unterschiedlichen Configuration Items in der CMDB erfaßt werden können, müssen eine Menga von Schemas entwickelt werden. Jedes Schema besitzt zur Beschreibung einer Klasse von Configuration Items eine Menge von Attributen. Diese Attribute müssen dann alle Configuration Items in der Klasse vollständig beschreiben können. Auch diese Festlegung der Schemas ist ein Teil der Identifizierung.

CI Beziehungen spezifizieren Nachdem die Configuration Items mit ihren Schemas definiert sind, müssen auch die möglichen Beziehungen zwischen den einzelnen Configuration Items spezifiziert werden. Erst mit den Beziehungen bietet die CMDB einen wesentlichen Vorteil gegenüber einer Inventurdatenbank an, weil nur so die Zusammenhänge dokumentiert werden können.

Weil nicht alle Beziehungen die gleiche semantische Bedeutung besitzen, müssen alle Beziehungen einem Typ besitzen. Dieser Typ soll dann die Bedeutung der Beziehung beschreiben und verdeutlichen. Folgende Beziehungstypen können in einer CMDB sinnvoll sein (in Anlehnung an [BKP 02]):

- Physische Beziehung
  - Ist Bestandteil von
  - Ist verbunden mit
- Logische Beziehung
  - Ist eine Kopie von
  - Bezieht sich auf
  - Wird verwendet von
  - Ist erforderlich für

Modellierung Bei der Modellierung der Configuration Items und ihren Beziehungen stellt sich die Frage, mit welchem Detailierungsgrad diese Strukturen modelliert werden sollen (siehe dazu auch die Kritik im Abschnitt 2.4.1 auf Seite 10). Je detailierter das Modell ist, desto spezifischere Antworten können auf die Anfragen an das Configuration Management gegebene werden. Aber damit steigt auch der Arbeitsaufwand für die Eingabe und für die Aktualisierungen der Daten. Diese Entscheidung fällt spätestens bei der Formulierung der Schemas.

Desweiteren entwickelt sich auch die IT-Infrastruktur ständig weiter, und damit muß auch das Modell sich ständig an die neue Situation anpassen. Deshalb ist eine regelmäßige Überprüfung der gewählten Strukturen notwendig und gegebenenfalls muß dann eine neue Identifizierung durchgeführt werden.

Die Informationen über die Configuration Items werden für die Unterstützung der anderen Prozesse gesammelt, damit diese Prozesse nicht selber diese Informationen verwalten müssen. Und wenn die anderen Prozesse sich weiterentwickeln, dann können sich auch die Anfragen an das Configuration Management ändern. Und diese Änderungen können wieder eine Anpassung der Schemas erfordern.

Weil bei der Identifizierung viele Faktoren berücksichtigt werden müssen und die resultierende Schemas den Nutzen des Configuration Management erheblich beeinflussen können, werden in einer anderen Diplomarbeit (siehe dazu [Sage 05]) Strategien und Konzepte für die Erstellung einer CMDB erarbeitet.

Ausgangskonfiguration Nachdem das Modell für die CMDB erstellt worden ist, muß die CMDB mit den Daten gefüllt werden. Dieser Vorgang entspricht einer Inventur. Damit die eigebene Ausgangskonfiguration vollständig und in sich konsistent ist, soll die Eingabe aller Configuration Items an einem bestimmten Stichtag durchgeführt werden. Falls die Eingabe länger als ein Tag dauert, dann müssen die Daten der Configuration Items jeweils vom gewählten Stichtag eingegeben werden.

Wenn die Daten der Configuration Items schon vorhanden sind, können natürlich diese Daten in die Ausgangskonfiguration übernommen werden. Jedoch soll dieser Import von Configuration Items nur dann durchgeführt werden, falls damit nicht die Datenkonsistenz der CMDB gefährdet wird.

Bei der Eingabe der Configuration Items erstellt die CMDB automatisch eindeutige Identfikationsnummern, welche dann auf den Gegenständen vermerkt werden sollen. Dann kann auch später sehr einfach zu jedem Gegenstand das passende Configuration Item in der CMDB gefunden werden.

Kontrolle Alle Änderungen an der IT-Infrastruktur erfordern analoge Änderungen in der CMDB. Zu diesen Änderungen gehören die Anschaffung neuer Betriebsmittel.

Für eine einfachere Adminstration soll IT-Infrastruktur so homogen wie möglich sein, damit die Infrastruktur überschaubar bleibt. Diese Homogenität kann dadurch erreicht werden, daß alle Configuration Items der DSL bzw. DHS des Release Managements entsprechen (siehe dazu den Abschnitt 2.3.6 auf Seite 10). Diese Vorgaben müssen aber auch regelmäßig überprüft werden und diese findet innerhalb der Kontrolle statt.

Statusüberwachung Jedes Configuration Item besitzt einen Status, der den aktuellen Zustand beschreibt. Weil der Zustand über den Einsatz eines Configuration Items entscheiden kann, muß der Status jedes Configuration Items überwacht werden.

Folgende Werte können dabei als Status sinnvoll sein (entnommen aus [BKP 02]):

- Geplant / bestellt
- Erhalten / vorrätig
- Getestet
- Implementiert
- Im Einsatz
- Nicht im Einsatz
- In Wartung
- Archiviert

Für eine mögliche Diagnose kann es interessant sein, wie der Verlauf des Status eines Configuration Items war. Aus diesem Grund sollen die historischen Werte des Status archiviert bleiben. Zum Beispiel kann man dann die Anzahl der Reparaturen aus den historischen Werten ablesen und auf die Stabilität des Configuration Items schliessen.

Verifizierung Damit die CMDB auch immer auf dem aktuellen Stand ist, muß ihr Inhalt regelmäßig über eine Verifizierung überprüft werden. Dazu wird zu einem festgelegten Stichtag überprüft, ob die Beschreibung der Configuration Items die entsprechenden Betriebsmitteln korrekt wiedergeben. Falls dabei Unterschiede festgestellt werden, dann werden die notwendigen Änderung an der CMDB in einer Verifizierungs-RfC festgehalten.

4.4.4 Zusammenarbeit

Das Configuration Management arbeitet für die Übernahme der Änderungen in die CMDB sehr eng mit dem Change Management zusammen. Zusätzlich bietet das Configuration Management seine Informationen allen anderen Prozessen an und deshalb benötigt das Configuration Management auch eine einheitliche Schnittstelle zu diesen Prozessen.

Change Management Das Configuration Management arbeitet sehr eng mit dem Change Management zusammen, weil alle Änderungen der CMDB durch passende RfCs beschrieben werden müssen. Dazu benötigt das Change Management sehr gute Kenntnisse über den Aufbau der CMDB und ihrer Strukturen, damit die notwendigen Änderungen für eine RfC an den Configuration Items in der CMDB auch effizient umgesetzt werden können.

Change Management	
Ziel:	Korrekte Durchführung von Änderungen an der IT-Infrastruktur
Qualitätsparameter:	weniger Regressionen nach einer Änderungen
Leistungsindikatoren:	Anzahl und Umfang der durchgeführten Änderungen
Input:	Änderungsanträge in Form von RfCs
Output:	Ergebniss der Änderung; Dokumentation der Änderung; Aktualisierung der CMDB
Resourcen:	Informationen über die Configuration Items aus der CMDB
Rollen:	Änderungsbeirat, Änderungsantragsteller, Änderungsprüfer, Änderungskoordinator

Tabelle 4.4: Prozessübersicht: Change Management

Andere Service Prozesse Das Configuration Management bietet allen anderen Prozesse die Informationen über die gewünschten Configuration Items an. Damit die Prozesse auch die richtigen Configuration Items finden können, muß das Configuration Management auch eine mächtige und gleichzeitig komfortable Suchfunktionen anbieten.

4.5 Change Management

Die Nutzung einer modernen IT-Infrastruktur benötigt eine ständige Wartung und häufige Anpassungen an neue Gegebenheiten, so daß die auftretende Menge an notwendigen Änderungen einerseits sehr schnell durchgeführt werden müssen, aber andererseits müssen bei einer Änderung die negativen Nebenwirkungen vermieden werden. Aus diesem Grund soll das Change Management die Durchführung aller Änderungen überwachen, damit alle Änderungen die selben Qualitätsanforderungen erfüllen.

4.5.1 Ziel

Das Ziel des Change Management ist die kontrollierte Umsetzung aller Änderungen, so daß die gesetzten Qualitätsanforderungen auch eingehalten werden.

Für die Durchführung einer Änderung erhält das Change Management einen Änderungsantrag in Form eines “Request for Change” (kurz: RfC). Der Prozess muß dann die eingehenden RfCs mit den vereinbarten Qualitätsanforderungen so effizient wie möglich umsetzen. Zusätzlich soll auch die Zeit für die Umsetzung der RfCs minimiert werden, so daß den anderen Prozessen keine höheren Kosten entstehen.

Jedoch dürfen die RfCs keine regelmäßige oder vorhersagbaren Änderungen beschreiben, weil dann diese Änderung ein Teil des laufenden Betriebes einer IT-Infrastruktur ist. Darunter fallen auch das regelmäßige Erstellen von Backups oder das Einspielen von Updates.

4.5.2 Rollen

Eine besondere Rolle in den ITIL Prozessen ist der Änderungsbeirat, weil der aus den Experten der verschiedenen Fachbereichen zusammengesetzt ist. Damit stellt der Änderungsbeirat die notwendigen Kompetenzen für die sichere Durchführung von Änderung zur Verfügung. Die restlichen Rollen besitzen sehr viele Gemeinsamkeiten mit den Rollen aus den anderen Prozessen.

Änderungsbeirat Die Aufgabe des Änderungsbeirat ist die Unterstützung des Change Managements durch die Beratung in allen Fragen von Experten. Aus diesem Grund soll der Änderungsbeirat über alle wichtigen Aktivitäten des Change Managements informiert sein. In der englischen ITIL Dokumentation heißt der Änderungsbeirat "Change Advisory Board" (kurz: CAB).

Änderungsantragsteller Der Änderungsantragsteller beschreibt alle Personen, die eine RfC dem Change Management übergeben, damit dieser die Durchführung überwachen kann. Zu den Aufgaben des Änderungsantragsteller gehört auch die Erstellung einer RfC. Damit die RfC ohne Problem vom Change Management bearbeitet werden kann, wird sie am Besten vom Änderungsantragsteller gemeinsam mit dem Change Management erstellt.

Änderungsprüfer Bevor eine Änderung durch das Change Management umgesetzt werden kann, muß jede Änderung vom Änderungsprüfer genehmigt werden.

Änderungskoordinator Damit die Realisierung einer Änderung reibungslos abläuft, muß der Änderungskoordinator die Abläufe schon im voraus planen. Und während der Implementierung einer Änderung koordiniert und überwacht er dann alle Vorgänge.

4.5.3 Aktivitäten

Die Abbildung 4.4 zeigt die Übersicht über die Aktivitäten des Change Managements. Wie in dieser Abbildung zu erkennen ist, ist der Input für das Change Management die RfCs, die von allen anderen Prozessen erstellt werden können. Alle Aktivitäten im Change Management besitzen dann nur noch das Ziel, diese RfCs so schnell und sicher wie möglich umzusetzen.

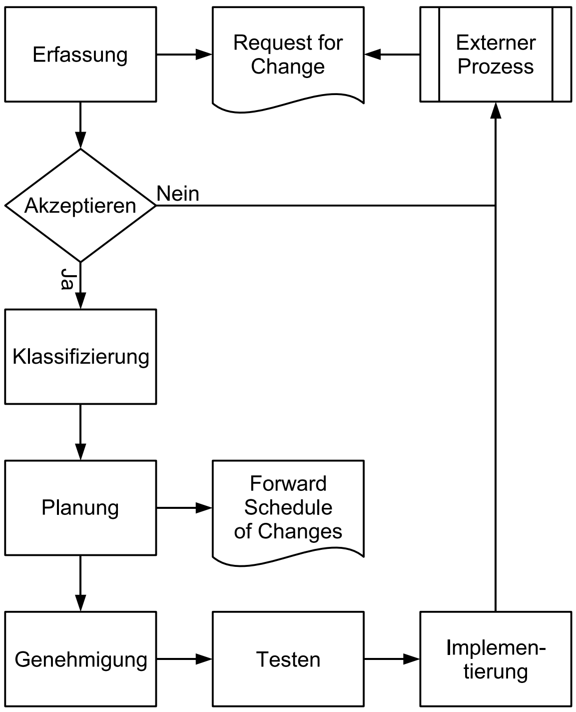

Abbildung 4.4: Übersicht über die Aktivitäten des Change Managements

Erfassen Bei einer Übermittlung einer RfC erhält das Change Management den Auftrag, die beschriebene Änderung zu planen und zu überwachen. Dazu muß zuerst der Prozess, der eine Änderung wünscht, eine korrekte RfC verfassen, welche seine Änderung vollständig beschreibt.

Die ausführliche Beschreibung einer Änderung in der RfC ist für die gute Planung und Überwachung notwendig. Zusätzlich benötigt die RfC auch alle Informationen für die notwendigen Änderung in der CMDB.

Idealerweise soll jede Änderung auch reversibel sein, damit bei Problemen die Änderung wieder rückgängig gemacht werden kann. Diese Informationen müssen ebenfalls in der RfC dokumentiert werden.

Für eine RfC sind deshalb folgende Informationen notwendig:

- Beschreibung aller Änderungen an den betroffenen CIs
- Begründung für die RfC
- Priorität
- Notwendige Resourcen
- Abhängigkeit von anderen RfCs
- Bedingungen während der Umsetzung (Ausfallzeit?)
- Herkunft der RfC
- Termin, bis wann die Änderung umgesetzt worden ist

Akzeptieren Da die Herkunft der RfCs von anderen Prozessen Managements hereinkommen, muß jede RfC erst vom Change Management akzeptiert werden. Die Grundlagen für eine Entscheidung, ob eine RfC akzeptiert wird, sind:

- Korrekter Aufbau der RfC
- Vollständigkeit
- Korrekte Angaben
- Autorisierung für die RfC
- Realisierbarkeit

Ein Akzeptieren einer RfC entscheidet noch nicht, ob und wann die RfC umgesetzt wird, sondern nur die RfC die Voraussetzungen für die weitere Bearbeitung erfüllt.

Klassifizieren Beim Klassifizieren eine RfC werden folgende Fragen beantwortet: In welchen Bereich fällt die Änderung, wie dringend und wie groß ist die Auswirkung der Änderung. Diese Klassifizierung entscheidet über die Reihenfolge der Umsetzung und den notwendigen Vorbereitungen.

Da diese Klassifizierung sehr wichtig für die spätere Umsetzung ist, soll diese Klassifizierung mit dem Antragsteller gemeinsam erarbeitet werden. Dabei kann der Antragsteller vorallem seine Vorstellungen einbringen und er kann offene Fragen beantworten.

Planen Damit das Change Management auch mehrere RfCs gleichzeitig bearbeiten kann, muß die Bearbeitung der RfCs zeitlich geplant werden. Das Ergebnis dieser Planung wird deshalb in den Änderungskalender eingetragen, der in ITIL als “Forward Schedule of Changes” (kurz FSC) genannt wird.

Da der FSC eine zentrale Informationsquelle für das Change Management ist, soll der CAB bei der Erstellung des FSC in beratender Form involviert sein. Durch diese Beratung sollen Konflikte innerhalb des FSC schneller endeckt und auch beseitigt werden können.

Die aktuelle Version des FSC soll allen Mitarbeitern des Unternehmens zur Verfügung stehen, damit die vorgenommen Änderungen für die Betroffenen nicht unerwartet auftreten und sich alle besser darauf vorbereiten können.

Genehmigung Für die Durchführung von größeren Änderungen benötigt man Resourcen, welche schon vor der Implementierung der Änderung zur Verfügung stehen müssen. Und damit diese notwendigen Resourcen auch beschafft werden können, benötigt man häufig die Genehmigungen von den entsprechenden Stellen.

Aus diesem Grund ist die Einholung aller notwendigen Genehmigungen ein fester Bestandteil des Change Management Prozesses. Dabei können folgende Arten von Genehmigungen notwendig sein:

- Finanzielle Genehmigung
- Technische Genehmigung
- Geschäftliche Genehmigung

Bei der finanziellen Genehmigung soll eine Kosten-Nutzen-Analyse durchgeführt werden, in der geprüft wird, ob der mögliche Nutzen die notwendigen Kosten rechtfertigt. Dies erfordert mindestens die Zusammenstellung aller anfallenden Kosten für diese Änderung. In dieser Kostenkalkulation soll auch die Arbeitszeit der Mitarbeiter eingerechnet werden.

Bei der technischen Genehmigung soll vorallem die Auswirkung auf die anderen technischen Systeme untersucht werden. Ein Teilaspekt der Auswirkungen ist die Frage, ob die Änderung wirklich erforderlich ist. Damit auch die Änderung erfolgreich umgesetzt werden kann, soll auch überprüft werden, ob die Änderung überhaupt realisierbar ist, weil der Abbruch einer Änderung während der Implementierung sehr hohe Kosten erzeugt.

Bei der geschäftlichen Genehmigung soll dann geprüft werden, welche Auswirkung die Änderung auf die ganze Organisation besitzt. Idealerweise kann durch eine Änderung die Produktivität gesteigert werden.

Testen Um Probleme nicht erst bei der Implementierung festzustellen, soll jede Änderung ausführlich getestet werden. Dazu müssen zuerst die durchzuführende Tests mit den zu erwartenden Ergebnissen festgelegt werden, bevor diese dann auch durchgeführt werden. Die Ergebnisse jedes Tests werden dann analysiert, um gegebenfalls notwendige Anpassungen an der Änderung vorzunehmen.

Implementierung Nachdem die Änderung ausführlich getestet worden ist, kann die Änderung nach Plan umgesetzt werden. Bei dieser Implementierung kontrolliert, koordiniert überwacht das Change Management alle Arbeiten bei der Durchführung der Änderung und überlässt die Durchführung den Spezialisten.

4.5.4 Zusammenarbeit

Das Change Management arbeitet sehr eng mit Configuration Management zusammen, damit die CMDB immer auf dem aktuellen Stand bleiben. Für die Durchführung der Änderungen benötigt das Change Management die RfC von allen anderen Prozessen. Und zusätzlich erfordert jede Änderung auch eine Anpassung in der CMDB.

Release Management	
Ziel:	kontrollierter Roll-Out von neuen Releases und die Verwaltung der DSL und DHS
Qualitätsparameter:	fehlerfreie Releases
Leistungsindikatoren:	Einhaltung der DSL und DHS
Input:	Anforderungen an die IT-Infrastruktur
Output:	DSL und DHS
Resourcen:	CMDB und Richtlinien für die Releases
Rollen:	Releaseersteller, Releaseabnehmer, Releasekoordinator

Tabelle 4.5: Prozessübersicht: Release Management

Configuration Management Das Change Management arbeitet besonders eng mit dem Configuration Management zusammen, weil jede Änderung auch in der CMDB dokumentiert werden muß. Und deshalb benötigt das Change Management genaue Details über die Strukturen in der CMDB, damit die Änderungen in der CMDB automatisiert werden können.

Andere Service Prozesse Da viele Prozesse eine Änderung des aktuellen Zustands als Ziel besitzen, müssen diese Änderungen als RfC formuliert werden, damit das Change Management die Änderung kontrolliert durchführen kann.

4.6 Release Management

Eine Vereinheitlichung der eingesetzten Hard- und Software ermöglicht eine effizientere Administration der IT-Infrastruktur, weil damit auch die Komplexität erheblich vereinfacht werden kann. Diese homogene Strukturen können aber nur erreicht werden, wenn entsprechende Vorgaben existieren und die auch unternehmensweit durchgesetzt werden. Diese Vorgaben für die Hard- und Software heißen in ITIL DSL und DHS. Und das Release Management besitzt die Aufgabe, die DSL und DHS zu verwalten und durchzusetzen.

4.6.1 Ziel

Das Ziel des Release Management ist eine homogene IT-Infrastruktur, welche den Vorgaben aus der DSL und aus der DHS entsprechen. Und damit die DSL und der DHS die Bedürfnisse der Organisation befriedigen können, müssen die beiden regelmäßig über neue Releases an die neue Situation angepasst werden.

Weil die Auswirkung eines Releases potentiell die ganze Organisation betreffen, müssen alle Releases sehr sorgfältig getestet werden, damit keine Probleme nach dem Roll-Out auftreten.

Und nach dem Roll-Out muß die Einhaltung der DHS bzw. DSL kontrolliert werden. Die Überprüfung findet in enger Zusammenarbeit mit dem Configuration Management statt.

4.6.2 Rollen

Die Rollen des Release Managements besitzen Ähnlichkeiten mit den Rollen des Change Managements. Jedoch fehlen die Rollen für die Implementierung, weil diese Tätigkeit in den Kompetenzbereich des Change Managements fällt.

Releaseersteller Die Rolle entwirft und stellt neue Releases zusammen für die Weiterentwicklung der DSL bzw. DHS.

Releaseabnehmer Bevor ein Release umgesetzt werden kann, muß der Releaseabnehmer es genehmigen. Zu seinem Aufgabenbereich gehört auch die Durchführung der notwendigen Tests.

Releasekoordinator Der Releasekoordinator plant und koordiniert die Umsetzung eines Releases.

4.6.3 Aktivitäten

Die hier beschriebenen Tätigkeiten beschäftigen sich mit den Erstellung und Auslieferung von Releases. Dabei muß jedes Release den bei der Einführung des Prozesses festgelegten Richtlinien und Qualitätsanforderungen erfüllen.

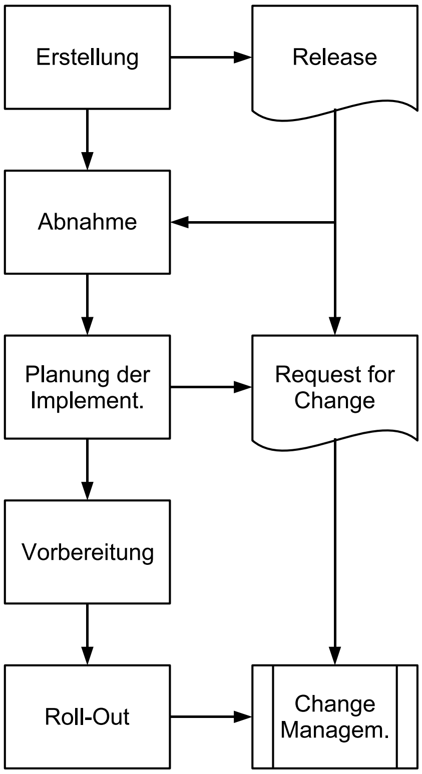

Abbildung 4.5: Übersicht über die Aktivitäten des Release Managements

Erstellung Jedes neue Release muß erst zusammengestellt und dokumentiert werden, bevor es weiter vom Release Management bearbeitet werden kann.

Die genaue Erstellung eines Releases unterscheidet sich für Hardware und Software. Wenn neue Hardware in die Organisation eingeführt werden soll, muß das Release zuerst die neue Hardware genau beschreiben und spezifizieren, damit später keine Verwechslungsgefahr besteht. Zusätzlich muß auch eine Vorgabe für die Konfiguration der Hardware erstellt werden, wenn die betreffende Hardware verschiedene Einstellungsmöglichkeiten besitzt. Diese Konfigurationsvorgabe soll vorallem Probleme durch unterschiedliche

Einstellungen vermeiden. Falls die Konfiguration auch nach dem Roll-Out geändert werden muß, müssen die möglichen Einstellungen und ihre Auswirkungen in die Releasedokumentation aufgenommen werden. Zu dieser Dokumentation gehört auch immer die Konfigurationsvorgabe, weil nur so später die Einhaltung des DHS überprüft werden kann. In die Dokumentation gehören auch die Spezifikationen der Hardware und eine Bedienungsanleitung für die Administratoren und eine für die Mitarbeiter.

Ein neues Softwarerelease muß zuerst zusammengestellt werden. Zuerst muß das Programm genauer spezifiziert werden, in dem die genaue Version festgelegt wird, welche mit dem Release eingeführt werden soll. Dann müssen die notwendigen Softwarelizenzen erworben werden, damit keine illigale Software in der Organisation benutzt werden. Und nachdem die Software korrekt lizenziert ist, muß die Software auf einen passenden Datenträger verfügbar sein, damit später beim Roll-Out die Software auf allen Rechnern eingespielt werden kann.

Jedoch besitzen die Releases für Hardware und Software auch Gemeinsamkeiten. Zuerst unterscheidet man zwischen einem Major und einem Minor Release: Wenn das Release größere Änderungen beinhaltet oder wenn sich die Funktionen der Hardware oder der Software erheblich verändert haben, dann ist dies ein Major Release. Ein Minor Release hingegen behebt nur Fehler oder schliesst Sicherheitslücken und besitzt sonst keine weiteren größeren Änderungen. Jedoch basiert jedes Minor Release auf ein vorhergehendes Major Release und übernimmt dabei alle Eigenschaften des Major Release mit der Ausnahme der behobenen Fehler. Deshalb kann ein Minor Releases viel einfacher und schneller als ein Major Release bearbeitet werden, weil viele Ergebnisse des zu Grunde liegende Major Release für das Minor Release übernommen werden kann.

Die Unterscheidung zwischen einem Major und einem Minor Release wird auch in der Versionsnummer jedes Releases verdeutlicht. Jede Versionsnummer besteht aus einer Major- und einer Minornummer. Diese beiden Teile werden über einen Punkt getrennt. Anhand von Beispielen soll dieses Konzept verdeutlicht werden:

- **1.0** zeigt die Einführung eines Major Releases.
- **1.1** ist ein Minor Release, welches auf dem Major Release 1.0 basiert.
- **1.2** ist ein weiteres Minor Release, welches ebenfalls auf dem Major Release 1.0 basiert.

Jedes Release muß immer eine neue und eindeutige Versionsnummer erhalten, damit die unterschiedlichen Releases auch sicher unterschieden werden können.

Zusätzlich muß in der Releasebeschreibung auch geklärt werden, ob die neue Version alle alte Versionen ersetzt, oder ob auch die alten Versionen noch weiterhin eingesetzt werden dürfen. Diese Information ist für die Aktualisierung der DSL und DHS wichtig.

Abnahme Um die Qualität eines Releases zu garantieren, muß jedes Release ausführlich überprüft werden. Jedoch für ein Minor-Release kann der Testumfang kleiner ausfallen, weil nur die überschaubare Anzahl der Änderungen überprüft werden muß.

Bei einem Major-Release müssen aber alle Eigenschaften auf Herz und Nieren geprüft werden. Dann kann ein Testprogramm wie folgt aussehene:

- korrekte Releasedokumentation
- Fehlerfreier Ablauf
- Praxisnahe Tests mit Produktionsdaten
- Benutzerfreundlichkeit
- Sicherheit gegenüber Angreifern
- Bestimmung der maximale Last
- Zusammenarbeit mit anderen Programmen bzw. Komponenten

Die durchgeführten Tests und die in der Testumgebung gewonnenen Ergebnisse dokumentiert der Releaseabnehmer in einem Abnahmedokument.

Falls das Release alle Tests erfolgreich bestanden hat, gibt der Releaseabnehmer das Release frei. Andernfalls muß entschieden werden, ob der nächste Test komplett oder nur teilweise wiederholt werden muß. Jedenfalls muß jedes Release die festgelegten Tests vollständig erfüllen, bevor die nächsten Schritte im Release Management unternommen werden.

Planung der Implementierung Nachdem das Release abgenommen und freigegeben worden ist, erfolgt die Planung des Roll-Outs des Releases. In diesem Plan ist der genaue Ablauf und ein Zeitplan der Implementierung des Releases mit den dazu erforderlichen Resourcen festgelegt. Für einen reibungslosen Ablauf müssen alle Resourcen vor dem Beginn des Roll-Outs vorhanden sein, damit keine unnötige Verzögerungen auftreten können. Die Planung soll wenn möglich die Ausfallszeiten für die Organisation minimieren und es soll auch genügend Zeit vorhanden sein, um auch auf unerwartete Situationen reagieren zu können.

Ein wichtiger Punkt in der Planung ist auch die Beschreibung für ein Backout des Releases, falls während oder nach dem Rollout ein Problem auftaucht, welches die Nutzung des Releases unmöglich macht. Die Situation ist sehr unwahrscheinlich, aber die Vorsichtsmaßnahme ist trotzdem notwendig, damit der weitere Betrieb auch nach dem Rollout gewährleistet werden kann. Ein wesentlicher Bestandteil jedes Backoutplanes ist auch die vorherige Datensicherung, damit auf keinem Fall wichtige Betriebsdaten verloren gehen können.

Vorbereitung Mit einem neuen Release verändern sich auch die Arbeitsumgebung und eventuell auch die Arbeitsabläufe der betroffenen Mitarbeiter. Da eine Umstellung zu einer kurzzeitigen Leistungsminderung führen kann, soll geprüft werden, welche Mitarbeiter eine Schulung für dieses neue Release benötigen. Zusätzlich muß die Releasedokumentation allen Mitarbeitern im voraus zur Verfügung gestellt werden, damit sie sich darauf vorbereiten können.

Roll-Out Alle vorhergenden Aktivtäten haben das Ziel, daß Roll-Out des Releases so einfach und problemlos wie möglich zu gestalten.

Ein wesentlicher Unterschied beim Roll-Out existiert zwischen Hardware und Software. Bei Hardware ist eine wesentliche Aufgabe des Roll-Outs die Logistik, d.h. die Auslieferung und Installation der neuen Hardware. Hingegen soll das Ziel für ein Softwarerelease immer eine vollständig automatische Verteilung und Installation der Software sein. Damit können dann später auch kleinere Softwareupdates sehr schnell auf allen Geräten installiert werden.

Nach der Installation muß das Release übeprüft werden, ob einerseits die Installation vollständig war und andererseits keine Fehler aufgetreten sind. Danach müssen die notwendigen Änderungen in der DSL bzw. DHS und an der CMDB durchgeführt werden.

4.6.4 Zusammenarbeit

Das Release Management beschränkt sich auf die Zusammenarbeit auf das Configuration und Change Management. Besonders das Configuration Management ist von der Arbeit des Release Managements betroffen, weil einerseits die DSL und DHS mit den Daten aus der CMDB überprüft werden können und andererseits jedes Roll-Out viele Änderungen ebenfalls in der CMDB benötigt.

Configuration Management Wie auch die anderen Prozesse nutzt das Release Management die Dienste des Configuration Managements als wichtige Informationsquelle. Jedoch überwacht das Release Management auch die Einhaltung der DSL und des DHS über die CMDB und fällt damit aus dem Rahmen in Bezug auf das Nutzungsverhalten der CMDB.

Change Management Für die Aufgabe des Roll-Outs muß das Release Management sehr eng mit dem Change Management zusammenarbeiten, weil die Durchführung von Änderungen die Hauptaufgabe des Change Managements ist und bleiben soll.

4.7 Zusammenfassung

In diesem Kapitel sind die ITIL Service Support Prozesse im Detail beschrieben worden. Jedoch stellt diese Beschreibung nicht die einzige mögliche Variante dar, sondern versucht auch die besonderen Bedürfnisse des Szenarios zu erfüllen (siehe das Kapitel 3 auf Seite 14). Dabei bleiben die Prozesse allgemein genug, damit sie auch in anderen Organisationen oder Unternehmen eingeführt werden können.

Diese Kapitel bildet ab jetzt die Grundlage für die weitere Entwicklung des Toolkonzeptes. Dazu werden im nächsten Kapitel die Anforderungen an die Programme erarbeitet, welche dann die hier vorgestellten Prozesse unterstützen können.

5 Anforderungen

Aus den ITIL-Service-Support-Prozessen, die im Kapitel 4 auf Seite 32 beschrieben sind, ergeben sich eine Menge von möglichen Ansätzen zur Softwareunterstützung. Diese Unterstützung per Software ermöglicht eine teilweise erhebliche Effizienzsteigerung der Tätigkeiten, weil die EDV sehr viel besser einen hohen Informationsfluß verarbeiten kann. Damit sind alle Tätigkeiten, in denen Daten archiviert und bereitgestellt werden, besonders gut für eine Softwareunterstützung geeignet.

Aus diesem Grund soll die Softwareunterstützung der ITIL Prozesse genauer spezifiziert werden, so daß entweder eine passende Software ausgewählt oder selbst entwickelt werden kann. In diesem Kapitel werden dazu die notwendigen Anforderungen beschrieben.

5.1 Einführung

Da die Anforderungsanalyse am Anfang einer langen Entwicklung steht, ist es wichtig, diese Analyse korrekt und umfassend zu erstellen. Der Grund für die Wichtigkeit einer guten Anforderungsanalyse ergeben sich bei notwendigen Änderungen der Anforderungen. Wenn diese Änderungen früh in der Entwicklungsphase durchgeführt werden, entstehen auch nur geringe Kosten. Aber je später eine Änderung noch durchgeführt werden muß, desto teurer wird diese, weil die Folgen erheblich größer sind.

Ziele der Anforderung Das Ziel dieser Arbeit ist die Entwicklung eines Toolkonzept für die Unterstützung der ITIL Prozesse. Jedoch muß dieses Ziel genauer spezifiziert werden, damit auch eine passende Lösung erarbeitet werden kann. Die Anforderungen ergeben sich einerseits aus den ITIL Prozessen (siehe das Kapitel 4 auf Seite 32) und andererseits aus dem Szenario dieser Arbeit (siehe das Kapitel 3 auf Seite 14).

Jedoch können die ITIL Prozesse nicht vollständig von der Software realisiert werden, so daß zuerst die Aufgaben für die Softwareunterstützung bestimmt werden müssen. Eine weitere Herausforderung für die Unterstützung der Prozesse ist die Komplexität, die nicht nur mit einer Software abgedeckt werden kann. Aus diesem Grund müssen die Anforderungen durch den Einsatz von mehreren Programmen erfüllt werden. Somit dienen diese Anforderungen zum einen für die Softwareentwicklung (später im Kaptel 7 auf Seite 82) und auch für die Evaluierung von vorhandenen Softwareprodukten (siehe dazu das Kapitel 6 auf Seite 71)

Jedoch können die Anforderungen auch von der Einsatzumgebung abhängen, so daß eine weitere Quelle für die Anforderungen das Szenario ist (siehe das Kapitel 3 auf Seite 14). Das Ziel der Anforderungen für das Szenario sind vorallem die Milderung der beschriebenen Probleme (siehe dazu den Abschnitt 3.5.2 auf Seite 29).

Aufbau Damit die Software die Prozesse optimal unterstützen können, sind in der ersten Hälfte dieses Kapitels die Anforderungen für die einzelnen Prozesse beschrieben. Jedoch ergeben sich auch Anforderungen für alle Prozesse, welche dann im zweiten Teil beschrieben werden.

Zuerst wird für jede Anforderung die Motivation und die dazugehörigen Gründe beschrieben. Diese Erklärung soll einerseits den Zusammenhang mit den Prozessen herstellen und andererseits auch die Auswirkungen beim Erfüllen einer Anforderung verdeutlichen.

Nach der Erklärung für die Gründe einer Anforderung wird die Anforderung formal spezifiziert. Dazu erhält jede Anforderung einen Namen und eine Nummer, damit später einfacher auf die Anforderungen eingegangen werden kann.

5.2 Aufgaben der Softwareunterstützung aus den ITIL Prozessen

In diesem Abschnitt soll aus der Beschreibung der ITIL Service Support Prozessen (vgl. Kapitel 4 auf Seite 32) die Aufgaben für eine mögliche Softwareunterstützung analysiert werden. Dabei stellt sich immer die Frage, ob eine Aktivität von einem Menschen oder von der Software ausgeführt werden soll.

Menschen sind intelligent und flexibel und können sich dadurch sehr schnell an neue Gegebenheiten anpassen. Aber Menschen unterliegen Schwankungen in der Qualität ihrer Arbeit. Bei sehr eintönigen und sich wiederholende Tätigkeiten produzieren aus diesem Grund Menschen häufig Fehler.

Sobald Tätigkeiten monoton und sich häufig wiederholen lohnt es sich, diese Aktivität zu automatisieren. Im Bereich des IT-Managements kann diese Automatisierung durch den Einsatz von Software geschehen. Dabei soll die Software vorallem alle Informationen archivieren und verarbeiten, so daß der Manager am Ende nur noch die für ihn relevante Informationen erhält.

5.2.1 Incident Management

Im Incident Management muß die Software vorallem alle vorhandenen Informationen über eine Störung zusammenfassen und archivieren. Zu einer Störung gehören deshalb folgende Informationen (entnommen aus dem Abschnitt 4.2.3 auf Seite 34):

- der Name des Support Mitarbeiters
- weitere Namen von Personen, die von der Störung betroffen sind
- genaue Beschreibung der Störung
- genaue Uhrzeit und Datum des ersten Auftretens
- Ablauf zum Reproduzieren der Störung
- Soll- und Istverhalten
- betroffene Geräte (welcher Computer?) und betroffene Software aufzählen
- mögliche Auswirkungen
- Bbehebungsversuche der Störung
- bekannte Workarounds

Anforderung 1 (Beschreibung einer Störung) *Eine Störungsmeldung muß alle Informationen enthalten, um eine Störung nachvollziehen zu können. Zusätzlich muß für eventuelle Nachfragen auch die Namen aller beteiligten Personen in die Meldung aufgenommen werden.*

Nach dem Eintreffen einer Störungsmeldung sieht der Prozess die Klassifizierung vor, in der die Dringlichkeit, das Ausmaß und die Priorität der Störung festgelegt werden (siehe dazu den Abschnitt 4.2.3 auf Seite 35).

Anforderung 2 (Klassifizierung der Störung) *Die Software muß jede Störungsmeldung von einem Benutzer klassifizieren lassen. Mit der Klassifizierung wird die Dringlichkeit, das Ausmaß und die Priorität der Störung festgesetzt.*

Häufig wird eine Störung durch mehrere Störungsmeldungen unabhängig voneinander gemeldet. Damit alle Störungsmeldungen, welche die selbe Störung beschreiben, gemeinsam bearbeitet werden, muß die Software dem Benutzer eine Übersicht über die vorhandenen Störungsmeldungen geben können.

Anforderung 3 (Störungsmeldung suchen) *Die Software muß alle Störungsmeldungen archivieren und dem Benutzer die Suche nach Störungsmeldungen ermöglichen. Die Suche soll bei der Angabe einer Störung auch alle ähnliche Störungsmeldungen finden können.*

Falls mehrere Störungsmeldungen die selbe Störung beschreiben, müssen diese auch gemeinsam bearbeitet werden:

Anforderung 4 (Störungsmeldungen verknüfpen) *Alle Störungsmeldungen, welche die selbe Störung beschreiben, müssen für die gemeinsame Bearbeitung miteinander verknüpft werden.*

Die Behebung einer Störung beginnt mit der Diagnose und endet dann mit der Behebung in Form einer RfC. Die Ergebnisse der Diagnose und die resultierende RfC müssen ebenfalls in der Störungsmeldung vermerkt werden.

Anforderung 5 (Störungsverlauf dokumentieren) *Alle Arbeiten und neue Erkenntnisse einer Störung müssen dokumentiert werden und im Zusammenhang mit der Störungsmeldung gebracht werden.*

Für die Umsetzung der Lösung ist das Change Management zuständig, so daß die Lösung die notwendige RfC für das Change Management enthalten muß (siehe Anforderung 32 auf Seite 62).

Sobald eine Lösung umgesetzt ist, müssen die Betroffenen darüber informiert werden, so daß einerseits die Lösung der Störung bestätigt werden kann und andererseits auch der normale Arbeitsbetrieb wieder aufgenommen werden kann (entspricht der Anforderung 59 auf Seite 69). Zusätzlich sollen die Betroffenen auf Nachfrage jederzeit über den aktuellen Stand der Störung ausführlich und korrekt informiert werden können.

Anforderung 6 (Betroffene einer Störung informieren) *Alle betroffenen und beteiligten Personen einer Störung müssen jederzeit über den aktuellen Stand einer Störung informiert sein.*

Prozessüberwachung Für die Prozessüberwachung des Incident Managements sind die Berichte über die Störungsmeldungen eine sehr gute Informationsquelle, um die Effizienz des Prozesses zu beurteilen. Den Inhalt der Berichte soll der Prozess-Manager wenn möglich frei wählen können. Jedoch ist ein gutes Angebot an festgelegten Berichten eine mögliche Vereinfachung. Dabei können folgende Informationen bezüglich des untersuchten Zeitraumes analysiert werden:

- Anzahl der neuen, offenen und der geschlossenen Störungen
- Dauer zwischen dem ersten Auftreten und der erfolgreicher Lösung
- Ergebnisse der Rückfragen nach der Behebung einer Störung
- Resourcenverbrauch pro Störung und für den ganzen Prozess
- Statistiken über Störungen, die nach verschiedenen Kriterien zusammengefasst sind
- Anzahl der Störungsmeldungen pro CI

Anforderung 7 (Berichte über alle Störungen) *Es müssen frei definierbare oder vorgefertigte Berichte über alle Störungen eines frei wählbaren Zeitraums automatisch erstellt werden können.*

5.2.2 Problem Management

Beim Problem Management steht die optimale Lösung der Probleme im Vordergrund. Um dieses Ziel zu erreichen, ist der Zeitdruck gegenüber dem Incident Management geringer, weil am Ende das Problem langfristig behoben sein soll.

Bei der ersten Tätigkeit, der Identifizierung neuer Probleme, kann aber die Software nur unterstützen, weil die Identifzierung nur mit einer genauen Beobachtung der aktuellen Situation möglich ist. Für diese Analyse benötigt der Problemmelder kompakte Informationen beispielsweise aus dem Incident Management. Diese Aufgabe kann die Software durch eine gute Recherchemöglichkeiten vereinfachen.

Anforderung 8 (Störungsanalyse für neue Probleme) *Die Software soll die Gemeinsamkeiten der vorhandenen Störungsmeldungen analysieren, so daß daraus neue Probleme erkennbar werden. (baut auf der Anforderung 3 auf der vorherigen Seite auf)*

Wenn dann ein Problem identifziert ist, muß dies auch in der Software dokumentiert werden. Dazu gehört mindestens eine ausführliche Beschreibung des Problems, damit auch später immer klar das Problem verstanden wird. Sehr hilfreich sind Angaben zu den Informationsquellen für ein Problem, so daß auch Dritte den Identifzierungsablauf nachvollziehen können.

Am Ende sollen folgende Informationen in einem Problembericht stehen (entnommen aus der Tätigkeit "Erfassung" im Abschnitt 4.3.3 auf Seite 38)

- Allgemeine Beschreibung des Problems
- Beschreibung des aktuellen Zustandes
- Beschreibung des erwünschten Zustandes
- Vorteile nach der Behebung des Problems
- Auswirkungen bei keiner Behandlung des Problems
- Liste der betroffenen Personen
- Quellen für die Problemidentifizierung

Anforderung 9 (Problembeschreibung) *In der Problembeschreibung sollen alle Informationen und die Namen aller betroffenen Personen für das Verständnis eines Problems dokumentiert sein. Zusätzlich müssen alle Quellen, welche zur Problemidentifikation benutzt worden sind, angegeben werden.*

Weil nicht alle Probleme die gleiche Dringlichkeit oder das selbe Ausmaß besitzen, ist eine Klassifizierung der Probleme sinnvoll (siehe dazu den Abschnitt 4.3.3 auf Seite 39).

Anforderung 10 (Problem klassifizieren) *Jedes Problem muß in der Software mindestens einer Kategorie zugeordnet werden können. Und für die weitere Bearbeitung muß die Software die Informationen über die Dringlichkeit, Priorität, das Risikos und den möglichen Nutzen einer Behebung jedes Problems aufnehmen und verwalten.*

Nachdem das Problem vollständig dokumentiert ist, muß eine Person die Verantwortung für die Lösung des Problems übernehmen (siehe dazu den Abschnitt 4.3.3 auf Seite 39). Dieser Person werden dann in der Software erweiterte Rechte im Bezug auf seine Probleme eingeräumt.

Anforderung 11 (Problemzuweisung) *Die Software muß für jedes Problem eine verantwortliche Person kennen, welche dann erweiterte Rechte in der Koordination und der Planung für das weitere Vorgehen bei der Problemlösung erhält.*

Wie auch im Incident Management können neue Informationen hinzugefügt und geändert werden und es soll auch jede Änderung reversibel sein (eine Folgerung ist die Anforderung 50 auf Seite 66). Eine weitere

Ähnlichkeit mit dem Incident Management ist die Verfolgung des aktuellen Status jedes Problems mit Hilfe eines Statusfeldes.

Im Laufe der Untersuchung des Problems werden dann folgende Informationen zu einem Problem hinzugefügt:

- Klassifizierung
- Ausmaß
- Dringlichkeit
- Priorität
- Liste der Bearbeiter
- Ergebnisse der Diagnose
- Lösungsvorschläge
- Lösungswahl mit Begründung
- resultierende RfC
- Ergebnisse der Überprüfung
- Abschlussdatum

Anforderung 12 (Verlauf eines Problems dokumentieren) *Alle Arbeiten und neue Erkenntnisse müssen dokumentiert und im Zusammenhang mit dem Problem gebracht werden.*

Der Punkt "Lösungswahl" (siehe dazu die gleichnamige Tätigkeit im Abschnitt 4.3.3 auf Seite 40) kann dabei zusätzlich von der Software unterstützt werden, in dem sie zuerst alle möglichen Lösungsvorschläge sammelt und dann den Benutzern ein Forum für die Diskussion über die beste Lösung anbietet.

Anforderung 13 (Auswahl der besten Problemlösung) *Die Software muß alle möglichen Lösungen für ein Problem sammeln und dann die Diskussion zur Wahl der besten Lösung unterstützen.*

Prozessüberwachung Bei der Prozessüberwachung soll die Qualität der Arbeit des Problem Managements untersucht werden. Dazu gehört auch eine Kosten und Nutzenrechnung für jedes Problem. Mit dieser Rechnung soll überprüft werden, ob die Lösung eines Problems auch ökonomisch sinnvoll war.

Anforderung 14 (Kontrolle der Problemlösungen) *Die Kosten jedes Problems müssen erfaßt werden, um eine effektive Kostenkontrolle des Prozesses zu ermöglichen. Diese Ergebnisse müssen dann in einem Bericht ausgewertet werden.*

Zusätzlich soll nach der Lösung jedes Problems die Situation weiter verfolgt werden, um eventuelle Nebenwirkungen einer Problemlösung zu erkennen.

Anforderung 15 (Auswirkungen einer Problemlösung überwachen) *Die Auswirkung einer Problemlösung müssen automatisch überwacht werden. Bei neuen Problemen oder neuen Störungen muß der Zusammenhang zum relevanten Problemen hergestellt werden.*

5.2.3 Configuration Management

Für das Configuration Management beschreibt schon ITIL die zentrale Aufgabe der CMDB. In der CMDB sollen alle Informationen der Configuration Items (siehe dazu die Definition 3 auf Seite 9) und ihre Beziehungen untereinander abgelegt werden können. Daraus folgt schon die erste Anforderung für das Configuration Management

Anforderung 16 (Aufgabe der CMDB) *Die CMDB muß alle Informationen über die Configuration Items erfassen und ihre Beziehungen und Abhängigkeiten dokumentieren. Jedes erfaßte Configuration Item muß dann überwacht und kontrolliert werden, so daß die CMDB immer auf dem aktuellen Stand bleibt. Die gesammelten Informationen müssen dann auch für andere Prozesse in komfortabler Weise zur Verfügung gestellt werden.*

Configuration Items Zuerst steht also die Aufgabe für die Software, die Strukturen der CIs in Form von Schemas zu erfassen und zu verarbeiten (sieh dazu die Tätigkeit "Identifizierung" und "CI spezifizieren" im Abschnitt 4.4.3 auf Seite 42).

Anforderung 17 (Festlegung der CI-Schemas) *Die beschreibende Eigenschaften der Configuration Items werden in Schemas abgelegt. Diese Schemas muß der Benutzer frei definieren und auch noch nachträglich ändern können, falls dabei die Datenkonsistenz erhalten bleibt.*

Damit die Schemas auch die Eigenschaften sinnvoll erfassen können, kann ein Schema als eine Liste von Attributen aufgefasst werden. Jedes dieser Attribute besitzt dann einen Namen und einen Datentyp, damit die eingegebenen Werte auch überprüfbar sind. Jedoch besitzen alle Configuration Items auch eine Menge von gemeinsamen Eigenschaften:

- Namen
- eindeutige Identifikation
- Status
- Klassifizierung
- Eigentümer
- Hersteller

Anforderung 18 (Beschreibung eines CI) *Ein Configuration Item besitzt eine definierte Anzahl von typisierten Attributen.*

Nachdem die Schemas festgelegt worden sind, müssen alle Configuration Items in die CMDB eingegeben werden (siehe dazu den Abschnitt 4.4.3 auf Seite 43):

Anforderung 19 (Eingabe der CI Informationen) *Das Programm muß die Eingabe der Configuration Items anbieten. Dabei muß jedes Configuration Item einem Schema gehorchen.*

Damit die Configuration Items jederzeit eindeutig identifizierbar sind, benötigt jedes Configuration Item eine Identifikationsnummer. Diese Nummer kann entweder eine ausgewählte Eigenschaft oder eine selbst definierte Seriennummer sein (dies ist eine Vorausetzung für die Anforderung 51 auf Seite 66).

Anforderung 20 (CI-Identifizierung) *Jedes Configuration Item muß eindeutig identifizierbar sein.*

Beim Status soll der aktuelle Zustand des CIs beschrieben werden (siehe dazu die Tätigkeit "Statusüberwachung" im Abschnitt). Dieser Zustand kann beispielsweise ausdrücken, ob das Configuration Item neu angeschafft worden ist oder in produktiven Betrieb oder defekt ist. Zusätzlich muß bei jeder Änderung des Status das Änderungsdatum mitgespeichert werden, damit der Zeitraum des aktuellen Status immer erkennbar ist. Desweiteren müssen die historischen Werte mit dem dazugehörigen Datum für jedes Configuration Item gespeichert werden, weil aus diesen Werten kann die Qualität des Configuration Items abgelesen werden.

Anforderung 21 (Statusverfolgung der CIs) *Die aktuelle und alle alten Werte des Status eines Configuration Items müssen vollständig archiviert werden.*

In der Klassifizierung sollen Configuration Items mit ähnlichen Eigenschaften zusammengefasst werden, weil nur so die möglicherweise sehr große Anzahl von Configuration Items handhabbar ist. Folgende Klassifizierung ist eine sinnvolle Möglichkeit:

- Software
- Hardware
- Konfiguration
- Dokumentation
- Policy

Anforderung 22 (CI Klassifizierung) *Alle Configuration Items werden in frei wählbare Klassen zusammengefasst (siehe auch die Anforderung 27 auf der nächsten Seite).*

Beziehungen Bis jetzt beschreibt die CMDB nur die Eigenschaften einzelner Configuration Items, aber nicht ihre Beziehungen. Aus diesem Grund fordert auch die ITIL Beschreibung des Configuration Managements die Aufzeichnung der unterschiedlichen Beziehungen zwischen den Configuration Items.

Anforderung 23 (Eingabe der Beziehungen zwischen CI) *Das Programm muß dem Benutzer die Eingabe der Beziehungen zwischen den Configuration Items ermöglichen. Die Bedeutung jeder Beziehung muß ebenfalls dokumentiert sein.*

Folgende Kategorisierung der Beziehungen sind möglich (entnommen aus Abschnitt 4.4.3 auf Seite 43):

- Physische Beziehung
  - Ist Bestandteil von
  - Ist verbunden mit
- Logische Beziehung
  - Ist eine Kopie von
  - Bezieht sich auf
  - Wird verwendet von
  - Ist erforderlich für

Anforderung 24 (Typisierte Beziehungen) *Jede Beziehung zwischen zwei Configuration Items besitzt einen Typ. Dieser Typ soll die Bedeutung und den Grund für diese Beziehung beschreiben.*

Die Anforderung 24 für typisierte Beziehungen ist notwendig, weil damit nicht sinnvolle Beziehungen von der Software abgelehnt werden können. Und damit sichert diese Anforderung die Datenkonsistenz in der CMDB und die Verifizierung (siehe Abschnitt 4.4.3 auf Seite 44) der CMDB.

Suche von Configuration Items Die eingegebenen Informationen über die Configuration Items werden für die anderen Prozesse bereitgestellt. Jedoch kennen die Prozesse selten die Identifikationsnummer der gewünschten Configuration Items (siehe die Anforderung 20 auf der vorherigen Seite). Deshalb muß die CMDB eine komfortable Suche anbieten, so daß der Nutzen des Configuration Management für die anderen Prozesse überhaupt realisierbar ist (siehe dazu Abschnitt 4.4.1 auf Seite 41).

Anforderung 25 (Suche von Configuration Items) *Alle Configuration Items in der CMDB müssen über eine Suchfunktion gefunden werden. Dabei muß der Benutzer nach allen Eigenschaften eines Configuration Items bzw. der Beziehung zwischen den Configuration Items suchen können. Das Ergebnis ist eine Liste von Configuration Items, welche das Suchkriterium erfüllen.*

Bildung der transitiven Hülle Eine weitere Verbesserung der Suchanfragen kann die Bildung einer transitiven Hülle sein. Dazu benötigt man die Beziehungen zwischen den Configuration Items. Mit der transitiven Hülle über die Configuration Items, welche mit einer angegebenen Beziehung zueinander stehen, können dann auch sehr komplexe Fragen beantwortet werden.

Beispielsweise kann in der CMDB die Abhängigkeiten zwischen den CIs abgelegt sein. Dann ist eine sehr häufige Frage, welche Auswirkung hat der Ausfall oder die Entfernung einer CI. Diese Frage kann dann durch die transitivie Hülle über die Abhängigkeiten und ausgehend von der einen CI beantwortet werden.

Die Bildung der transitive Hülle ist für die Diagnose im Incident (siehe Abschnitt 4.2.3 auf Seite 36) und im Problem Management (siehe Abschnitt 4.3.3 auf Seite 40) enorm wichtig.

Anforderung 26 (Bildung der transitiven Hülle) *Die Software muß die transitiven Hülle über die Beziehungen zwischen den Configuration Items bilden können. Für diese Hülle benötigt die Software ein Configuration Item als Ausgangspunkt und ein Suchkriterium für die gewünschten Beziehungen. Als Ergebnis liefert die Software eine Liste aller gefundenen Configuration Items.*

Profile Für die Definition der Schemas der Configuration Items soll der Architekt grundsätzlich beliebige Strukturen definieren können. Jedoch ergibt sich aus dem Szenario eine spezifische Eigenschaft, welche jede Software berücksichtigten muß. Da die Konfiguration der Rechner im dem vorliegenden Szenario von unterschiedlichen Profilen abhängt (vgl. dazu Abschnitt 3.2.2 auf Seite 17), soll die Software zu jedem Configuration Item und ihren Beziehungen auch das dazugehörige Profil definieren. Jedoch sind nicht alle Profile voneinander disjunkt und somit soll die Software auch die Zusammensetzung von Profilen ermöglichen.

Beispielsweise kann es ein Basisprofil existieren, welche die Hardwarekonfiguration enthält. Dieses Basisprofil ist immer ein Teil jedes anderen Profils, weil sich die Hardware zwischen den Profilen nicht ändert. Alle anderen Profile besitzen dann eine Referenz auf dieses Basisprofil und enthalten dann automatisch auch die Configuration Items aus diesem Profil.

Man kann auch die Profile mit Configuration Items und ihre Beziehungen umsetzen, jedoch erschwert es dann die Benutzung der Software, weil dann der Architekt die Pflege der Profile selber übernehmen muß.

Anforderung 27 (Profilen) *Alle Configuration Items und ihre Beziehungen sind einem Profil zugeordnet. Jedes Profil kann eine Spezialisierung eines anderen Profils sein.*

Agenten Da die manuelle Eingabe von Configuration Items sehr arbeitsaufwendig ist, soll auch eine Anbindung von speziellen Programmen, den Agenten, möglich sein, welche die notwendigen Information für ein gegebenes Schema automatisch eintragen können. Dazu muß aber Agent spezifisch für ein Schema programmiert werden.

Um jedoch mögliche Fehleingaben des Agenten zu erkennen, müssen alle Informationen, welche von Agenten stammen, entsprechend markiert werden. Damit können dann die Differenzen zwischen der manuellen Eingabe und der automatischen Erfassung erkannt und gegebenenfalls behoben werden (siehe dazu die Anforderung 29 auf der nächsten Seite).

Anforderung 28 (Agenten) *Die Software muß eine Schnittstelle anbieten, mit der externe Software automatisch neue Configuration Items anlegen oder vorhandene ändern kann.*

Unterscheidung zwischen Soll- und Ist-Zustand Für die Kontrolle der CMDB kann es sinnvoll sein, den Soll- und Ist-Zustand zu vergleichen. Der Ist-Zustand kann sinnvollerweise nur durch einen Agenten (siehe die Anforderung 28) erfolgen, damit nicht das Ergebnis durch eine falsche, manuelle Eingabe verfälscht wird.

Dabei muß die Daten über den Ist-Zustand klar vom Soll-Zustand getrennt werden. Diese Trennung kann die Software durch die Nutzung von speziellen Profilen (siehe die Anforderung 27) ermöglichen.

Diese Fähigkeit der Differenzierung des Soll- mit dem Ist-Zustand ermöglicht eine erhebliche Erleichterung für das Incident und Problem Management. Jedoch erfordert diese Funktion einerseits den Einsatz von Agenten und von Profilen und ist somit eine sehr komplexe Anforderung.

Anforderung 29 (Soll- und Ist-Zustand) *Die Software muß zwischen einem Soll- und Ist-Zustand der Configuration Items unterscheiden. Der Ist-Zustand muß automatisch erfaßt werden und dann mit dem Soll-Zustand verglichen werden.*

Prozessüberwachung In der Prozessüberwachung soll einerseits die Kostenentwicklung für den Prozess kontrolliert werden, so daß der Nutzen durch die Einführung des Configuration Managements nicht durch die entstehenden Kosten aufgehoben werden. Da der Nutzen des Configuration Managements nicht direkt meßbar ist, soll vorallem das Zugriffsverhalten der anderen Prozesse auf das Configuration Management untersucht werden, um mögliche Effizienzsteigerung für das Configuration Management zu erkennen (siehe dazu auch den Abschnitt 2.4.1 auf Seite 10).

Anforderung 30 (CMDB Protokoll) *Die Software soll das Zugriffsverhalten auf die CMDB protokollieren. Aus den Protokollen muß dann ein Bericht für die Prozessüberwachung des Configuration Management erstellt werden.*

5.2.4 Change Management

Das Ziel des Change Managements ist die sichere und effiziente Durchführung von Änderungen an der IT-Infrastruktur (siehe dazu den Abschnitt 4.5.1 auf Seite 45). Damit das Change Management auch effizient mit den anderen Prozessen zusammen arbeiten kann, müssen alle Änderungen in einer einheitlichen Form beschrieben werden. Daraus folgt, daß die Software die Form eines RfC vorgeben muß, damit sie innerhalb des Change Managements auch die enthaltenen Informationen verarbeiten kann.

Anforderung 31 (Eigenschaften einer RfC) *Jede Änderung an der CMDB muß in Form einer RfC beschrieben werden, welche folgende Eigenschaften erfüllt:*

- *Vollständigkeit*
- *Eindeutigkeit*
- *Konsistenz (siehe auch die Anforderung 51 auf Seite 66)*
- *Reversibilität (allgemeiner ist die Anforderung 50 auf Seite 66)*
- *Format ermöglicht eine automatische Änderung der CMDB (siehe die Anforderung 39 auf der nächsten Seite)*

In jeder RfC sollen dann mindestens folgende Informationen enthalten sein (vgl. die Liste im Abschnitt 4.5.3 auf Seite 47):

Anforderung 32 (Aufbau einer RfC) *Folgende Informatione muß jede RfC beinhalten:*

- *Beschreibung aller Änderungen an den betroffenen CIs*
- *Begründung für die RfC*
- *Priorität*
- *Notwendige Resourcen*
- *Abhängigkeit von anderen RfCs*
- *Bedingungen während der Umsetzung (Ausfallzeit?)*
- *Herkunft der RfC*

- *Termin, bis wann die Änderung umgesetzt worden ist*

Diese RfC durchläuft dann den vorgeschriebenen Prozess für das Change Management. Nach der Erfassung der RfC sieht der Prozess die Entscheidung vor, ob die RfC akzeptiert wird (siehe dazu den Abschnitt 4.5.3 auf Seite 47). Dabei kann die Software teilweise die formalen Vorgaben an eine RfC eigenständig überprüfen, wenn die Syntax für eine RfC in einer formalen Sprache festgelegt worden ist. Das Ergebniss dieser Entscheidung muß der Benutzer der Software mitteilen können, damit sie entsprechend darauf reagieren kann.

Anforderung 33 (RfC akzeptieren) *Die Software überprüft die Syntax jeder RfC und zeigt die RfC zur weiteren Überprüfungen dem Änderungsprüfer an. Der Prüfer kann dann die RfC akzeptieren und teilt seine Entscheidung der Software mit. Nur falls die RfC akzeptiert wurde, wird die weitere Bearbeitung von der Software zugelassen.*

Auch für die Klassifizierung (siehe Abschnitt 4.5.3 auf Seite 47) bietet die Software für jede Änderung eine Auswahl dem Änderungskoordinator an, die den weiteren Ablauf der RfC beeinflußen.

Anforderung 34 (RfC klassifizieren) *Die Software ermöglicht dem Änderungskoordinator die Einteilung der RfC in einen Bereich. die Zuordnung einer Priorität und der Beschreibung der möglichen Auswirkung.*

Anhand der Klassifizierung wird dann die Umsetzung nach den Resourcenvorgaben im Forward Schedule of Changes geplant, so daß die vereinbarten Termine für die RfCs auch eingehalten werden können (siehe auch die Anforderungen 60 auf Seite 69 und 59 auf Seite 69).

Anforderung 35 (Planung der Änderung) *Alle Aktivitäten im Change Management müssen im elektronischen Terminkalender verwaltet werden. Beachte dabei die Anforderung 60 auf Seite 69.*

Anforderung 36 (Forward Schedule of Changes) *Die Software verwaltet die Planung aller RfC unter Berücksichtigung der Klassifizierung und den vorhandenen Resourcen im Forward Schedule of Changes (kurz: FSC).*

Bevor eine RfC umgesetzt werden darf, muß jede RfC von den zuständigen Personen genehmigt werden (siehe dazu die Tätigkeit "Genehmigung" im Abschnitt 4.5.3 auf Seite 48). Dabei kann die Software den Ablauf der Genehmigung organisieren.

Anforderung 37 (Genehmigung der Änderung) *Jede Änderung benötigt eine finanzielle, technische und geschäftliche Genehmigung. Die Software muß dazu für jede Änderung diese Genehmigungen der zuständigen Personen einholen und archivieren.*

Nach dem Bestehen der erforderlichen Test (siehe Abschnitt 4.5.3 auf Seite 48) kann dann die RfC freigegeben werden und endgültig nach dem vorliegenden Zeitplan implementiert werden. Bei diesen Ablauf soll die Software vorallem die nächsten Schritte vorgeben und die Ergebnisse der einzelnen Aktivitäten erfassen.

Anforderung 38 (Testergebnisse einer Änderung) *Die Testergebnisse jeder Änderung müssen in der betroffenen RfC dokumentiert werden.*

Bei der Implementierung müssen aus Effizienzgründen die notwendigen Änderungen an der CMDB automatisch von der Software durchgeführt werden. Daraus ergeben sich Anforderungen an die Syntax für die RfC, so daß eine eindeutige und reversible Änderung an der CMDB möglich ist (siehe die Anforderung 31 auf der vorherigen Seite).

Anforderung 39 (Automatische Änderung der CMDB) *Jede Änderung in Form einer RfC muß automatisch die notwendigen Änderungen in der CMDB durchführen können. Zusätzlich muß auch die Umkehrung einer Änderung automatisch erfolgen (siehe auch die Anforderung 50 auf Seite 66).*

Nach der erfolgreichren Umsetzung einer RfC können die gesammelten Daten der Änderung für spätere Diagnosen im Incident und Problem Management sehr sinnvoll sein. Aus diesem Grund müssen die Details jeder Änderung auch später wieder auffindbar sein:

Anforderung 40 (RfC suchen) *Die Software muß die Suche nach den angelegten RfCs mit einem beliebigen Suchkriterium anbieten.*

Prozessüberwachung Die wichtigsten Erfolgsfaktoren für das Change Management sind die Kosteneffizienz für die Durchführung von Änderung und die Qualität der durchgeführten Änderung. Falls einer der beiden Faktoren abnimmt, muß der Prozess-Manager entsprechend reagieren.

Anforderung 41 (Kostenkontrolle des Change Managements) *Alle Kosten und die verwendete Arbeitszeit für eine Änderung müssen von der Software abgefragt und dokumentiert werden. Aus diesen Daten müssen dann die Berichte für die Prozessüberwachung erstellt werden (siehe auch die Anforderung 52 auf Seite 66).*

Für die Bestimmung der Qualität sind regelmäßige Audits der durchgeführten Arbeiten notwendig. Die Ergebniss dieser Audits kann dann die Software verwalten und in einen Zusammenhang mit den Kosten bringen (siehe dazu den Abschnitt **??** auf Seite ??).

Anforderung 42 (Qualitätskontrolle der Änderungen) *Die Ergebnisse eines Audits aller Änderungen müssen den betreffenden Änderungen in der Software zugeordnet werden.*

5.2.5 Release Management

Für das Release-Management muß die Software vorallem den Ablauf der neuen Releases unterstützen. Zu den üblichen Aufgaben der Informationsverwaltung kann der Software-Roll-Out weitgehend automatisiert werden. Damit ist auch eine schnelle und effiziente Methode für regelmäßige Sicherheitsupdates vorhanden.

Releases testen Ein wichtiger Schritt für das Erreichen eines hohen Qualitätsstandard beim Release Manageement ist das ausfürhliche Testen jedes Releases (siehe dazu den Abschnitt 4.6.3 auf Seite 51). Dafür benötigt das Release Management eine eigene Testumgebung, wo die unterschiedlichen Bedingungen simuliert werden können. Weil aber viele Tests sehr lange und umfangreich sein können, müssen die Durchführung der Tests so weit wie möglich automatisiert werden. Diese Automatisierung ermöglicht auch eine exakte Reproduktion der Testbedingungen, welche bei der Problemdiagnose eines Release sehr hilfreich sein kann.

Anforderung 43 (Automatische Tests) *Die Software muß alle Testdurchläufe der Releases automatisch durchführen können und die Ergebnisse dabei protokollieren. Am Ende soll ein Testbericht für den Releaseabnehmer verfügbar sein.*

Roll-Out Eine besondere Aufgabe kann die Software für das Release Management beim Roll-Out von Softwarerelease übernehmen, in dem es diese Roll-Outs automatisiert durchführt (siehe dazu den Abschnitt 4.6.3 auf Seite 52). Diese Automation ist aber abhängig von der eingesetzten Software und insbesondere abhängig vom Betriebssystem. Deshalb soll für das Roll-Out entsprechende Schnittstellen für plattformspezifische Installationsprogramme erstellt werden, so daß diese Automatisierung überhaupt möglich ist (siehe auch die Anforderung 63 auf Seite 69).

Anforderung 44 (Automatische Softwareverwaltung) *Alle Programme in der DSL müssen automatisch auf die ausgewählten Rechner installiert, konfiguriert und gewartet werden. Ebenfalls muß die Installation aller Updates automatisch erfolgen.*

DSL und DHS Zusätzlich verwaltet das Release Management die DSL und DHS. Dazu ist es auch erforderlich die Umsetzung der DSL und DHS regelmäßig zu überprüfen. Fall die Syntax für die DSL und der DHS auch maschinell lesbar ist, kann bei Bedarf auch diese regelmäßige Überprüfung teilweise automatisiert werden.

Beim DHS und für die Beschaffung der Softwarelizenzen für die DSL kann zusätzlich die Einbindung eines Warenwirtschaftssystems sinnvoll sein. Damit kann die Beschaffung von neuer Hardware und neuer Lizenzen zusammen mit dem gesamten Einkauf des Unternehmens integriert werden.

Anforderung 45 (Verwaltung der DSL und DHS) *Die Software muß die beiden Policen DSL und DHS verwalten und die Einhaltung überwachen. (Verwende dazu auch die Anforderungen 29 und 28 auf Seite 61)*

Prozessüberwachung Für das Release Management besitzt die Qualität der ausgelieferten Releases die höchste Priorität. Aus diesem Grund muß in der Prozessüberwachung die Auswirkungen eines Releases in Form von Incidents- und Problemmeldungen achten.

Anforderung 46 (Regressionen von Releases) *Alle negativen Auswirkungen eines Releases in Form von Störungen oder Problemen müssen den betreffenden Releases zugeordnet werden.*

5.3 Anforderungen nach Modulen

Die Softwareunterstützung für die einzelnen ITIL-Service-Support-Prozesse besitzen teilweise große Übereinstimmungen. Damit diese Gemeinsamkeiten nicht mehrmals eigenständig in Software umgesetzt werden, soll in diesem Abschnitt die Anforderungen der gemeinsamen Module definiert werden.

5.3.1 Benutzerverwaltung

Für die Unterscheidung der unterschiedlichen Benutzer braucht die Anwendung eine eigene Benutzerverwaltung, damit die verschiedenen Rollen in den Prozessen auch sinnvoll abgebildet werden können (siehe dazu Abbildung 2.2 auf Seite 6). Zuerst müssen alle Benutzer sich bei der Software sicher authentifizieren, damit kein unrechtmäßiger Zugriff möglich ist.

Für jede Aufgabe eines Benutzers erhält er eine entsprechende Rolle. Die Rollen beschreiben dann zuerst die Aufgabe und definieren dann zusätzlich die notwendigen Rechte für diese Aufgabe. Somit erhält in diesem Rollenmodell jeder Benutzer alle notwendigen Rechte für seine Arbeit.

Anforderung 47 (Benutzerauthentifikation) *Alle Benutzer müssen über ihren Benutzernamen und ein geheimes Passwort sicher authentifiziert werden.*

Anforderung 48 (Benutzerautorisierung) *Die Rechte eines Benutzers in der Software werden über seine zugewiesenen Rollen definiert.*

Die Verwaltung der Benutzer, der Rollen und ihre Rechte muß vollständig in der Software möglich sein, damit der Prozessmanager seine Aufgabe auch innerhalb der Software durchführen kann (siehe dazu die Abbildung 2.3 auf Seite 7).

Anforderung 49 (Benutzerverwaltung) *Die Benutzer, die Rollen und ihre Rechte müssen alle innerhalb der Software veränderbar sein.*

5.3.2 Datenmodell

Bei der Beschreibung der Anforderungen für die einzelnen Prozesse sind die notwendigen Informationen aufgezählt worden, welche die Software verwalten und verarbeiten muß. Da für die Prozessunterstützung viele Informationen gesammelt und verarbeitet werden, ergeben sich auch gemeinsame Anforderungen für alle Prozesse im Bezug auf ihre Datenverarbeitung. Diese gemeinsamen Anforderungen sollen nun genauer erklärt werden.

Verfügbarkeit alter Versionen Viele Informationen werden im Laufe der Bearbeitung durch einen Prozess geändert. Jedoch kann bei diesen Änderungen Fehler passieren, so daß die Software eine Änderung rückgängig machen soll. Zusätzlich kann für eine spätere Analyse die Dokumentation jeder Änderung neue Erkenntnisse zur Prozessoptimierung ermöglichen. Daraus ergibt sich die allgemeine Anforderung an das verwendete Datenmodell, daß alle Versionen verfügbar bleiben.

Anforderung 50 (vollständige Erhaltung aller Information) *Alle Versionen der Prozessinformationen müssen erhalten bleiben. Auch falls der Benutzer Daten entfernen möchte, müssen trotzdem die gelöschten wiederherstellbar sein. Für die Datenkonsistenz müssen alle Referenzen immer auf die aktuelle Version zeigen. Und für eine spätere Kontrolle müssen bei jeder Änderung der Name des Autors und der Zeitpunkt dokumentiert werden.*

Diese Verfügbarkeit alter Versionen erhöht den Speicherbedarf enorm, da aber die Prozesse vorallem Text verarbeiten, ist dieser Speicherbedarf mit den aktuellen Datenträgern sehr einfach zu erfüllen.

Eigenschaften einer Datenbank Bei der Verwaltung von Informationen ergeben sich viele Anforderungen, damit die Datenkonsistenz erhalten bleibt. Diese Anforderungen sind beispielsweise bei der Entwicklung von Datenbanksystemen wichtig und sollen auch für die Entwicklung dieser Software gelten.

Durch den Einsatz einer vorhandenen Datenbanksoftware können aber viele Anforderungen einfach erfüllt werden (siehe dazu das Buch über Datenbanken [kaea 01]).

Anforderung 51 (Eigenschaften einer Datenbank) *Die Software muß die Erhaltung der Datenkonsistenz garantieren können, in dem sie folgende Eigenschaften erfüllt*

- *Redundazfreiheit mit Normalformen für die Vermeidung diverser Anomalien*
- *Referentielle Integrität*
- *Unterstützung von Transaktionen*
- *ACID-Eigenschaften*

5.3.3 Erstellung von Berichten

Ein wesentlicher Bestandteil jedes Prozesses ist die Prozessüberwachung. Dazu müssen regelmäßige Berichte über die Leistungen und Defizite des Prozesses erzeugt werden. Und aus diesem Grund soll die Software aus den vorhandenen Daten automatisch aussagekräftige Berichte erstellen können.

Dazu werden der Aufbau und der Inhalt eines Berichtes erst definiert, damit dann die Software in regelmäßigen Abständen einen entsprechenden Bericht erstellen kann. Idealerweise kann der Benutzer auch noch die Erstellung des Berichtes über das Ändern eines Parameters beeinflussen.

Anforderung 52 (Erstellung eigener Berichte) *Die Software muß dem Benutzer die Definition neuer Berichte ermöglichen. Dabei kann der Benutzer die Selektion der Daten und die Auswertung der Daten selber definieren.*

Bei der Definition eines Berichtes ist zuerst die Zusammenstellung der Informationen für eine Datenbasis notwendig. Nachdem das Programm die notwendigen Daten besitzt, können diese für einen Bericht aufbereitet werden. Dazu gehört auch die Anwendung der verschiedenen Aggragtions- und Berechnungsfunktionen auf die vorhandene Datenbasis.

Beispielsweise können folgende Aggregationsfunktionen implementiert werden:

- Anzahl
- Summe
- Maximum
- Minimum
- Durchschnitt
- Median
- Gewichtetes Mittel
- Gewichtete Summe

Da die Berichte regelmäßig erstellt werden, muß der zu untersuchende Zeitraum im Bericht und damit auch in der Suchanfrage als Parameter erscheinen. Dies kann beispielsweise durch spezielle Variablen innerhalb der Suchanfrage realisiert werden. Der Wert dieser Variable wird dann vor der Berichterstellung abgefragt und bei der Erstellung des Berichtes dann entsprechend benutzt.

Anforderung 53 (Parameter eines Berichtes) *Jeder Bericht bezieht sich auf einen bestimmten Zeitraum, welcher der Benutzer beim Zusammenstellen des Berichts als Parameter angeben muß. Weitere Parameter können ebenfalls das Ergebnis eines Berichtes beeinflussen.*

5.3.4 Kommunikation

Für die Unterstützung der Managementaufgabe ist die Kommunikation zwischen den beteiligten Menschen die wichtigste Aufgabe, weil nur durch die Kommunikation können die verschiedenen Aufgaben und Arbeiten koordiniert werden. Aus diesem Grund muß die Software für die Unterstützung des Managements die Kommunikation fördern. Der Mangel an Kommunikation kann zu vielfältigen Problemen in der Zusammenarbeit von Personen führen, weil ihre eigene Arbeit nicht auf die Arbeit der anderen abgestimmt ist. Dieses Phänomen ist auch im Szenario als ein Problem erkannt worden (siehe Abschnitt 3.5.2 auf Seite 30).

Synchrone Kommunikation In einer synchronen Kommmunikation tauschen Sender und Empfänger ohne eine merkbare Zeitverzögerung ihre Informationen aus. Eine natürliche synchrone Kommunikationsform für den Menschen ist das Gespräch.

Eine technische Unterstützung erhält das Gespräches mit dem Telefon. Und aus diesem Grund soll die Software das Kommunikationsmedium Telefon fördern, in dem es die Telefonnnummern jeder Personen kennt und diese Telefonnummer jederzeit dem Anwender anbieten kann.

Anforderung 54 (Unterstützung der synchronen Kommunikation) *Die Software muß die synchrone Kommunikation zwischen den Benutzer fördern, in dem sie mindestens die Telefonnummer der Benutzer verwaltet und veröffentlicht.*

Asynchrone Kommunikation Eine viel bessere Unterstützung kann aber für die asynchrone Kommunikation angeboten werden. Bei der asynchronen Kommunikation kann die Übermittlung einer Nachricht vom Sender zum Empfänger länger dauern.

Eine weit verbreitete asynchrone Kommunikationsform im Internet ist die eMail. Und weil die eMailintegration relativ einfach ist, soll die Software die Übertragung von nicht zeitkritische Nachrichten mit Hilfe der eMail realisieren. Mit Hilfe der eMail kann die Software vorallem den Benutzer auf neue Informationen aufmerksam machen. Jedoch soll der Benutzer jederzeit die Kontrolle über das Versenden von eMails an seine Adresse besitzen, in dem er selber definieren kann, über welche Nachrichten er informert werden will.

Damit beim Lesen einer von der Software erstellten eMail auch der Übergang in die Software nahtlos abläuft, soll jede eMail eine direkte Verknüpfung an die richtige Stelle in der Software besitzen. Bei einer Webanwendung kann dies durch die Übermittlung einer URL erfolgen.

Anforderung 55 (Unterstützung der asynchronen Kommunikation) *Die Software muß die asynchrone Kommunikation zwischen den Benutzer ermöglichen, in dem sie einerseits die Adressen der Benutzer veröffentlicht und andererseits das Schreiben von Nachrichten an jeden Benutzer ermöglicht.*

5.3.5 Prozessunterstützung

Jeder Prozess beschreibt den Arbeitsablauf mit einer Kombination verschiedener Tätigkeiten. Damit die Software einen Prozess effektiv unterstützen kann, muß der Bedienungsablauf der Software zu den beschriebenen Prozessen passen.

Anforderung 56 (Prozessunterstützung) *Die Software soll die Einhaltung der vorgegebenen Reihenfolge der Prozessaktivitäten fördern, in dem sie Vorschläge für die nächsten Schritte macht.*

Jedoch führt eine strenge Vorgabe des Arbeitsablaufs zu einer Bürokratisierung, welche zu statischen Organisation und damit zu Effizienzverlusten führen kann (siehe [PRW 03]). Deshalb muß die Software auch gleichzeitig flexibel in der Nutzung sein.

Anforderung 57 (Flexibile Nutzung) *Die Software muß dem Benutzer auch Abweichungen vom vorgeschriebenen Arbeitsablauf erlauben. Dazu gehört auch die Korrektur einer schon abgeschlossenen Tätigkeit. Desweiteren muß sie das Überspringen von Abläufen ermöglichen, wenn der Benutzer dies ausdrücklich erwünscht.*

Die Benutzer ändern aber den vorgebenen Ablauf nur dann, wenn sie dafür auch Gründe besitzen. Diese Gründe für eine Änderung sind wichtige Informationen für die weitere Optimierung des Prozesses (siehe den Abschnitt 2.2.1 auf Seite 5).

Anforderung 58 (Protokollierung des Nutzungsverhalten) *Alle Aktionen eines Benutzers müssen unter Beachtung des Datenschutzes[1] protokolliert werden.*

Die Protokolle über das Nutzungsverhalten kann dann Hinweise für weitere Prozessoptimierungen geben.

5.3.6 Benachrichtigung über Ereignisse

Damit die Mitarbeiter schnell und richtig auf neue Ereignisse reagieren können, muß die Software die Benutzer benachrichtigen. Wenn diese Mitteilung entfällt, dann können die Betroffenen entweder gar nicht oder falsch darauf reagieren. Diese Situation ist auch im Szenario als ein Problem erkannt worden (siehe Abschnitt 3.5.2 auf Seite 30).

Dieses Problem kann die Software mildern, in dem sie die betroffenen Personen automatisch über wichtige Ereignisse informiert. Welche Ereignisse für eine Person wichtig sind, kann jedoch am besten die betroffene Person selbst entscheiden.

[1] Das Datenschutzgesetz (DSG) schreibt die Regeln für die Überwachung von Personen vor. (DSG 3. Abschnitt “Privater Bereich”)

Anforderung 59 (Ereignissnachrichten) *Die Software muß jedem Benutzer automatisch über Ereignisse informieren. Dabei kann der Benutzer die gewünschten Ereignisse auswählen, über die er informiert werden möchte. Die Übermittlung der Nachricht kann entweder über eine synchrone Kommunikation (siehe Anforderung 54 auf Seite 67) oder eine asynchrone Kommunikation (siehe Anforderung 55 auf der vorherigen Seite) stattfinden*

5.3.7 Terminplanung

Viele Prozesse benötigen ein abgestimmtes Arbeiten der verschiedenen Personen und diese Abstimmung kann mit einer gemeinsamen Terminplanung unterstützt werden. Aus diesem Grund soll die Software für jeden Benutzer einen Terminplan anbieten und organisieren. In diesem Terminplan kann dann der Benutzer seine Termine für die nächste Zeit eintragen und nachsehen. Für die Koordination der verschiedenen Benutzer ist eine Zusammenarbeit dieser Terminpläne sinnvoll, so daß die Benutzer ihre Termine aufeinander abstimmen können.

Anforderung 60 (Ein umfassender Terminplan) *Alle Termine und Aufgaben einer Person müssen in genau einem Terminplan organisiert werden.*

Anforderung 61 (Gruppenunterstützung im Terminplan) *Die Terminpläne aller Personen müssen zusammenarbeiten, so daß gemeinsame Termine von mehreren Personen in allen betroffenen Terminplänen korrekt vermerkt sind.*

Zusätzlich zu den Benutzerterminplänen können auch die Prozesse einen Terminplan besitzen. Ein in ITIL beschriebener Terminplan ist der FSC aus dem Change Management (siehe dazu die Tätigkeit “Planen” im Abschnitt 4.5.3 auf Seite 47). Der FSC muß deshalb in der Terminplansoftware integriert sein, so daß keine Terminkollisionen auftreten können.

Anforderung 62 (Terminsynchronisation mit dem Prozessablauf) *Es können aus der Software für die Prozessunterstützung auch automatische Termineinträge in den Terminkalender eingetragen werden, so daß der Prozessablauf mit dem Terminplan synchronisiert ist.*

Fazit Weil die Terminplanung sehr komplex sein kann und der Terminplan auch auserhalb der ITIL Prozesse sinnvoll eingesetzt werden kann, soll eine fertige Terminplansoftware eingesetzt werden. Zusätzlich soll für eine optimale Zusammenarbeit alle mit dem selben Terminplanprogramm arbeiten, damit Inkompabilitäten zwischen verschiedenen Programmen ausgeschlossen werden kann.

5.3.8 Zusammenarbeit verschiedener Programme

Viele Anforderungen können besser von schon vorhandenen Softwareprodukten erfüllt werden (siehe z.B. die Anforderung 60), so daß es nicht ökonomisch ist, alle Anforderungen mit einem Programm erfüllen zu wollen. Für einen effizienten Einsatz von verschiedenen Programmen müssen diese aber auch sehr eng zusammenarbeiten. Dies führt dazu, daß alle Programme Schnittstellen anbieten müssen, welche von anderen ohne Probleme genutzt werden können.

Anforderung 63 (Schnittstellen für externe Programme) *Jedes Programm muß sinnvolle Schnittstellen anbieten, damit der notwendige Informationsaustausch zwischen den verschiedenen Programmen stattfinden kann.*

5.4 Zusammenfassung

In diesem Kapitel sind die Anforderungen für eine Softwareunterstützung der Service Support Prozesse beschrieben. Diese Anforderungen folgen zum größten Teil aus den Prozessen, aber einige erfüllen spezifische Eigenschaften des gewählten Szenarios. Damit ist jedoch die Grundlage für das weitere Vorgehen geschaffen worden, um am Ende ein vollständiges Toolkonzept für die Prozesse zu erreichen.

Im nächsten Kapitel werden deshalb anhand den hier beschriebenen Anforderungen verschiedene Programme evaluiert. Die Ergebnisse der Evaluierung fliesst dann in die Vorstellung eines Toolkonzeptes aus vorhandenen Programmen ein.

6 Evaluierung

In diesem Kapitel soll ein kurzer Überblick über die auf dem Markt vorhandene Softwareprodukte gegeben werden, welche die Anforderungen (siehe Kapitel 5 auf Seite 54) erfüllen können.

Für eine vollständige Unterstützung aller besprochenen Prozesse ist entweder ein sehr umfangreiche und mächtige Software notwendig. Oder man untersucht, ob eine Kombination aus mehreren vorhandene Softwareprodukten für die Prozessunterstützung möglich ist. Aus ökonomischer Sicht ist jedoch der Einsatz von vorhandener Softwareprodukte häufig sinnvoller, weil die hohen Kosten der Softwareentwicklung auf alle Lizenznehmer verteilt werden und damit die Kosten für den einzelnen erheblich geringer sind.

Die ausgewählte Software ist zum großen Teil Open Source Software[1] und somit kostenlos im Internet verfügbar. Aber nicht nur der Kostenvorteil spricht für den Open Source Softwareeinsatz, sondern weil notwendige Anpassungenen oder Weiterentwicklungen sehr einfach möglich sind. Dieser Vorteil erlaubt es, die Programme an die Besonderheiten im Szenario anzupassen oder neue Schnittstellen zu den anderen Programmen zu implementieren.

Falls jedoch ein kommerzielles Produkt erhebliche Vorteile gegenüber den Open Source Produkten besitzt, so ist der Einsatz eines kommerziellen Produktes zu empfehlen. Jedoch beschränkt sich diese Evaluierung nur auf die kommerzielle Produkte, welche schon jetzt am Lehrstuhl eingesetzt werden.

6.1 Aufbau

Die Evaluierung von vorhandenen Produkten soll die Einsatzmöglichkeiten der Produkte demonstrieren. Am Anfang jeder Evaluierung wird die untersuchte Software kurz vorgestellt. In manchen Bereichen hat sich aber ein gemeinsamer Standard etabliert, so daß viele Programme einen sehr ähnlichen Funktionsumfang besitzen. In diesem Fall wird dann nicht ein einzelnes Programm evaluiert, sondern das Produktsegment.

In der Vorstellung werden die möglichen Prozesse beschrieben, welche mit dem vorgestellten Produkt unterstützt werden können. Jedoch bietet kein Open Source Programm eine vollständige Unterstützung aller Prozesse. Deshalb wird jedes Programm hier nur für den Einsatz einzelner Prozesse evaluiert.

Da die Anforderungen nicht alles bis in das letzte Detail festschreiben, können verschiedene Programme mit unterschiedlichen Ansätzen und Konzepte die selben Anforderungen erfüllen. Für eine sinnvolle Evaluierung muß deshalb auch das zu Grunde liegende Konzept und die Vorgehensweise in der Benutzung des Programms erklärt werden. Diese Erklärung beschreibt dann die angebotene Unterstützung des Programms für einen Prozess und führt gleichzeitig auch die verwendete Terminologie des Programms ein.

Innerhalb dieser Beschreibung werden dann die erfüllten Anforderungen aufgezählt. Das Fazit liefert dann eine Gesamtbewertung, wie gut das untersuchte Programm die Prozesse innerhalb des Szenarios unterstützen kann.

[1] Eine mögliche Definition von Open Source Software ist auf `http://de.wikipedia.org/wiki/Open_Source` zu finden.

6.2 Mailinglisten

Als erstes soll der Einsatz einer Mailingliste als Diskussionsforum für die verschiedenen Prozesse untersucht werden. Für den Betrieb von Mailinglisten existieren dabei verschiedene Programme, welche sich vorallem in der Bedienung der Administration unterscheiden. Allen gemeinsam ist die Einrichtung einer speziellen eMailadresse, welche alle ankommenden eMails an alle eingeschriebenen Personen weiterleitet. Eine Mailingliste stellt somit ein Forum für den Informationsaustausch bereit.

Zwei bekannte Programme zur Verwaltung von Mailinglisten sind *Mailman*[2] und *Majordomo*[3]. Am Lehrstuhl werden schon mehrere Mailinglisten betrieben und damit haben die Mitarbeiter Erfahrungen gesammelt und kennen die Vor- und Nachteile einer Mailingliste.

Das Arbeitsprinzip einer Mailingliste ist sehr einfach: Sie verteilt alle eMails, welche an die Mailingliste geschickt werden, an alle eingeschriebenen Teilnehmer. Die Mailingliste ermöglicht damit komfortabel die Kommunikation über eMails innerhalb einer ganzen Gruppe. Weil alle in einer Gruppe die selben Informationen erhalten, bleiben alle Mitglieder auf dem aktuellen Stand der Dinge. Somit können Probleme vermieden werden, weil jemand nicht informiert worden ist. Damit fördert die Mailingliste die Informationsaustausch in einer Gruppe und erfüllt damit teilweise die Anforderung 55 auf Seite 68.

6.2.1 Unterstützung für Incident und Problem Management

Der Einsatz einer Mailingliste ist sehr flexibel und kann in vielen Bereichen eingesetzt werden. Hier soll nun der Einsatz einer Mailingliste für die beiden Prozesse des Incident und Problem Managements beschrieben werden.

Vorgehen Beim Auftreten einer Störung oder nach der Identifikation eines Probleme wird eine eMail an die passende Mailingliste mit der ausführlichen Beschreibung geschickt. Diese Meldung kann formlos sein. Damit aber später auch die Störungen bzw. Probleme später leichter aufgefunden werden können, soll ein Format oder ein Formular für die Meldung von Störungen bzw. Problemen vereinbart werden. Und zusätzlich sichert die Formatvorgabe, daß jede Meldung alle notwendigen Informationen enthält und damit die Bearbeitung reibungslos anlaufen kann. Damit erfüllt die Mailingliste die Anforderungen 1 auf Seite 55 und 9 auf Seite 57.

Für ein kleines Team kann sogar nur eine Mailingliste für das Incident und Problem Management ausreichen, weil beide Prozesse sehr viele Gemeinsamkeiten besitzen und das Volumen an eMails für beide Prozesse überschaubar bleibt. Über die Mailingliste können dann die Störungen und die Probleme gemeinsam besprochen werden und eventuell werden die Zusammenhänge damit einfacher sichtbar (erfüllt damit teilweise die Anforderung 8 auf Seite 57). Die notwendigen Untersuchungen und Arbeiten müssen dabei auch über die Mailingliste koordiniert werden, so daß keine Arbeit mehrfach von mehreren Personen durchgeführt werden. Normalerweise kündigt man deshalb in der Mailingliste an, welche Arbeiten man plant zu unternehmen. Und nachdem man seine Arbeit abgeschlossen hat, stellt man das Ergebniss seiner Arbeit ebenfalls in der Mailingliste vor.

Bei einfachen Problemen oder Störungen kann deshalb der Arbeitsablauf in einer Mailingliste wie folgt aussehen:

1. **Meldung** Sende eine eMail mit der Beschreibung einer Störung oder eines Problemes an die Mailingliste (erfüllt Anforderungen 1 und 9)
2. **Zuweisung** Einer übernimmt die Lösung der gemeldeten Störung bzw. Problems (erfüllt die Anforderung 11) und kündigt dies mit einer eMail an

[2] *Mailman*: `http://www.list.org/`
[3] *Majordomo*: `http://www.greatcircle.com/majordomo/`

3. **Abschluß** Nach erfolgreicher Lösung des Problems bzw. Störung beschreibt der Durchführende seine Lösung (erfüllt die Anforderung 6 und 12)

Zu jedem Zeitpunkt können natürlich über die Mailingliste neue Fragen gestellt und beantwortet werden. Falls mehrere mögliche Lösungswege existieren, können auch die Vor- und Nachteile der unterschiedlichen Lösungen diskutiert werden. Und am Ende einigt man sich dann auf eine Lösung (erfüllt damit die Anforderung 13 auf Seite 58).

Nach dem Beheben einer Störung müssen zusätzlich alle beteiligte Personen informiert werden. Falls aber dieser Personenkreis nicht die Mailingliste liest, müssen diese Personen auf einen anderen Weg informiert werden. Entweder informiert man die betroffenen Personen direkt per eMail oder man richtet eine Mailingliste für alle Mitarbeiter ein und leitet die Bestätigung der Störungsbehebung an diese Mailingliste weiter. Mit diesem Umweg kann dann auch die Anforderung 6 auf Seite 56 erfüllt werden.

Fazit Mit dem Einsatz einer Mailingliste kann somit die Meldung von Problemen, die Koordination der notwendigen Arbeiten, und der Informationsaustausch zwischen den Beteiligten effektiv unterstützt werden.

Jedoch kann eine Mailingliste nicht alle notwendigen Anforderungen für das Incident und Problem Management abdecken. Da die Informationen in einer Mailingliste eher formlos sind, fehlen effiziente Recherchemöglichkeiten. Ein großer Mangel ist die eingeschränkte Fähigkeit nach bestimmten Informationen zu suchen und es können nur über eine Volltextsuche einzelne eMails aufgespürt werden. (verfehlt damit die Anforderung 3 auf Seite 56). Jedoch ist für die Recherche von alten Störungen oder Problemen es notwendig, daß alle eMails der Mailingliste archiviert werden und die Suche auch auf dieses Archiv direkten Zugriff besitzt. Desweiteren bietet die Mailingliste nur unzureichende Informationen für eine spätere Analyse der durchgeführten Arbeiten (verfehlt ebenso die Anforderung 7 auf Seite 56), weil die dafür notwendige Informationen nur in natürlicher Sprache vorliegen und somit für ein Programm nur schwer zugänglich sind.

6.2.2 Unterstützung für Change Management

Viele Open Source Projekt nutzen auch Mailinglisten, um ihre Arbeiten am gemeinsamen Projekt zu koordinieren. In diesen Fällen wird die Mailingliste vorallem zur Besprechung der Weiterentwicklung des Projektes genutzt. Und damit ergeben sich viele Ähnlichkeiten der Mailinglisten in Open Source Projekte und mit dem Change Management. Aus diesem Grund soll der Einsatz einer Mailingliste für das Change Management genauer untersucht werden.

Vorgehen Jede Änderung beginnt man damit, daß man eine eMail mit der RfC an die Mailingliste für das Change Managemenet sendet. Für die Einhaltung der Anforderung 32 auf Seite 62 ist eine Formatvorlage für die eMail mit der Rfc erforderlich. Weil aber die Mailingliste den Inhalt der eMails nicht interpretieren kann, ist die Zusammenarbeit mit einer CMDB nur dann möglich, wenn Änderungen an der CMDB auch über eMails möglich sind. Da dies jedoch sehr unwahrscheinlich ist und keine Leistung der Mailingliste darstellt, erfüllt sie damit auch nicht die Anforderungen 31 und 39 auf Seite 63.

Die Entscheidung, ob die eingereichte Änderung akzeptiert wird, kann ebenfalls sehr einfach mit einer eMail an die Mailingliste dokumentiert werden (erfüllt die Anforderung 33 auf Seite 63). Falls die RfC akzeptiert wird, kann sie auch gleich klassifiziert werden, welche ebenfalls über das Versenden einer eMail dokumentiert wird. Prinzipiell wäre damit die Anforderung 34 auf Seite 63 erfüllt.

Jedoch zeigt sich hier ein wesentlicher Nachteil beim Einsatz einer Mailingliste: Weil jede neue Information nur über eine neue eMail dokumentiert werden kann, ist der Zusammenhang zwischen der eingereichten RfC und der neuen Information nur sehr lose. Dies führt dann bei einer Suche zu Problemen, weil die Verknüpfung aller eMails zu einer Störung bzw. zu einem Problem für die Software nicht sichtbar ist. Deshalb besitzt die Mailingliste erhebliche Probleme beim Erfüllen der Anforderung 40 auf Seite 64.

Der nächste Schritt im Change Management ist die Planung des weiteren Vorgehens. Auch hier kann die Mailingliste nur das Ergebnis der Planung veröffentlichen. Damit erfüllt die Mailingliste nur eingeschränkt die Anforderungen 35 und 36 auf Seite 63. Für die Genehmigung und das Testen einer Änderung (siehe die Anforderungen 37 und 38 auf Seite 63) bietet sie ebenfalls nur ein Forum zum Diskutieren und Veröffentlichen von Entscheidungen oder Fakten an. Und weil die Mailingliste keine Informationen über die Störungen oder der Probleme verarbeiten kann, kann sie auch keine Berichte für die Prozessüberwachung erstellen. (verfehlt die Anforderungen 41 und 42 auf Seite 64).

Fazit Die Mailingliste kann nur sehr eingeschränkt den Prozess des Change Managements unterstützen. Jedoch kann die Mailingliste für eine sehr primitive Unterstützung eines einfachen Change Managements gute Dienste leisten. Doch der größte Nutzen zeigt sich nur in Verbindung mit anderen Programmen, welche die Mängel der Mailingliste beheben können. Dieses Modell folgen auch die meisten Open Source Projekte, in dem sie die Kombination einer Mailingliste mit einem Versionsverwaltungprogramm [4] nutzen.

6.3 OTRS - Open Ticket Request System

Die Software *OTRS* ist ein *Open Source Ticket Request System*, welche die effiziente Bearbeitung von Trouble Tickets per eMail oder Telefon oder über das WWW ermöglicht. Die Lizenz für *OTRS* ist die bekannte GNU GPL, unter der viele Open Source Software vertrieben werden. Auf der OTRS-Homepage[5] ist neben der eigentlichen Software auch die Dokumentation und ein Demosystem zum Ausprobieren verfügbar.

Der Einsatz von *OTRS* kann vorallem den Service Desk, das Incident Management und mit Einschränkungen auch das Problem Management unterstützen. *OTRS* sieht sich als Trouble Ticket System an und übernimmt damit auch die üblichen Bezeichnungen für ein Trouble Ticket System. In dieser Evaluierung soll *OTRS* vorallem für den Einsatz im Incident Management untersucht werden, weil diese Arbeit einerseits den Service Desk nicht behandelt und andererseits gute Gründe gegen den Einsatz von *OTRS* im Problem Management existieren.

Ein Grund, warum es besser im Incident als im Problem Management eingesetzt werden soll, ist die Zielausrichtung: (siehe dazu den Abschnitt 4.2.1 auf Seite 33). Jedes Trouble Ticket soll schnell behoben werden. Aus diesem Grund bietet es eine sehr effiziente Arbeitsaufteilung aller Trouble Tickets.

Zuerst unterscheidet das Programm *OTRS* zwei Personengruppen, nämlich die Kunden und die Agenten. Die Kunden eröffnen neue Trouble Ticket mit dem Versenden einer eMail und die Agenten bearbeiten dann diese Tickets. Für eine weitere Aufgabenteilung erlaubt es die Aufteilung der Agenten in verschiedene Gruppe, welche dann unterschiedliche Trouble Tickets bearbeiten können.

Vorgehen Ein wesentliches Merkmal von *OTRS* ist die intensive Nutzung der Kommunikation über eMails (Anforderung 55 auf Seite 68). Man eröffnet ein neues Trouble Ticket sehr einfach mit dem Versenden einer eMail an *OTRS*. Aus diesem Grund benötigt *OTRS* wie auch eine Mailingliste mindestens eine eigene eMailadresse. Nach dem Eintreffen erhält jede eMail automatische eine eindeutige Trouble Ticket Nummer. Mit dieser Nummer kann dann jede weitere eMail genau einem Trouble Ticket zugeordnet werden.

Für eine bessere Übersicht über die Trouble Tickets können die Tickets in verschiedene Queues (engl. für Warteschlange) eingeordnet werden. Die Namen der Queues und die Regelen für die automatische Einordnung eines Trouble Tickets in eine Queue kann von dem Administrator beliebig konfiguriert werden. Beispielsweise können mehrere eMailadressen für OTRS eingerichtet werden und die Regeln beschreiben dann die Zuordnung der Adressen auf die Queues, so daß jede eMail an eine Adresse in die entsprechende

[4] bekanntes Beispiel dafür ist das Programm *CVS*

[5] OTRS-Homepage: `http://www.otrs.org/`

Queue automatisch eingeordnet wird. Die Arbeitsweise der Queues kann man sich wie die Ordner in den eMailprogrammen vorstellen. Weil das Programm *OTRS* noch weitere Ähnlichkeiten mit eMailprogrammen besitzt, soll die nächste Tabelle diese beiden Programme gegenüberstellen:

OTRS	**eMailprogramm**
Trouble Ticket	eMail
Queue	Ordner
Zuweisungsregeln für Queues	Filterregeln
Priorität eines Tickets	Priorität einer eMail
Trouble Ticket Nummer	Message ID der eMail (mit Einschränkungen)

Der nächste Schritt in der Bearbeitung eines Tickets ist die Übernahme durch einen Agenten. Dazu erhält jeder Agent eine Liste von neuen Trouble Tickets. Aus dieser Liste kann sich dann jeder Agent seine Trouble Tickets mit dem Befehl Lock auswählen und ab diesen Zeitpunkt ist er dann alleine für die Bearbeitung verantwortlich. Weil jeder Agent für seine Störungen alleine verantwortlich ist und keine Zusammenarbeit der Agenten in *OTRS* vorgesehen ist, können nur sehr schwer die Zusammenhänge mehrerer Störungsmeldungen erkannt werden. Falls also mehrere Meldungen die selbe Störung beschreiben, wird in *OTRS* jede Störungsmeldung einzeln bearbeitet (und erfüllt damit nicht die Anforderung 4 auf Seite 56).

Mit der Übernahme der Störung kann der Agent die Priorität und die Queue des Tickets beliebig ändern (erfüllt damit die Anforderung 2 auf Seite 55). Nachdem die Störung klassifiziert wurde, muss die Störung vom Agenten so schnell wie möglich untersucht und behoben werden. Jedoch stellt das Programm *OTRS* dazu keine weitere Hilfestellung. Der Agent kann dann nur die Ergebnisse seiner Arbeit innerhalb des Tickets dokumentieren (erfüllt damit die Anforderung 5 auf Seite 56).

Zum erfolgreichen Abschliessen einer Störung muß der Agent dem Kunden eine eMail schreiben, in der er die Behebung der Störung beschreibt. Damit erfüllt das Programm auch die Anforderung 6 auf Seite 56.

Bei der Bearbeitung von vielen Störungen ergeben sich weitere Möglichkeiten der Effizienzsteigerungen. Zum einen kann das Programm *OTRS* automatisch den Kunde über den aktuellen Stand per eMail informieren. Dazu können für jedes Ereignis (z.B. Änderung der Priorität oder der Queue) eine Musteremail erstellt werden, welche dann automatisch beim Eintreten des Ereignisses an den Kunden versendet wird. Damit erfüllt das Programm *OTRS* die Anforderung 59 auf Seite 69 ausgezeichnet.

Zusätzlich können auch Vorlagen für die normale Kommunikation zwischen dem Agenten und dem Kunden definiert werden. Damit wird die Arbeit der Agenten erheblich vereinfacht, weil sie einfach aus der Liste der Vorgaben eine eMail aussuchen können, welche an den Kunden verschickt werden soll.

Durch die Bearbeitung vieler Trouble Tickets entwickeln die Agenten ein Fachwissen über das zu betreuende System. Und damit dieses Wissen auch den anderen Agenten zur Verfügung steht, bietet *OTRS* die Verwaltung eines FAQ an. Diese FAQ kann öffentlich für alle Benutzer sein oder auch nur für den internen Gebrauch der Agenten bestimmt sein.

Bei der Bearbeitung von vielen Trouble Tickets ist die Kontrolle des Prozesses ohne Softwareunterstützung nur schwer möglich. Aus diesem Grund bietet das Programm *OTRS* die statistische Auswertung der Trouble Tickets. Diese Statistiken können einfache Trends aufzeigen und sind damit sehr hilfreich für die Prozessüberwachung (siehe die Anforderung 7 auf Seite 56). Wünschenswert wäre noch die genauere Analyse der Störungen, so daß besser die Zusammenhänge zwischen den Störungen erkannt werden. Weil aber *OTRS* die Ursachen der Trouble Tickets nur formlos in eMails dokumentiert, kann es auch keine Analyse der Ursachen durchführen. Damit ist die Weiterverarbeitung der Störungsmeldung innerhalb des Problem Managements nur sehr eingeschränkt möglich (siehe dazu die Anforderung 7 auf Seite 56).

Zusammenarbeit mit einer CMDB Eine weitere Ursache für die eingeschränkte Ursachenanalyse ist die fehlende Anbindung an eine CMDB und somit fehlt dem Programm *OTRS* einfach die Information, auf welche Configuration Items sich die Störungen beziehen. Es bietet nur zwei Möglichkeiten an, auf externe Informationen zu verweisen. Die erste Methode ist die Verwendung von den vier freien Feldern (im Programm: Free Fields). Jedes Feld besteht dabei aus einen Variablennamen und dem zugewiesenen Wert und darin kann man beispielsweise die Identifikationsnummern der betroffenen Configuration Items ablegen. Und man kann dann später nach den Werten in diesen freien Felder suchen und damit sind eventuelle Zusammenhänge zwischen den Trouble Tickets erkennbar.

Die zweite Methode ist die Aufnahme von einer URL, welche auf externe Informationen zeigt, in die Beschreibung des Trouble Tickets. Die Idee der Einbettung von URLs in einen Text ist sehr verbreitet in vielen eMailprogrammen, welche dann die URLs im Text automatisch erkennen und zu einem Hyperlink verwandeln. Und weil das Programm *OTRS* sehr viel mit eMails arbeitet ist die Übernahme dieser Idee offensichtlich. Jedoch zeigt sich der große Nachteil dieser Methode im direkten Vergleich zu den freien Feldern, weil die Suche die eingetragenen URLs nur mit einer Volltextsuche auffindbar sind. Weshalb benötigt man diese Methode der Einbindung externer Informationen, wenn die erste Methode erheblich besser ist? Die Verwendung der freien Felder ist jedoch auf nur vier Felder eingeschränkt und kann damit nur eine sehr kleine Anzahl von Zusatzinformationen aufnehmen. Diese Felder können beispielsweise die Identfikationsnummern anderer Trouble Tickets aufnehmen, wenn ein Ticket zu diesen in einem Zusammenhang stehen. Auch die Aufnahme der CI Ids wäre eine sinnvolle Verwendung der Felder. Eine Möglichkeit diese Beschränkung aufzuheben besteht natürlich in der Änderung des Quellcodes, damit mehrere Beziehungen eingegeben werden kann.

Die Erhöhung der Anzahl der freien Felder ist sinnvoll, weil damit auch die Beziehungen zwischen den Trouble Tickets noch besser dokumentiert werden könnte (siehe dazu die Anforderung 4 auf Seite 56). Dann können alle Trouble Tickets, welche die selbe Störung beschreiben, über die freien Felder miteinander verknüpft werden und die beauftragten Agenten können darüber besser zusammen an einer Lösung arbeiten. Daran erkennt man die möglichen Vorteile von vielen freien Feldern, weil dann die Zusammenhänge zwischen den Trouble Tickets und auch zu den externen Informationen besser zum Vorschein kommen.

Zusammenfassend kann man feststellen, daß die Zusammenarbeit von *OTRS* mit anderen Programmen nur sehr eingeschränkt möglich ist. Jedoch kann es damit ein Teil der Anforderung 63 auf Seite 69 erfüllen.

Fazit Das Programm *OTRS* erlaubt eine sehr effiziente und schnelle Bearbeitung von vielen Trouble Tickets und ist somit für die Unterstützung des Incident Managements geeignet. Jedoch sind seine Stärken auch gleichzeitig seine Schwächen, weil in der späteren Analysephase die notwendigen Informationen nicht vorhanden sind (siehe dazu die Anforderungen 7 und 8 auf Seite 57). Und damit erschwert es mögliche grundlegende Probleme zu identifizieren. Die offensichtlichen Schwächen zeigen auch, daß das Programm *OTRS* überhaupt nicht für das Problem Management geeignet ist, weil für eine optimale Problemlösung die Agenten enger zusammenarbeiten müssen.

6.4 ITracker

ITracker ist nach eigener Auskunft ein "Issue Tracking System". Die ITracker Homepage[6] ist sehr schlicht und bietet für das direkte Ausprobieren ein Demosystem an. Es ist ein J2EE 1.3 Anwendung und aus diesem Grund benötigt *ITracker* einen Java J2EE 1.4 kompatiblen Application Server als Laufzeitumgebung. Ein bekannter Open Source J2EE Application Server ist JBoss[7]. Zusätzlich benutzt *ITracker* eine beliebige JDBC-kompatible Datenbank für die Informationsablage.

Die Beschreibung von *ITracker* enthält den Begriff Issue, welches allgemeine ein Thema oder ein Problem beschreiben kann. Die verschiedenen Issues werden dann in ein oder mehrere Projekte zusammengefasst. Jedes Projekt kann dabei für die Issues beliebige viele eigene Felder definieren, so daß dieses Programm sehr flexibel einsetzbar ist.

[6] ITracker Homepage: `http://www.cowsultants.com/`
[7] JBoss: `http://www.jboss.org/`

Vorgehen In dieser Evaluierung soll der Einsatz für das Incident und Problem Management untersucht werden. Zuerst eröffnet man dazu ein oder mehrere Projekte, in denen die Störung- bzw. Problemmeldungen verwaltet werden sollen. Dann können für jedes Projekt noch weitere frei definierbare Felder hinzugefügt werden. Jedes Feld besitzt dabei ein Datentyp für eine bessere Kontrolle der Dateneingabe. Aktuell unterstützt das Programm *ITracker* folgende Datentypen:

- **String** für normale Texteingabe
- **Integer** für die Eingabe von ganzen Zahlen; z.B. eine Anzahl
- **Date** für die Eingabe eines Datums
- **Lists** für die Auswahl aus einer Liste vorgegebener Werte

Durch die individuellen Anpassungen können folgende Anforderungen sehr einfach erfüllt werden:

- **Beschreibung einer Störung** siehe die Anforderung 1 auf Seite 55
- **Beschreibung eines Problems** siehe die Anforderung 9 auf Seite 57
- **Klassifizierung der Störung** siehe die Anforderung 2 auf Seite 55
- **Klassifizierung eines Problems** siehe die Anforderung 10 auf Seite 57

Damit das Programm auch von mehreren Benutzer gleichzeitig verwendet werden kann, erlaubt das Programm auch die Verwaltung der Benutzer mit verschiedenen Rechten in den Projekten. Und damit erfüllt es gleich die Anforderungen 47 und 49 auf Seite 65. Falls man die Projekte aufgabenorientiert anlegt, dann ist auch die Anforderung 48 erfüllbar.

Nachdem die Projekte für das Incident und Problem Management eingerichtet sind, können die Benutzer sehr bequem über das WWW neue Issues in die verschiedenen Projekte eintragen und bearbeiten. Zusätzlich zu der Weboberfläche bietet das Programm auch die Verwaltung der Issues über die Verwendung eines *Eclipse* [8] Plugins an. Und für die direkte Einbindung in andere Programme bietet es sogar eine Web Service[9] API an. Damit erfüllt es vorbildlich die Anforderung 63 auf Seite 69.

Mit dem Eintragen der Issues sind die Probleme oder Störungen jedoch nicht beseitigt. Aber falls sich durch die Bearbeitung eines Issues eine Änderung notwendig wird, so kann man diese Änderung auch bequem über die angebotenen Oberflächen durchführen. Jedoch gibt das Programm *ITracker* nicht die Reihenfolge der Arbeitsschritte vor, so daß jederzeit alle Felder eines Issues bearbeitet werden können. Damit der Verlauf der Änderungen auch dokumentiert wird, fordert das Programm bei jeder Änderung einen Kommentar hinzuzufügen (minimale Umsetzung für die Anforderung 5, 12 und 50). Weil jedoch mehrere Personen direkten Änderungszugriff auf die Issues besitzen können, bietet das Programm eine automatische Benachrichtigung über eMail bei einer Änderung an. Damit erfüllt es auch die Anforderung 6 und 59 auf Seite 69.

Last but not least bietet das Programm auch die Erstellung von Berichten für die Projekte an. Für die Berichte nutzt ITracker die freie Bibliothek JFreeReport[10] und erfüllt damit ausgezeichnet die Anforderung 52 auf Seite 66.

Fazit Das Programm ITracker erfüllt sehr viele Anforderungen durch seine hohe Flexibilität und kann damit sowohl das Incident Management als auch das Problem Management unterstützen. Im Gegensatz aber zum Programm *OTRS* gibt es keine Vorgehensweise vor, so daß die Benutzer ihre Arbeitsweise selber kontrollieren müssen. Wenn man beispielsweise nur die Priorität eines Issues ändern möchte, dann bietet das Programm *ITracker* trotzdem die Änderung aller Felder des Issues an. Im Vergleich dazu bietet das

[8] Eclipse ist eine Open Source Entwicklungsumgebung und Plattform für die Sprache Java. Die Homepage von Eclipse ist: `http://www.eclipse.org/`
[9] Web Service: `http://www.w3.org/2002/ws`
[10] JFreeReport: `http://www.jfree.org/`

Programm *OTRS* einen eigenen Menüpunkt, in dem man nur die Priorität des Trouble Tickets festlegen kann.

Ein kleines Problem ist aber die Performance des Programms: Sowohl bei der Benutzung des öffentlichen Demosystems als auch der eigenen Testinstallation brauchte das Programm *ITracker* häufiger mehrere Sekunden, um eine Anfrage zu bearbeiten. Welche Gründe diese Verzögerungen besitzen, konnte jedoch nicht mit Bestimmheit herausgefunden werden.

6.5 Einsatz von Officepaketen

Häufig werden in Organisation für viele Aufgaben die bekannten Officepakete eingesetzt, weil diese Programme sehr flexibel in der Anwendung und auch von Computeranfängern zu bedienen sind. Die bekannten Officepakete umfassen folgende Programme:

- **Textverarbeitung** zur Erstellung von Dokumenten
- **Tabellenkalkulation** zum Rechnen (wird häufig als einfache Datenbank missbraucht)
- **Terminkalender** zum Organisieren der verschiedenen Termine
- **eMailprogramm** zum Verschicken und Lesen von eMails

Die Verwendung der oben beschriebenen Programme ist offensichtlich und deshalb soll hier der spezifische Einsatz von zwei Programmen für die Unterstützung der Service Support Prozesse beschrieben werden. Als erstes soll der Einsatz der Tabellenkalkulation als CMDB vorgeführt werden und danach sollen die Vorteile für die Zusammenarbeit durch den effektiven Einsatz eines elektronischen Terminkalenders demonstriert werden.

Tabellenkalkulation Das Ziel einer Tabellenkalkulation ist die Durchführung von einfachen Berechnungen. Jedoch benutzen viele Anwender eine Tabellenkalkulation als eine Datenbank, in denen sie ihre gesammelten Daten eingeben und verarbeiten. Damit löst in vielen Fällen die Tabellenkalkulation den Karteikasten ab.

Und deshalb ist es auch nicht ungewöhnlich, wenn die Papierdokumentation der IT-Infrastruktur häufig zuerst in eine Tabellenkalkulation eingegeben wird. Diese Methode wird deshalb sogar offiziell im itSMF Buch [BKP 02] vorgeschlagen:

> Das Management der CIs erfolgt in der Konfigurations-Management-Datenbank (Configuration Management Database, CMDB). Man kann sich die CMDB als große Kartei vorstellen, in der sämtliche IT-Betriebsmittel registriert und die verschiedenen Beziehungen zwischen den einzelnen Karten festgehalten werden. In ihrer einfachsten Form basiert die CMDB auf einfachen Formularen oder auf einer Reihe von Spreadsheets.

Der Vorteil einer Tabellenkalkulation ist der einfache Umgang und die flexible Gestaltung der Dateneingabe. Aber mit steigender Anzahl der Configuration Items zeigen sich auch die Nachteile beim Einsatz einer Tabellenkalkulation:

- keine Unterstützung für die gleichzeitige Bearbeitung durch mehrere Benutzer
- die Beziehungen zwischen den CIs sind sehr leicht zerbrechlich
- und komplexe Beziehungen können nur schwer eingegeben werden
- existiert keine Überprüfung, ob die Eingabe korrekt ist

Aus diesen Gründen darf eine Tabellenkalkulation nur für eine überschaubare Menge von Configuration Items als CMDB missbraucht werden. Jedoch kann der Einsatz einer Tabellenkalkulation der erste Schritt für die Einführung einer professionellen CMDB sein, weil dann schon Erfahrungen über den Aufwand und den Nutzen einer CMDB gesammelt worden sind.

Terminkalender Schon in den Anforderungen wird der Einsatz eines elektronischen Terminplanes gefordert (siehe dazu den Abschnitt 5.3.7 auf Seite 69). Aus diesem Grund ist der Einsatz einer vorhandenen Terminplanungssoftware sinnvoll.

Gleich am Anfang steht die Forderung, daß alle Termine in genau einem Terminplan zu verwalten ist (siehe Anforderung 60 auf Seite 69). Wenn nämlich mehrere unabhängige Termineplanungsprogramme eingesetzt werden, dann können Terminkollisionen auftreten, weil die betroffenen Termine in unterschiedlichen Programme eingetragen worden sind. Aus diesem Grund muß sich jeder auf einen Terminplaner beschränken. Für den Fall, daß auf unterschiedlichen Computern (oder PDA, Handy, usw.) jeweils ein Terminplanungssoftware benötigt werden, dann müssen die eingesetzten Programme ihre Termine synchronisieren können.

Ein zusätzlicher Nutzen bringt die Vorgabe eines Terminplanungsprogramms für die ganze Organisation, weil dann auch sehr einfach Termine für Gruppen organisiert und verwaltet werden können (dies beschreibt auch die Forderung 61 auf Seite 69).

In vielen Prozessen müssen die Prozessausführenden ihre Aktivitäten koordinieren und planen und sollen deshalb auch ein Terminplanungsprogramm nutzen. Eine besondere Bedeutung erhält der Terminplaner aber im Change Management, weil in ITIL explizit ein Forward Schedule of Changes beschreiben wird. Diese Beschreibung führt dann zu der Anforderung 62 auf Seite 69.

Um Termine zwischen verschiedenen Programmen auszutauschen haben sich verschiedene Standards gebildet, welche ein Dateiformat beschreiben. Damit können die Programme ihre Termine exportieren und später wieder importieren. Einer der bekanntesten Dateiformate ist der “iCal” Standard und sein Vorgänger “vCal”. Damit soll die Anforderung 62 auf Seite 69 auch unabhängig von der eingesetzten Terminplanungssoftware erfüllbar bleiben.

Fazit Die genaue Wahl für ein Terminplanungsprogramm hängt sehr viel mehr von den vorhandenen Plattformen und den persönlichen Vorlieben als von harten Kriterien ab. Vorallem bei kommerziellen Programmen existieren nur sehr geringe Unterschiede, weil sie die Terminplanung sehr umfassend unterstützen. Um jedoch alle Vorteile dieser Programme zu erreichen, müssen die Programme auch korrekt eingerichtet werden.

6.6 Einbindung von vorhandenen Programmen

Die bisher vorgestellten Programme unterstützen nur einzelne Prozesse, aber sie bieten dann eine umfassende Unterstützung der Prozesse an. Häufig werden aber auch Programme benötigt, welche eine spezifische Aufgabe erfüllen sollen und dann steht nicht die Unterstützung eines Prozesses im Vordergrund. Auch im Szenario werden verschiedene Programme genutzt, welche die Administration bei sehr speziellen Problemen vereinfacht.

Hier werden nun zwei Programme vorgestellt, welche schon im Szenario im Einsatz sind. Dabei können die beiden Programme auch einzelne Aufgaben aus den ITIL-Prozessen übernehmen, so daß eine kurze Evaluierung im Kontext der Prozessunterstützung sinnvoll ist.

6.6.1 Nagios

Die Anforderungen für für das Configuration Management beschreibt auch die Überwachung, ob der Istzustand mit dem Sollzustand übereinstimmt (siehe Anforderung 29 auf Seite 62). Weil die Netzkonfiguration im Szenario sehr komplex ist, ist die manuelle Überprüfung des aktuellen Netzzustandes sehr aufwendig.

Diese Arbeit kann aber teilweise von einer Software übernommen werden und dies erleichtert die Administration erheblich. Für das Szenario ist zum einen das Programm *Nagios*[11] ausgewählt worden. Die Kurzbeschreibung aus der Homepage von *Nagios* zeigt dabei, welche Objekte überwacht werden können.

> *Nagios*®is a host and service monitor designed to inform you of network problems before your clients, end-users or managers do.

Für die Überwachung benötigt *Nagios* auch Informationen über die zu überwachende Objekte. Diese Informationen werden in *Nagios* in Konfigurationsdateien abgelegt. Jedoch benutzt *Nagios* eine eigene Syntax für die Ablage der Informationen, welches folgende Nachteile bringt:

- Zum Konfigurieren des Programms ist eine intensive Einarbeitung in das Programm notwendig.
- Es können nur von erfahrenen Benutzern diese Daten geändert werden.
- Die eingegebenen Daten können nur vom Programm *Nagios* eingelesen und verwendet werden.

Weil die Daten über die Configuration Items nicht über eine Benutzeroberfläche zu verwalten sind, kann das Programm *Nagios* nicht die Aufgabe einer CMDB übernehmen. Für die aufwendige Konfiguration von *Nagios* könnte die Übernahme der betreffenden Configuration Items als Konfiguration sehr viel Arbeit ersparen. Weil dieser Import nicht verfügbar ist, müssen die Daten über die Netztopologie sowohl in die CMDB als auch in die Konfigurationsdatei von *Nagios* eingegeben werden. Durch diese mehrfache manuelle Eingabe kann Fehler auftreten, so daß die CMDB und *Nagios* unterschiedliche Daten erhalten. Dieses negative Beispiel verdeutlicht wieder die enormen Vorteile eines Datenaustausches zwischen verschiedenen Programmen. (siehe dazu die Anforderung 63 auf Seite 69).

Fazit Damit bleibt dem Programm *Nagios* nur die Aufgabe der Überwachung der Configuration Items und kann kein Ersatz für eine CMDB sein. Diese Aufgabe erfüllt es aber ausgezeichnet, so daß der Einsatz das Programms für die Überwachung der komplexen Netzstruktur sinnvoll ist. Leider bleibt die Zusammenarbeit mit anderen Programmen nur oberflächlich, in dem die benutzten URLs von *Nagios* als Referenzen in anderen Programmen eingebunden werden.

6.6.2 NEIS

NEIS ist ein Programm von Harald Rölle, welches erweiterte Informationen zu *Nagios* anbietet. Dabei ist das Wort “NEIS” ein Akronym und steht für “Nagios Extended Information System”.

Das Ziel von *NEIS* ist die Bereitstellung und Verwaltung der vorhandenen Dokumentationen aller Configuration Items. Damit besitzt es eine ähnliche Zielstellung wie eine Dokumentenverwaltungsprogramm.

Das Programm zeigt dabei die vorhandenen Dokumente über das Web an, so daß es von überall aus leicht erreichbar ist. Dabei legt es die Dokumente auf dem Dateisystem in Verzeichnissen geordnet ab. Jedoch verwendet *NEIS* keine Datenbank für die Verwaltung der Metainformationen über die Dokumente, so daß die Suche nach einem Dokument eine sehr intensive Festplattennutzung benötigt und damit bei sehr vielen Dokumenten länger dauern kann (erforderlich für die Anforderung 25 auf Seite 60). *NEIS* wählt wie das Programm *Nagios* für die Verwaltung der Informationen den Weg über einzelne Konfigurationsdateien auf dem Dateisystem. Weil die Konfiguration wie bei *Nagios* eine eigene Syntax besitzt, ist die Weiterverwendung dieser Informationen nur über die Implementierung eines passenden Parsers möglich. Heute steigen deshalb viele Programme auf das XML-Format um, damit auch andere Programme einfacher die gewünschten Informationen extrahieren können.

Die Zusammenarbeit von *NEIS* mit *Nagios* geschieht aus bekannten Gründen über die Verwendung von URLs. Somit können zu jedem Configuration Item in *NEIS* entweder die vorhandene Dokumentation gelesen werden oder den aktuellen Status über *Nagios* abgerufen werden.

[11] *Nagios*: `http://www.nagios.org/`

Fazit Mit der Verwaltung der vorhandenen Dokumentation erfüllt das Programm *NEIS* einen kleinen Teil der Anforderungen einer CMDB, weil auch die Dokumentation unter den Begriff Configuration Items fallen (siehe dazu die Definition 3 auf Seite 9). Damit werden die Anforderungen für eine CMDB einfacher, wenn einerseits ein Dokumentenverwaltungssystem wie *NEIS* eingesetzt werden und andererseits die neue CMDB mit diesem Verwaltungssystem zusammenarbeiten kann (siehe die Anforderung 16 auf Seite 59).

Der Nutzen eines Dokumentenverwaltungssystem ist vorallem bei der Diagnose von Störungen oder Problemen sehr hilfreich, weil so sehr schnell die passende Anleitung gefunden werden kann. Somit bringt der Einsatz von *NEIS* einen sofort sichtbaren Erfolg.

6.7 Empfehlung für den Einsatz im Szenario

Das Ergebnis dieser Evaluierung ist, daß alle vorgestellten Programme für das Szenario sinnvoll sein können. Weil jedoch die Programme unterschiedliche Bereiche der Prozesse unterstützen, soll hier ein kurzer Überblick der Softwareprodukte für das Szenario gegeben werden:

- **Incident Management** kann im Szenario am Besten mit dem Programm *OTRS* unterstützt werden. Jedoch zeigen sich die Stärken von *OTRS* erst bei der Bearbeitung vieler Störungsmeldungen. Falls die Anzahl der Störungsmeldungen gering bleibt, können diese wenigen auch über die vorhandenen Mailinglisten bearbeitet werden.
- **Problem Management** wird von keiner vorgestellten Software optimal unterstützt. Jedoch besitzt die Mailinigliste die höchste Flexibilität und aus diesem Grund soll im Szenario die Probleme über eine Mailingliste bearbeitet werden. Für die Zeitplanung wäre eine gemeinsame Terminplanungssoftware eine ideale Ergänzung.
- **Configuration Management** besitzt sehr spezifische Anforderungen für das gewählte Szenario, welches von keinem bekannten Open Source Programm vollständig unterstützt wird.
- **Change Management** ist ohne eine CMDB nicht sinnvoll und somit ergeben sich ähnliche Probleme für das Szenario wie beim Configuration Management.
- **Release Management** ist für den kleinen Rechnerraum im Szenario eher trivial. Aus diesem Grund reicht für den Anfang die textliche Beschreibung der DSL und DHS aus. Falls höhere Anforderungen sich ergeben, kann als Zwischenschritt das Release Management ähnlich wie das Change Management unterstützt werden.

Zusätzlich profitieren alle Prozesse durch eine gute Kommunikation und Koordination aller Arbeiten zwischen allen Beteiligten. Diese Aufgabe kann allgemein mit dem Einsatz von vorhandenen Officepaketen realisiert werden. Wenn die Mitarbeiter im Lehrstuhl sich auf ein spezielles Officepaket einigen, dann kann der erzielbare Nutzen auch optimiert werden, weil der Austausch von Informationen innerhalb einer Softwareproduktreihe am besten abläuft.

Das Fazit der Evaluierung der vorhandenen Softwareprodukte ergibt aber auch Mängel im Angebot von guter Software für die Unterstützung des Configuration und Change Managements. Die gefundenen Open Source Produkte sind meistens für eine spezifische Einsatzumgebung entwickelt worden. Weil die Anpassung einer vorhandenen Software einen vergleichbaren Aufwand besitzt wie eine vollständige Eigenentwicklung, soll im nächsten Kapitel ein selbstentwickeltes Programm, welches spezifisch für das Szenario entwicklet worden ist, vorgestellt werden.

7 Design und Implementierung

Bei der Suche nach einem Programm für die ITIL-Unterstützung des Szenarios stellt man fest, daß für den Bereich des Configuration und Change-Managements keine geeignete Programme existieren. In diesem Bereich beschränken sich die bekannten Open-Source Produkte auf einen spezifischen Einsatzumgebung und vernachlässigen dabei andere Teile vollständig. Häufig wird deshalb nur das Verwalten der Netzkonfiguration unterstützt, welches für die angebotenen Praktika sehr wichtig ist, aber nicht ausreichend für eine vollwertige CMDB ist. Dabei vernachlässigen die vorhandenen Programme häufig die Verwaltung der installierten Programme und ihre Einstellungen.

Bei einer kurzen Betrachtung von kommerziellen Softwareprodukten ergibt sich ein anderes Problem: Die kommerziellen Produkte sind für große Rechenzentren oder für viele Computer ausgelegt. Damit die Produkte diese Aufgabe auch bewältigen können, benötigen sie eine umfangreiche Konfiguration. Damit scheiden die meisten kommerzielle Produkte aus Komplexitätsgründen aus, weil der Mehraufwand zur Einrichtung und Wartung der neuen Software den Nutzen vollständig aufheben würde. Und weil der Nutzen für das Szenario bei kommerziellen Produkten nicht absehbar ist, sind eventuell hohe Lizenzkosten für das Szenario nicht gerechtfertigt.

Aus diesem Grund bietet sich für dieses Szenario eine Eigenentwicklung an, weil damit die spezifischen Bedürfnisse am besten befriedigt werden können.

7.1 Design

Die Softwareunterstützung des Configuration und Change-Managements soll so flexibel sein, daß es in unterschiedlichen Einsatzumgebungen einsatzbar ist. Zusätzlich ist die Flexibilität notwendig, damit auch Änderungen innerhalb der Umgebung in der Software abbildbar sind. Die notwendige Flexibilität kann man durch eine Abstraktion auf die Gemeinsamkeiten aller Configuration Items erreichen.

7.1.1 Ziel

Das Ziel der Software ist die Unterstützung der beiden ITIL Prozesse Configuration und Change-Management für das gegebene Szenario, weil für diese beiden Prozesse keine befriegenden Open Source Programme gefunden worden sind (siehe dazu den Abschnitt 6.7 auf der vorherigen Seite). Dabei sollen die Anforderungen aus Kapitel 5 auf Seite 54 für die beiden Prozesse so weit wie möglich umgesetzt werden. Mit dem Einsatz der neuen Software soll der Arbeitsaufwand für die Administration gemindert werden. Da die restlichen ITIL Prozesse durch den Einsatz von fertigen Softwareprodukten unterstützt werden, soll die neue Software die Integration der fremden Produkte ermöglichen.

Weil das neue Programm nicht alle Anforderungen alleine erfüllen kann, soll es deshalb mit den folgenden Produkten zusammenarbeiten:

- die Terminkalender erhalten ihre Daten über den iCal Standard (RFC2445)
- die Zusammenarbeit mit *OTRS* als Incident Management erfolgt über die Verwendung von URLs.
- ebenfalls sollen Programme *NEIS* und *NAGIOS* über den Einsatz von URLs verknüpft werden.
- die Kommunikation zwischen den Anwendern soll über eMail und Telefon gefördert werden.

Ein wichtiges Ziel dieser Diplomarbeit ist die Optimierung der Arbeitsabläufe der Administration des Rechnerraums. Dies bedeutet aber auch, daß die Benutzung der Software die Administration erleichtern und nicht behindern soll. Aus diesem Grund ist die Benutzerführung sehr offen gestaltet, d.h. der Benutzer soll selber bestimmen können, in welcher Reihenfolgen er seine Arbeit durchführen möchte (erfüllt damit die Anforderung 57 auf Seite 68). Diese Auffassung der Benutzerführung steht dann natürlich im Widerspruch zur Prozessorientierung der Administration. Jedoch sollen die Benutzer nicht in ihrer Arbeit eingeschränkt werden und deshalb gibt das Programm ihnen die Freiheit, sich auch nicht an die vorgeschlagenen Prozesse zu halten. Somit bleibt die Umsetzung der Prozesse in den Händen der Benutzer, d.h. die Personen sind für die Einhaltung der Prozessvorgaben selbst verantwortlich.

7.1.2 Wahl der Plattform

Für einen effizienten Einsatz einer CMDB muß die Bedienung von allen Arbeitsplätzen im Lehrstuhl möglich sein. Weil der Lehrstuhl sehr unterschiedliche Rechnerarchitekturen einsetzt, war die Entscheidung einer Webanwendung von Anfang an klar. Dies bedeutet, daß die Software auf einem Webserver läuft und die Anwender dieses Programm über ihren Webbrowser bedienen können.

Technologieentscheidung Für die Implementierung einer CMDB standen zwei mögliche Technologien zur Frage, wie die Daten verwaltet werden sollen. Weil die Entscheidung auf eine Technologie nicht eindeutig war, soll hier kurz die Alternative vorgestellt werden.

Eine Möglichkeit wäre die Verwendung eines klassischen relationalen Datenbankensystems, welche heute schon sehr ausgereift sind. Ein weiterer Vorteil für die Datenbanken ist die heute große Auswahl der angebotenen Programme. Jedoch zwingt der Einsatz einer relationalen Datenbank eine strikte Festlegung des Datenbankschemas, damit keine Updateanomalien auftreten können. Am einfachsten wäre es also, wenn beim Entwurf des Datenbankschemas alle Details der Configuration Items bekannt sind. Weil aber diese Eigenschaften beim Datenbankentwurf nicht bekannt sind, ergeben sich Probleme bei der Definition eines sinnvollen Datenbankschemas.

Die Alternative ist die Verwendung des neuen Datenformats XML[1], welche momentan aktiv in vielen Bereichen eingesetzt wird. Der Vorteil von XML ist die Lösung des Problems, welche relationale Datenbanken besitzen: Das Schema von XML wird selbst in XML beschrieben, im sogenannten XML-Schema[2]. Somit ermöglicht XML die Modellierung mehrerer Abstraktionsebenen:

- **XML** beschreibt die grundlegende Grammatik einer XML Datei
- **XML-Schema** beschreibt die Syntax und die Semantik einer XML-Schema-Beschreibung
- **CI-Schema** kann die allgemeine Struktur eines Configuration Items beschreiben
- **CI** kann dann die einzelnen Configuration Items beschreiben und instanziert damit das CI-Schema

Jedoch ist die Beschreibung der Beziehungen zwischen verschiedenen XML-Dokumenten noch nicht weit verbreitet, obwohl es einen Standard für die Syntax dafür schon verabschiedet worden ist (siehe XML Pointer[3]). Das geringe Interesse an diesem Standard erkennt man beispielsweise im Fehlen von Klassen im *Java Developement Kit* von Sun Microsystems oder im *.Net Framework* von Microsoft. Weil jedoch der wesentliche Unterschied einer Inventurdatenbank und einer CMDB die Beziehungen zwischen den Configuration Items ist, ist dieser Punkt eine erhebliche Einschränkung für den Einsatz von XML.

Am Ende ist die Entscheidung auf relationale Datenbank gefallen, weil diese Technologie ausgereifter und weiter verbreitet ist. Jedoch kann diese Entscheidung in wenigen Jahren anders ausfallen, wenn Microsoft

[1] neuen Datenformats XML: `http://www.w3c.org/XML/`
[2] XML-Schema: `http://www.w3c.org/XML/Schema`
[3] XML Pointer: `http://www.w3c.org/XML/Linking`

WinFS[4] veröffentlicht. *Microsofts WinFS* soll ein Framework für die Verwaltung aller vorhanden Informationen mit XML sein. Wenn alle Visionen dieses Framework auch realisiert werden können, dann könnte die Realisierung einer CMDB sehr einfach mit den angebotenen Diensten des *WinFS* möglich sein.

Entscheidung über die verwendete Software Für eine schnelle Entwicklung eines Prototypen ist die Softwareauswahl LAMP (*Linux*, *Apache*, *MySQL*, *PHP*) ausgewählt worden. Diese Auswahl ist bekannt und sehr verbreitet, so daß sich dieser Begriff eingebürgert hat. Wegen dem hohen Bekanntheitsgrad bieten viele Linuxdistributionen diese Plattform vorkonfiguriert an. Zusätzlich erbt die Programmiersprache PHP von Perl[5] eine sehr gute Stringverarbeitung, die die Verarbeitung von HTML stark vereinfacht.

Der Vorteil der schnellen Entwicklungszeit mit LAMP bringt aber auch Nachteile:

- *PHP* ist eine Skriptsprache mit einem schwachen Typsystem und damit treten viele Fehler erst zur Laufzeit auf.
- *PHP* besitzt nur eine rudimentäre Unterstützung für Objektorientierung und erschwert damit die Erweiterbarkeit.
- *MySQL* unterstützt ANSI SQL 99 nicht vollständig.

Folgende Versionen für die Entwicklung wurden verwendet:

Software	verwendete Version	
	Fedora Core 2	*Gentoo Linux 2004.3*
Apache	2.0.51	2.0.52
MySQL	3.23.58	4.0.22
PHP	4.3.8	4.3.10

7.1.3 Datenmodell

Das Configuration und Change Management benötigen beide die Informationen über die abgelegten Configuration Items. Und diese Informationen werden alle in der CMDB gespeichert. Die Benutzung einer Datenbank kann man wie folgt charakterisieren:

- **Eingabe** die Daten aller Configuration Items müssen zuerst eingegeben werden, bevor sie später von anderen abgerufen werden kann.
- **Suchen** Der Vorteil einer elektronischen Datenbank ist die einfache und schnelle Suche über den gesamten Datenbestand (vgl. die Anforderung 25 auf Seite 60).
- **Ausgabe** Und nachdem die gewünschten Daten gefunden sind, müssen die Informationen auch wieder ausgegeben werden.

Und diese drei Funktionen muß die Software mindestens erfüllen, damit der Einsatz als CMDB sinnvoll ist. Jedoch reicht diese triviale Beschreibung nicht für einen effektiven Einsatz der Software aus, so daß nun die weiteren Eigenschaften an die CMDB und ihr Datenmodell beschrieben werden soll.

Beschreibung von Configuration Items Die wichtigste Aufgabe einer CMDB ist die Verwaltung der Configuration Items. Und dazu müssen die Configuration Item erstmal beschrieben werden. (siehe dazu die Anforderung 18 auf Seite 59). Weil aber viele Configuration Items unterschiedliche Strukturen besitzen, muß die Software flexibel an die unterschiedlichen Strukturen anpassbar sein. Ein wesentlicher Nachteil vieler Open Source Programme ist die statische Vorgabe der unterstützen Configuration Items, und somit können diese Programme nicht an die jeweilige Situation angepasst werden. Aus diesem Grund soll das verwendete Datenmodell die Configuration Items über eine Liste von typisierten Attributen beschreiben

[4]Microsoft WinFS: `http://longhorn.msdn.microsoft.com/lhsdk/winfs/conoverviewofwinfs.aspx`
[5]Perl: `http://www.perl.org/`

können (siehe dazu die Anforderung 18 auf Seite 59). Die notwendigen Strukturen eines Configuration Items werden deshalb in sogenannten CI Schema beschrieben (siehe die Anforderung 17 auf Seite 59).

Eine weitere Anforderung an das Datenmodell ist der ständige Verfügbarkeit alter Versionen (siehe dazu den gleichnamigen Absatz 5.3.2 auf Seite 66). Diese Forderung wird im Datenmodell mit einer trivialen Versionsverwaltung gelöst, in dem jede Änderung eines Configuration Items eine neue Version (siehe dazu die Tabelle "civersion" im Abschnitt B.1 auf Seite 113) anlegt. Damit werden keine Informationen überschrieben, sondern bei jeder Änderung wird eine neue Kopie mit den geänderten Daten angelegt. Somit bleiben jederzeit alle alten Versionen des Configuration Items erhalten und für den Benutzer einsehbar.

Für die Unterstützung des Change Managements benötigen jedoch alle Änderungen eine kurze Begründung, so daß auch später die Gründe für die Änderung nachvollzogen kann. Für eine vollständige Beschreibung einer Änderung werden dann automatisch das Änderungsdatum und der Name des Autors eingetragen.

Zuletzt besitzt jedes Configuration Item einen Status und ist Teil eines Profils (siehe dazu die Anforderung 21 und 27 auf Seite 61). Deshalb müssen diese beiden Informationen ebenfalls im Datenmodell berücksichtigt werden.

Benutzerverwaltung Weil auf die CMDB mehrere Personen gleichzeitig arbeiten können und die Benutzer dabei unterschiedliche Rechte besitzen, müssen in der Datenbank auch die Informationen über die Benutzer und ihre Rechte gespeichert werden. In der Anforderung 48 auf Seite 65 wird dazu ein Rollenmodell beschrieben und dieses Modell soll die Software auch unterstützen. Deshalb erhalten die Benutzer eine Menge von Rollen und zu jeder Rolle sind dann die notwendigen Berechtigungen definiert.

Damit dieses Rollenmodell auch wirksam sein kann, müssen sich die Benutzer auch authentifizieren (siehe die Anforderung 47 auf Seite 65). Die bekannteste und einfachste Authentifikationsmethode für eine Webanwendung ist die Überprüfung eines geheimen Passworts.

Fazit Aus den verschiedenen Anforderungen ergeben sich weitere Details für das zu verwendete Datenmodell. Das daraus resultierende Datenbankschema wird später im Abschnitt B auf Seite 111 im Detail beschrieben.

7.2 Implementierung

Die Implementierung soll die entwickelten Ideen demonstrieren, so daß der Nutzen beim Einsatz eines Configuration und Change-Managements auch im gewählten Szenario sichtbar werden. Jedoch kann aus Zeitgründen diese Implementierung nur ein Prototyp entwickelt werden.

Da jede Software zur besseren Identifikation auch einen Namen besitzen soll, ist der Name dieses Prototypen *ConfDB* und ist die Abkürzung für "Configuration Database". Der Name soll die Aufgabe als CMDB verdeutlichen.

7.2.1 Benutzeranleitung

Ein wichtiger Bestandteil jeder Software ist die Anleitung für die Benutzer, damit die Bedienung und die Arbeitsweise der Software von den Anwendern verstanden wird. Dabei kann eine integrierte Hilfe im Programm die Bedienung erleichtern, weil sie jederzeit und schnell verfügbar ist. Jedoch ist das Lesen längere Texte an einem Bildschirm nicht angenehm, so daß für die Beschreibung der komplexeren Zusammenhänge eine Dokumentation auf Papier vorzuziehen ist.

Installation Es wird angenommen, daß die Software für die Plattform (siehe Abschnitt 7.1.2 auf Seite 83) schon erfolgreich und korrekt installiert und eingerichtet ist.

Als erstes muß dann das Softwarepaket in einem neuen Dokumentenverzeichnis des Webservers entpackt werden. Der Webserver muss so konfiguriert sein, daß er Endungen des Dateinamens mit .php als *PHP*-Dateien interpretiert.

Dann müssen in der Konfigurationsdatei includes/config.php die Parameter[6] für die *MySQL*-Datenbank mitgeteilt werden. Mit den eingegebenen Werten muß dann *PHP* eine Verbindung erfolgreich aufbauen können.

Der nächste Schritt ist die Einrichtung der Datenbank. Dazu müssen zuerst die notwendigen Tabellen in der Datenbank erzeugt werden. Dies kann durch das Ausführen der Datei install/createtables.sql geschehen, welche die korrekten SQL-Befehle zur Erstellung der Tabellen enthält. Mit der Erstellung der Tabellen ist das Programm *ConfDB* erfolgreich installiert.

Benutzer einrichten Jedoch fehlt zum Benutzen des Programms noch das Erstellen eines Benutzers. Dies kann entweder manuell durch direkte Eingabe der SQL-Befehle geschehen oder man führt das beigefügte Skript install/createadmin.sql aus, welches die notwendigen SQL-Befehle enthält. Dieses Skript erstellt in der Datenbank einen Benutzer "Admin" mit allen Rechten. Das Passwort für diesen Benutzer steht ebenfalls in diesem Skript, aber aus Sicherheitsgründen soll dieses vor dem Ausführen zu ein sicheres Passwort geändert werden.

Nach der Erstellung eines Benutzers mit allen Rechten kann die weitere Konfiguration von *ConfDB* über die Weboberfläche durchgeführt werden. Dazu muß man zuerst die Rollen erstellen, indem man den Menüpunkt Add Role im Programm auswählt. Darauf zeigt es ein Formular an, welches nun ausgefüllt werden muß.

Für jede Rolle müssen dann die notwendigen Berechtigungen festgelegt werden. Das Setzen der Berechtigungen erfolgt über den Menüpunkt Set Permissions for Role. Dort wählt man zuerst die zu bearbeitende Rolle aus und kann dann für die verschiedene Objekte die Berechtigungen setzen. Der Aufbau der Berechtigungen ist dabei dreigeteilt. Zuerst stehen die Namen der verschiedenen Objekte, auf die die Berechtigungen anzuwenden sind. Momentan unterstützt das Programm das Setzen der Berechtigung für folgende Objekte:

- profile
- schema
- ci
- user
- role
- task

Jedes Objekt besitzt dann vier mögliche Operationen, die für den Benutzer erlaubt oder verboten sind. Die vier Operationen lauten wie folgt:

- Create
- Read
- Write
- Delete

Zu jedem dieser Operationen bietet dann das Programm die Auswahl zwischen Allow oder Deny an.

[6]Für eine Datenbankverbindung ist der Rechnername, der Benutzername, das dazugehörige Passwort und der Name der Datenbank notwendig

Arbeitsweise der Berechtigungen Bis jetzt ist aber noch nicht erklärt, wie diese Berechtigungen im Programm wirken. Und deshalb soll nun kurze die Arbeitsweise beschrieben werden.

Falls keine Berechtigung für ein Objekt in der Datenbank zu einer Rolle definiert ist, dann ist die Vorgabe von *ConfDB* das Ablehnen des Zugriffes. Wenn ein Benutzer mehrere Rollen besitzt, dann erhält der Benutzer Zugriff auf ein Objekt, wenn dieser Zugriff mindestens in einer seiner Rolle erlaubt ist.

Ein Beispiel soll dies verdeutlichen: Ein Anwender U besitzt zwei Rollen A und B. Die Rolle A erlaubt nur den Zugriff auf die Profile und die Rolle B erlaubt nur den Zugriff auf die Rollen. Dann besitzt der Anwender Zugriff auf die Profile und auf die Rollen.

Damit der neu angelegte Benutzer auch Zugriff auf die Daten erhält, muß nun dem Benutzer seine Rollen zugeordet werden. Dies kann man über den Menüpunkt Change User Roles bewerkstelligen.

ConfDB einrichten Das Programm *ConfDB* bietet eine sehr flexible CMDB an und deshalb benötigt die Software eine Reihe von Einstellungen, damit sie die Strukturen der Configuration kennt.

Zuerst sollte man die möglichen Zustände von Configuration Items definieren (siehe dazu die Anforderung 21 auf Seite 59). Dies geschieht durch den Menüpunkt Add States. Als wichtigeste Eigenschaft eines Zustandes ist der Name, der später bei jedem Configuration Item angezeigt wird. Die beiden weiteren Eigenschaften Is Obsolete und Is Damaged soll dem Programm *ConfDB* die Bedeutung dieses neuen Zustandes erklären. Is Obsolete bedeutet, daß Configuration Items in diesem Zustand nicht mehr vorhanden sind oder nie wieder einsatzbereit sein werden. Is Damaged bedeutet einfach ein defektes Configuration Item, welches aber wieder für den Einsatz reparariert werden kann.

Weil in einer CMDB potentiell sehr viele Configuration Items abgelegt werden können, ermöglicht das Programm *ConfDB* die Configuration Items in Profilen zusammenzufassen (vgl. die Anforderung 27 auf Seite 61). Ein Configuration Item kann dabei nur jeweils genau in einem Profil liegen. Diese Einschränkung soll die Verwaltung der Configuration Items und ihre Profile erleichteren. Mit dem Menüpunkt Add Profile können deshalb neue Profile angelegt werden. Mit den Profilen kann auch eine primitive Unterstützung für des Release Managements sich bewerkstelligen, in dem man zwei Profile mit den Namen “DSL” und “DHS” erstellt (vgl. dazu die Anforderung 45 auf Seite 65). In diese beiden Profile können dann die Informationen der beiden Policen als Configuration Items abgelegt werden.

Bis jetzt sind aber nur kleine Details einer CMDB eingerichtet worden. Ein Kernproblem bei einer CMDB ist das Festlegen, welche Configuration Items mit welchem Detailierungsgrad in der CMDB abgelegt werden sollen (siehe dazu den Abschnitt **??** auf Seite ??). Nachdem diese beiden Fragen geklärt sind, kann man für die gewünschten Configuration Items passende Schemas anlegen (siehe dazu die Anforderung 17 auf Seite 59). In einem Schema sind dann die Attribute beschrieben, die ein Configuration Item auszeichnet. Zum Erstellen eines neuen Schemas benutzt man den Menüpunkt Add Schema. Dann wird man nach einem Namen, einer Beschreibung und die Anzahl der frei wählbaren Attribute gefragt. Und darauf muß man die genauen Eigenschaften[7] der Attribute genauer spezifizieren. Nach dem Anlegen der Schemas ist das Programm *ConfDB* für Dateneingabe der Configuration Items bereit.

Configuration Items eingeben Nachdem alle Einstellungen für die Eingabe der Configuration Items abgeschlossen ist, können nun die Informationen der Configuration Items in die Datenbank eingegeben werden (erfüllt damit die Anforderung 19 auf Seite 59).

Die Eingabe geschieht über den Menüpunkt Add CI und dann präsentiert das Programm eine Auswahl der definierten Schemas. Jetzt muß man das Schema für das neue Configuration Item auswählen. Dann fordert das Programm zur Eingabe aller Attributwerte des neuen Configuration Items auf. Dabei unterscheidet das Programm zwischen den allgemeinen Attributen, welche jedes Configuration Items besitzen muß, und vom Benutzer definierte Attribute, welche im Schema deklariert sind.

[7] Jedes Attribut benötigt einen Namen, ein Typ und die Markierung, ob dieses Attribut zwingend für die Beschreibung des Configuration Items erforderlich ist.

Zu den allgemeinen Attributen jedes Configuration Items gehören folgende Informationen:

- **Schema** Dieses Configuration Item gehorcht diesem Schema.
- **Author** beschreibt die Person, welche diese Informationen eingegeben hat.
- **Profile** Das Configuration Item gehört zu diesem Profil.
- **State** Und es befindet sich in dem Zustand.
- **Name** Jedes Configuration Item besitzt neben seiner ID auch einen frei wählbaren Namen.

In der Spalte Type wird der Name des Attributtyps angezeigt. Dabei bedeutet ein fett gedruckter Attributtyp, daß dieses Attribut zwingend erforderlich für dieses Configuration Item ist. Nach der Eingabe der Daten kann mit dem Knopf Create das Configuration Item abgespeichert werden. Zur Bestätigung für die erfolgreiche Eingabe zeigt das Programm die CI ID und den gewählten Namen. Idealerweise beschriftet man die Hardware mit diesen beiden Informationen, so daß ein enger Zusammenhang zwischen der realer Welt und dem Configuration Item erhalten bleibt.

Eingabe der Beziehungen zwischen den Configuration Items Nachdem die Daten aller Configuration Items eingegeben sind, müssen für eine sinnvolle Nutzung auch die Beziehungen zwischen den Configuration Items definiert werden (erfüllt damit die Anforderung 23 auf Seite 60).

Eine neue Beziehung kann mit dem Menüpunkt Relate 2 CIs erstellt werden. Dann fordert das Programm den Benutzer auf, die Beziehung zu spezifizieren. Die wichtigste Information einer Beziehung sind die beiden Configuration Items, welche in eine Beziehung gebracht werden sollen. Dazu benötigt das Programm von beiden die CI ID. Damit auch das Programm die Bedeutung dieser neuen Beziehung verstehen kann, kann der Benutzer aus folgender Liste die Bedeutung festlegen:

- "A" depends on "B"
- "A" is part of "B"
- "A" is connected with "B"

Dabei repräsentieren die Buchstaben "A" und "B" jeweils die beiden Configuration Items. Für weitere Erklärungen der Bedeutung und die Gründe für diese Beziehung steht noch ein Texteingabefeld für einen Kommentar zur Verfügung. Nach der Bestätigung der Eingabe wird die Beziehung in die Datenbank eingetragen.

Configuration Items suchen Nachdem die Configuration Items mit ihren Beziehungen eingegeben sind, kann der Benutzer auch die Datenbank abfragen. Um ein spezifisches Configuration Item zu suchen bietet das Programm den Menüpunkt Search CIs an (erfüllt die Anforderung 25 auf Seite 60). Auch hier frägt das Programm zuerst nach dem Schema des zu suchenden Configuration Items ab. Nachdem der Benutzer das Schema ausgewählt hat, kann es nach bestimmten Attributwerten suchen. In der Spalte Operator kann der Benutzer dabei den Vergleichsoperator festlegen. Dabei bedeutet der Vergleichsoperator any, daß jeder Wert dieses Attributes zulässig ist. Die Auswahl der vorhandenen Vergleichsoperatoren hängt dabei vom Attributstyp ab. Mit dieser Methode können mehrere Bedingungen vom Benutzer definiert werden. Falls keine Bedingungen definiert sind, d.h. alle Vergleichsoperatoren sind any, dann liefert das Programm eine Liste aller Configuration Items des gewählten Schemas. Wenn hingegen mehrere Bedingungen definiert sind, dann müssen die Configuration Items alle gleichzeitig erfüllen (Dies entspricht einer Bool'schen AND-Verknüpfung der einzelnen Kriterien).

Bei der Ansicht jedes Configuration Items existiert ein Menüpunkt View Relations of this CI, welche alle Beziehungen des gewählten Configuration Items anzeigt. Damit können die Beziehungen zur Navigation zwischen den Configuration Items benutzt werden.

Als Beispiel für eine automatische Auswertung der Beziehungen und der Zustände ist der Menüpunkt Detect Problems. Mit diesem Menüpunkt stellt das Programm eine Liste von defekten Configuration Items zusammen. Zusätzlich werden auch alle Configuration Items hinzugefügt, die direkt oder indirekt von einem defekten Configuration Item abhängen (mit der Beziehung depends on). Diese Liste der Configuration Items zeigt dem Administrator die möglichen Auswirkungen eines Defekts und er kann damit auch viel besser die Auswirkungen abschätzen.

Diese Aktion zeigt auch eine sinnvolle Implementierung für die Anforderung 26 auf Seite 61, welche die Bildung einer transitiven Hülle beschreibt. Ohne diese Hülle fände das Programm nur die defekten Configuration Items und die davon direkt abhängigen. So können auch die Configuration Items gefunden werden, welche nicht direkt von einem defekten abhängen.

Configuration Items ändern Wenn eine Änderung von einem Configuration Item notwendig ist, dann kann der Benutzer mit dem Menüpunkt Change CI diese Änderung durchführen. Zuerst muß der Benutzer die ID des zu ändernden Configuration Item angeben. Und danach zeigt das Programm ein Formular zur Eingabe einer Änderung an. Im ersten Teil des Formulars mit der Überschrift Change Information können die allgemeinen Information zu dieser Änderung beschrieben werden. Dazu gehören folgende Daten:

- **Priority** beschreibt die Priorität der Änderung.
- **Execution Time** legt den Zeitpunkt für die Durchführung dieser Änderung fest.
- **Comment** ermöglicht eine Beschreibung der Änderung.
- **Finished** zeigt an, ob diese Änderung schon vollständig durchgeführt worden ist. Falls hier False ausgewählt worden ist, kann diese Änderung im Rahmen eines Tasks koordiniert werden.

Im zweiten Abschnitt mit der Überschrift CI Changes ermöglicht das Programm die Attributwerte des gewählten Configuration Items zu ändern. Dabei besitzt das Attribut Active eine besondere Bedeutung. Weil jedes Configuration Item mehrere Versionen besitzen kann, wird mit dem Attribut Active festgelegt, welche Version gerade aktuell ist. Die aktive Version zeigt somit den aktuellen Stand des Configuration Items an. Alle inaktiven Versionen sind deshalb aktuell nicht gültig, sondern dokumentieren die historischen Werte oder zeigen die zukünftigen und in der Planung befindlichen Werte an.

Es kann für jedes Configuration Item nur eine aktive Version existieren, welche den aktuellen Zustand dieses Configuration Items beschreibt. Alle inaktive Versionen bleiben nur zum Zweck der Dokumentation der Änderungsgeschichte erhalten.

Weil häufig eine Menge von Änderungen einen gemeinsamen Grund, beispielsweise die Behebung einer Störung oder eines Problems, besitzen, können mehrere offene Änderungen innerhalb eines Tasks koordiniert werden. Dazu erstellt man mit dem Menüpunkt Add Task einen neuen Task. Nach der allgemeinen Beschreibung des Tasks können dann die vorhandenen und offenen Änderungen mit aufgenommen werden. Weil Tasks auch über das iCal-Format exportiert werden können, ermöglicht *ConfDB* damit ein gute Koordination der Tasks innerhalb eines Terminkalenderprogrammes.

iCal Export Der angesprochene iCal-Export ermöglicht die Übernahme der Planungsinformationen in den Terminplan der Benutzer. Der Export wird über eine eigene URL angeboten, welche dann entweder die Tasks aller Benutzer oder eines bestimmten Benutzers im iCal-Format anzeigt. Damit die Terminkalender regelmäßig mit dem Export auf den aktuellen Stand gebracht werden, muß der Export sehr einfach und schnell geschehen. Aus diesem Grund benötigt der Export keine Authentifizierung des Benutzers und somit sind alle Tasks öffentlich. Dies hat auch den Vorteil, daß jeder auch die Tasks der anderen ohne Probleme einsehen kann.

Die URL für den Export lautet `http://$HOSTNAME/$CONFDBPATH/icaltasks.php`. Dabei stehen die Variablen[8] für die spezifischen Teile der Adresse. Desweiteren kann die Ausgabe über die folgende Variablen gesteuert werden:

[8] Die Namen der Variablen beginnen mit dem $-Zeichen.

- **html** Falls diese Variable gesetzt ist, wird der MIME-Type auf text/html geändert. Normalerweise wird der Export mit dem MIME-Type text/calendar ausgeliefert.
- **user** Ohne diese Variable werden die Tasks aller Benutzer exportiert. Und mit dieser Variable können die Tasks eines bestimmten Benutzers ausgewählt werden.

Diese Variablen werden in der üblichen Form in der URL übergeben[9]. Jedoch wird diese Variablenübergabe am einfachsten mit ein paar Beispielen erklärt:

- `icaltasks.php` exportiert alle Tasks mit dem text/calendar-MIME-Type.
- `icaltasks.php?html` exportiert die Tasks im text/html-MIME-Type.
- `icaltasks.php?user=admin` exportiert nur die Tasks des Benutzers “admin”
- `icaltasks.php?html&user=admin` exportiert die Tasks des Benutzers “admin” im HTML-Modus an.

Die exportierten Tasks können dann in jedem Terminkalender mit iCal-Unterstützung ohne Probleme importiert werden.

7.2.2 Einsatz von ConfDB im Szenario

Weil das Programm *ConfDB* speziell für das Szenario entwickelt worden ist, soll eine Grundeinstellung für das Szenario vorgestellt werden. Diese Beispielkonfiguration kann auch für andere Einsatzumgebungen als Grundlage benutzt werden, so daß weniger Arbeit bei der Konfiguration benötigt wird.

Vorschlag für die Einrichtung Als erstes müssen die verschiedenen Rollen definiert werden. Für das Szenario kann folgende Rollenverteilung sinnvoll sein:

- **Administrator** besitzt alle Rechte in *ConfDB*
- **Betreuer** besitzt alle Rechte über die CIs und Tasks und Leserecht über die Schemas
- **Tutor** besitzt das Leserecht über die CIs und Schemas und alle Rechte über die Tasks

Als nächstes können die möglichen Zustände definiert werden, in denen sich die Configuration Items befinden können. In der Tabelle 7.1 beschreibt eine sinnvolle Vorgabe für die Zustände. Diese Vorgabe orientiert sich dabei an den Empfehlungen von itSMF (siehe [BKP 02]).

Zustand	Beschreibung	Optionen
im Einsatz	CI wird gerade produktiv benutzt	
defekt	CI ist fehlerhaft und auser Betrieb	Is Damaged
entfernt	CI ist verkauft, entsorgt oder nicht mehr vorhanden	Is Obselete
in Reparatur	CI ist defekt und wird gerade repariert	Is Damaged
geplant	CI soll in der nächsten Zeit beschafft werden	
auf Lager	CI liegt im Lager und ist jederzeit einsatzbereit	

Tabelle 7.1: ConfDB Vorgabe der Zustände für das Szenario

Für den Einsatz von *ConfDB* sind unterschiedliche Profile notwendig, weil im Szenario die Computer für die beiden Praktika unterschiedlich konfiguriert werden (siehe den Abschnitt 3.2.2 auf Seite 17). Mit den Profilen in der Tabelle 7.2 auf der nächsten Seite kann die Menge der Configuration Items aus dem Szenario sinnvoll geordnet werden.

[9]Die Details für den Aufbau der URL stehen in der RFC1738

Profil	Beschreibung
Hardware	eingesetzte Hardware im Rechnerraum für RNP und SecP
RNP Software	eingesetzte Software für das RNP
RNP Konfiguration	Softwarekonfiguration für RNP
RNP Netzkonfiguration	Netzkonfiguration für RNP
SecP Software	eingesetzte Software für das SecP
SecP Konfiguration	Softwarekonfiguration für SecP
SecP Netzkonfiguration	Netzkonfiguration für SecP
DHS	Vorgaben für den Einsatz von Hardware
RNP DSL	genehmigte Software für den Einsatz im RNP
SecP DSL	genehmigte Software für den Einsatz im SecP

Tabelle 7.2: ConfDB Vorgabe der Profile für das Szenario

In beiden Praktika werden im Laufe der Durchführung die Konfiguration des Rechnerraums an die Aufgaben entsprechend angepaßt. Um diese zeitliche Änderungen des Rechnerraums besser zu dokumentieren, können für jedes Aufgabenblatt ein weiteres Profil erstellt werden.

Ausgangskonfiguration Bevor der Nutzen einer CMDB sichtbar werden, müssen erst alle Daten der Configuration Items in die Datenbank eingetragen werden. Dazu müssen zuerst die Schemas für die Configuration Items definiert werden. Der gewählte Detailierungsgrad bei den Schemas bestimmt dann wesentlich den Aufwand für die Eingabe der Configuration Items und den möglichen Nutzen.

Nachdem alle Schemas definiert sind, können alle Configuration Items in das Programm eingegeben werden. Dabei empfiehlt es sich bei allen Hardware Configuration Items, die erzeugten CI IDs auf die Hardware beispielsweise mit einem Klebestreifen zu dokumentieren.

Nach der Eingabe der Configuration Items müssen auch alle Beziehungen eingegeben werden, damit später auch die Zusammenhänge zwischen den Configuration Items sichtbar werden. Für die Fehlersuche sind dabei die Abhängigkeitsbeziehungen (in *ConfDB* mit der Eigenschaft depends on) am wertvollsten, weil so die Auswirkungen beim Ausfall eines Configuration Items automatisch ermittelt werden kann (siehe dazu die Beschreibung im Abschnitt 7.2.1 auf Seite 88).

Beispiel für ConfDB anhand des Computers pcrnp10 Die Vorteile der Software *ConfDB* kann man anhand eines Beispiels einfacher verdeutlichen. Deshalb soll aus diesem Grund ein kleiner Teil der Rechnerkonfiguration des Computer pcrnp10 (vgl. dazu Abschnitt 3.3.4 auf Seite 22) mit den angebotenen Mitteln von *ConfDB* dargestellt werden.

In der Abbildung 7.1 auf der nächsten Seite erkennt man einen Ausschnitt der Netzkonfiguration des Computers pcrnp10. Dabei stellen die Rechtecke die einzelnen Configuration Items und die Pfeile die Beziehungen in *ConfDB* dar. Dieses Beispiel benutzt für die Configuration Items die folgenden drei Schemas:

- **Computer** faßt alle Komponenten eines Computers zusammen. Und für die einfache Lokalisierung des Computers wird bei diesem Schema auch der Raum angegeben, in dem der Computer steht.
- **Ethernet-NIC** erfaßt alle Daten einer Ethernet-Netzwerkkarte. Insbesondere ist für das Netzmanagement die MAC-Adresse sehr nützlich. Die Angabe des verwendeten Netzwerkchips erleichtert die Installation der Netzwerkkarte bei den verschiedenen Betriebssystemen.
- **IP-Adresse** zeigt die IP-Adresse einer Netzwerkkarte. Und damit die Netzadresse von der Hostadresse unterschieden werden kann, wird auch noch die Netzmaske angegeben.

An diesem einfachen Beispiel erkennt man auch die einfache Datenverwaltung der Configuration Items, weil man sich auf die wesentlichen Eigenschaften beschränken kann. Falls jedoch mehr Daten zur Beschreibung notwendig sind, können einfach detailliertere Schemas angelegt werden.

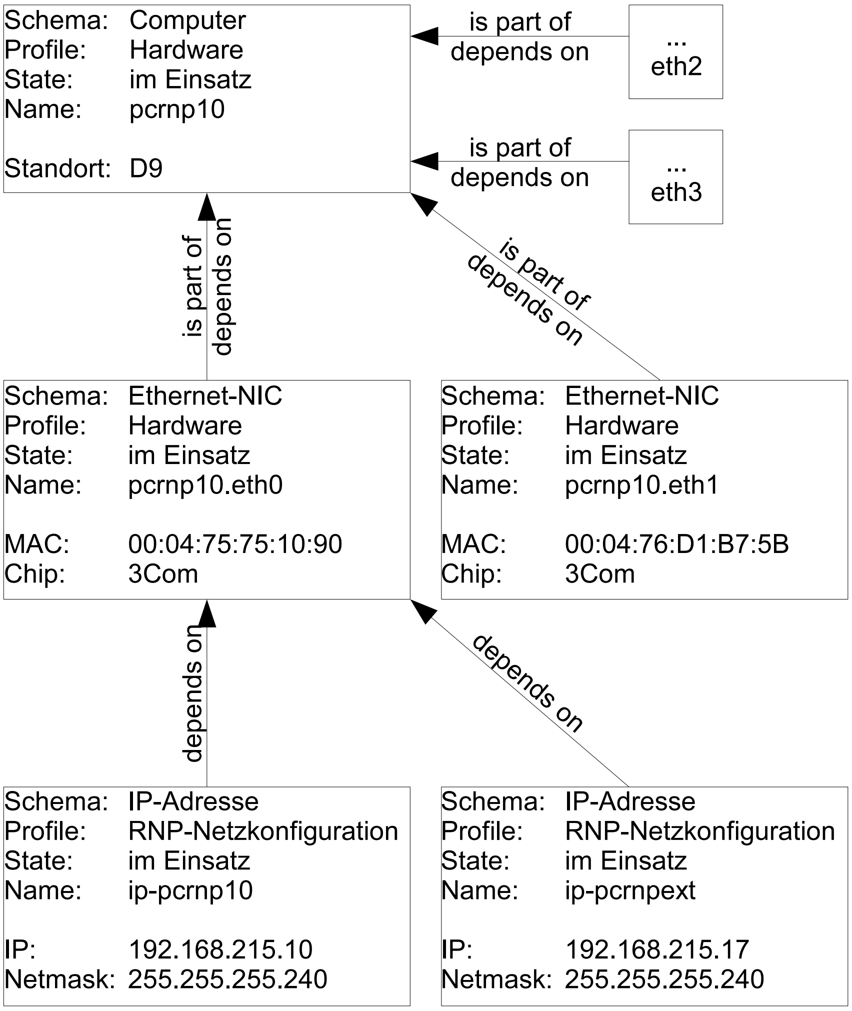

Abbildung 7.1: Beispielkonfiguratioin in ConfDB vom Computer pcrnp10

Die Namen für die Profile und für die Zustände entsprechen den Vorschlägen aus der Tabellen 7.1 und 7.2 auf der vorherigen Seite. Und weil die vorhandene Hardware in beiden Praktika benutzt werden, können alle Configuration Items der gesamten Hardware in einem Profil zusammengefaßt werden. Hingegen unterscheiden sich Netzkonfigurationen der beiden Praktika erheblich, so daß für beide Praktika jeweils ein eigenes Profil für die Netzkonfiguration existiert.

Bedeutsam sind auch die Pfeile in der Abbildung, welche sie die Beziehungen zwischen den Configuration Items darstellen. An jedem Pfeil steht die Bedeutung dieser Beziehung. Beispielsweise soll die Beziehung zwischen den Configuration Items “pcrnp10.eth0” und “ip-pcrnp10” folgendes bedeuten: “ip-pcrnp10 depends on pcrnp10.eth0”

Die Abbildung 7.1 zeigt natürlich nicht die vollständige Konfiguration, weil dann das Diagramm zu viel Platz benötigt. Und die genauen Details liefern auch keine neuen Erkenntnisse für die Benutzung des Programms *ConfDB*. Diese Unvollständigkeit wird in der Abbildung mit der Andeutung der Configuration Items “eth2” und “eth3” offensichtlich.

Toolunterstützung des Incident Managements Man kann die Vorteile des Programms *ConfDB* am Besten im Zusammenhang mit dem Incident Management erkennen. Aus diesem Grund soll die Bedienung des Programmes *ConfDB* in Zusammenarbeit mit dem Trouble Ticket System *OTRS* näher beleuchtet werden.

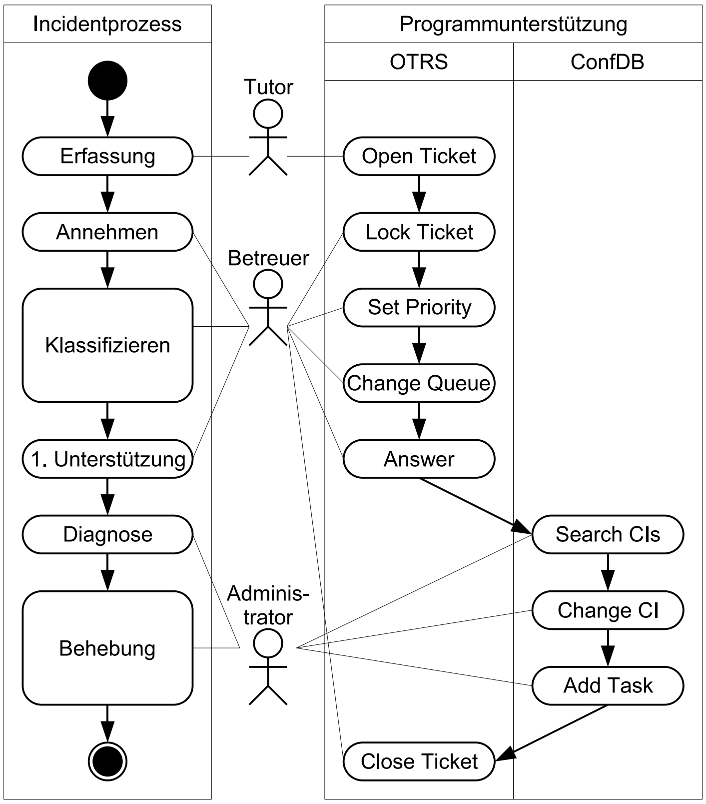

Abbildung 7.2: Toolunterstützung für das Incident Management

Die Abbildung 7.2 zeigt eine vereinfachte Darstellung der möglichen Unterstützung von den beiden Programmen *OTRS* und *ConfDB* für die Tätigkeiten des Incident Managements. In der ersten Spalte stehen die Tätigkeiten für das Incident Management. Die dargestellten Tätigkeiten beschränken sich aber auf die wesentlichen, welche auch sinnvoll mit den beiden Programmen unterstützt werden können. Zum Vergleich zeigt die Abbildung 4.1 auf Seite 35 wesentlich detailierter den Incident Management Prozess.

In der zweiten Spalte stehen die Rollen, welche diese Tätigkeit im Szenario ausführen sollen. Die gewählten Rollen sollen den wahrscheinlichsten Fall dokumentieren, daß ein Tutor während seiner Arbeit eine Störung entdeckt und diese Störung den Praktikumsbetreuer über *OTRS* meldet.

In der dritten und vierten Spalte stehen die Menüpunkte in den Programmen *OTRS* und *ConfDB*. Diese Menüpunkte sollen eine mögliche Unterstützung der Tätigkeit andeuten. Die dazugehörige Tätigkeit steht immer auf der gleichen Höhe in diesem Diagramm. Es ist offensichtlich, daß eine Tätigkeit nicht vollständig mit einem Programm abzudecken ist.

Weil die Abbildung 7.2 aber nur einen groben Überblick über den Ablauf geben kann, soll nun im Detail dieser Ablauf in einem Use Case 7.2.2 auf der nächsten Seite dargestellt werden.

Die Vorteile dieses Vorgehen hängt sehr stark von den auftretenden Störungen ab. Aber folgende Vorteile sollen beim konsquenten Einsatz diese Ablaufs erreichbar sein:

- Keine Störung bleibt unbehandelt, weil alle gemeldeten Störungen in *OTRS* verwaltet werden.

Name	**Zusammenarbeit der Programme OTRS und ConfDB**
Akteure	Tutor, Betreuer, Administrator
Ablauf	1. **Start** Der Tutor stellt eine Störung im Rechnerraum fest. Jedoch ist die Ursache für die Störung nicht offensichtlich und beschließt deshalb, diese Störung zu melden. 2. **Erfassung** Zur Erfassung einer neuen Störung schreibt der Tutor eine eMail an die Mailadresse von *OTRS*. In dieser eMail beschreibt er im Detail die Störung. 3. **Annehmen** Nach dem Eintreffen der eMail meldet *OTRS* das neue Ticket allen Betreuern. Ein Betreuer übernimmt dann dieses Trouble Ticket mit dem Menüpunkt Lock. Ab diesem Zeitpunkt ist dieser Betreuer für die Behebung der Störung verantwortlich. 4. **Klassifizieren** Nach dem der Betreuer sich mit der Störung vertraut gemacht hat, setzt er eine angemessene Priorität für diese Störung fest. Zusätzlich kann er die Queue für diese Störung ändern, falls die Störung in der falschen Queue liegt. 5. **1. Unterstützung** Unter Umständen ist ein Workaround oder die Lösung für die Störung schon bekannt. Dann können diese Informationen gleich dem Tutor oder dem Betroffenen sofort mitgeteilt werden. Für die Übermittlung von Informationen bietet das Programm *OTRS* einen Menüpunkt zum Beantworten der eMail, in der die Störung gemeldet worden ist. 6. **Diagnose** Für die Diagnose und die Behebung der Störung ist ab jetzt der Administrator zuständig. Aus der Störungsbeschreibung können die möglichen Ursachen eingrenzt werden, so daß die Ursachenforschung zielorientiert durchgeführt werden kann. Für den Vergleich des Ist- mit dem Soll-Zustands liefert das Programm *ConfDB* die notwendigen Informationen für den Soll-Zustand. Das Ergebnis einer Diagnose ist die Ursache für eine Störung, welche als eine Menge von Configuration Items in OTRS vermerkt wird. 7. **Behebung** Jetzt werden die notwendigen Änderungen am Rechnerraum vom Administrator durchgeführt. Zusätzlich muß jede Änderung mit dem Menüpunkt Change CI dokumentiert werden. Falls mehrere Configuration Items geändert werden, bietet *ConfDB* die Zusammenfassung der einzelnen Änderungen mit einen Task an. 8. **Ende** Nach dem der Betreuer die Beseitigung der Störung verifiziert hat, kann er in *OTRS* das Trouble Ticket mit dem Befehl Close schliessen.
Vorbedingungen	OTRS und ConfDB sind fertig eingerichtet und bereit
Nachbedingungen	Störung ist behoben und dokumentiert
Qualitätsanforderungen	Die Behebung der Störung wird überprüft und bestätigt

Tabelle 7.3: Use Case: Zusammenarbeit der Programme OTRS und ConfDB

- Alle Änderungen zur Behebung der Störungen werden in *ConfDB* dokumentiert.
- vollständige und aktuelle Dokumentation des Rechnerraums, weil alle durchgeführten Änderungen auch im Programm *ConfDB* aufgezeichnet werden.

Weil alle Informationen über die Störungen und die daraus resultierenden Änderungen aufgezeichnet werden, können diese Informationen bei einem späteren Auftreten der selben Störung wiederverwendet werden und damit kann sehr viel Zeit und Arbeit eingespart werden. Zusätzlich bleibt die Dokumentation der Rechnerkonfiguration immer auf dem aktuellen Stand und erhöht damit den Nutzen für die Administration erheblich.

7.2.3 Zusammenarbeit der Programme

Weil für die Unterstützung der beschriebenen Prozesse nicht mit nur einem Programm möglich ist, müssen die eingesetzten Programme zusammenarbeiten (siehe dazu den Abschnitt 5.3.8 auf Seite 69). Diese Zusammenarbeit kann von einer trivialen Referenz auf die Information bis zur vollständigen Integration von zwei Programmen reichen. Jedoch sind die vorgeschlagenen Programme unabhängig von einander entwickelt worden, so daß die Integration der Programme nur sehr lose möglich ist.

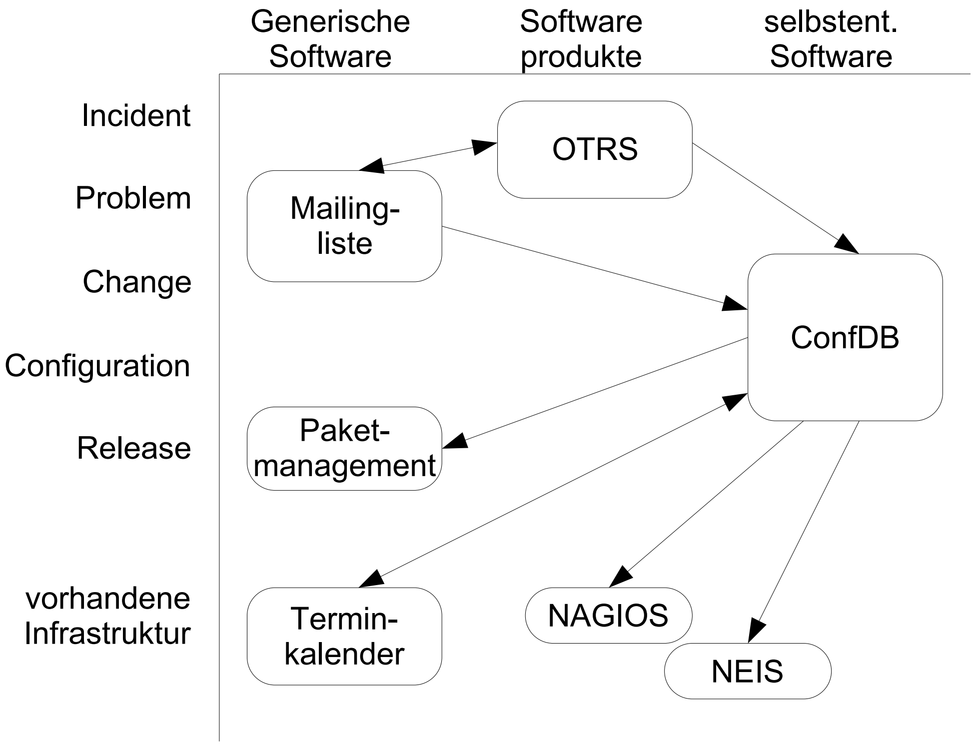

Abbildung 7.3: Zusammenarbeit der Programme

Die Abbildung 7.3 zeigt die unterschiedlichen Programme geordnet nach ihrem Einsatzzweck. Der Einsatzzweck steht an der Ordinate[10] und zeigt den Namen des zu unterstützenden Prozesses an. Auf der Abszisse[11] steht die Kategorisierung der Programme:

- **generischer Software** beschreibt Programmklassen, welche von unterschiedlichen Herstellern mit einem ähnlichen Funktionsumfang angeboten werden.

[10] Die Ordinate ist die vertikale Achse; häufig auch als Y-Achse bezeichnet.
[11] Die Abszisse ist die waagrechte Achse; häufig auch als X-Achse bezeichnet.

- **Softwareprodukt** bennent hingegen ein spezifisches Softwareprodukt, welches für dieses Szenario ausgewählt worden ist.
- **selbstentwickelte Software** sind speziell für den Lehrstuhl entwickelte Programme.

Die Pfeile in der Abbildung 7.3 auf der vorherigen Seite zeigen die möglichen Informationswege. Und über den Informationsaustausch können die Programme zusammenarbeiten (siehe dazu die Anforderung 63 auf Seite 69). Häufig besteht aber die Zusammenarbeit nur über die Aufnahme einer URL eines anderen Programms, so daß der Benutzer mit einem Mausklick von einem Programm auf die Information im anderen zugreifen kann.

Zusammenarbeit von ConfDB mit OTRS Das Trouble Ticket System *OTRS* kann nur sehr schwer die betreffenden Configuration Items identifizieren, weil es weder ein CMDB noch eine Anbindung an eine CMDB besitzt. Aus diesem Grund soll hier die Zusammenarbeit zwischen *OTRS* und *ConfDB* nochmal betrachtet werden. Die mögliche Zusammenarbeit und ihr möglicher Nutzen werden schon in der Evaluierung des Programms erläutert (siehe den Abschnitt 6.3 auf Seite 75).

Dort wird beschrieben, daß in OTRS entweder die URL eines Configuration Items in der CMDB in die Beschreibung aufgenommen werden kann oder es können die Identifikationsnummern der Configuration Items in die Free Fields eingetragen werden. Für die andere Richtung können in *ConfDB* entweder die URL oder die Trouble Ticket Nummer als Attribut eines Configuration Items aufgenommen werden.

Zusammenarbeit von ConfDB oder OTRS mit einer Mailingliste Die Integration einer Mailingliste ist ohne großen Aufwand nur in eine Richtung möglich, in dem die URL, welche auf eine Seite der Programme *ConfDB* oder *OTRS* zeigt, in eine eMail eingegeben wird. Damit kann jeder Leser dieser eMail sofort die entsprechenden Informationen erhalten. Diese Zusammenarbeit ist sehr trivial, jedoch kann sie innerhalb einer Diskussion über die Mailingliste sicher stellen, daß alle einen Zugriff auf die selben Informationen erhalten.

Zusammenarbeit von ConfDB mit einem Paketmanagementprogramm Das Programm *ConfDB* soll auch die installierte Software auf den Rechnern verwalten. Dabei sind die notwendigen Informationen bei vielen Linuxdistribution schon vorhanden, weil die Distributionen ihre Software in Pakete einpacken. Und jedes dieser Pakete enthält ausführliche Informationen über die Software. Und die zwei bekanntesten Paketformate sind dabei:

- **RPM** steht für "RPM Packet Manager" und wird von Red Hat, SuSE und viele andere benutzt.
- **DEB** steht für das Debian-Paketformat und wird von Debian verwendet.

Deshalb wäre es ideal, diese Informationen aus den Softwarepaketen mit den Configuration Items zu synchronisieren. Bis jetzt existiert noch keine Möglichkeit diese Informationen auszutauschen, aber die Entwicklung eines einfachen Programms wäre für die Zukunft möglich. Dieses Programm muß dann Paketinformationen auslesen und die so gewonnen Daten in die Datenbank von *ConfDB* eintragen. Jetzt müssen aber die Informationen manuell in das Programm *ConfDB* eingetragen werden.

Unter *Microsoft Windows* ist die automatische Installation schwieriger, jedoch nicht unmöglich. Eine Möglichkeit ist die Verwendung des Systems Management Server[12], welches ebenfalls von *Microsoft* angeboten wird. Weil aber im Szenario nur sehr wenige Rechner unter *Windows* arbeiten, ist dieses Programm nicht erforderlich. Für das Szenario ist die manuelle Eingabe der eingesetzten Programme unter *Windows* einfacher und erheblich günstiger.

[12] Systems Management Server: `http://www.microsoft.com/smserver/`

Zusammenarbeit von ConfDB mit einer Terminplanungssoftware Weil die Verwaltung von Terminen sehr komplex ist, ermöglicht *ConfDB* den Export der offenen Tasks im iCal-Format (die Details werden in Abschnitt 7.2.1 auf Seite 89 beschrieben). Dies kann dann in einem Terminplanprogramm seiner Wahl importiert werden. Damit behält man im Terminplan immer den Überblick über seine Tasks aus dem *ConfDB*-Programm. Beim Export aus *ConfDB* übergibt es auch die URL für den betreffenden Task, so daß der Zusammenhang aus dem Terminplaner auf den Task in *ConfDB* erhalten bleibt. Diese übergebene URL zeigen viele Programme als Hyperlink an, so daß man mit einem Mausklick wieder auf der Seite von *ConfDB* ist.

Mit dem Export der Tasks im iCal-Format ermöglicht das Programm *ConfDB* eine enge Zusammenarbeit über den Austausch von Informationen mit einem Terminplanungsprogramm.

Zusammenarbeit von ConfDB mit den beiden Programmen Nagios und NEIS Die Zusammenarbeit der drei Programme erfolgt im Wesentlichen mit der Aufnahme einer URL, weil alle drei Programme Webanwendungen sind. Einen direkten Informationsaustausch ist nicht möglich, weil vorallem die Informationen über die Configuration Items in *Nagios* in einem eigenen Format abgelegt werden.

In der Zukunft wäre ein Informationsaustausch möglich, wenn man beispielsweise das Programm *ConfDB* so erweitert, daß es direkt die Informationen aus dem Programm *OTRS* einlesen kann (und damit erfüllt man die Anforderung 63 auf Seite 69).

Fazit Mit dem Einsatz des Programms *ConfDB* kann eine gemeinsame Datenbasis für viele Programme geschaffen werden. Jedoch beschränkt sich die Zusammenarbeit im Wesentlichen auf die Verwendung von URLs, so daß die Benutzer sehr leicht zwischen den Programmen navigieren können. Jedoch fehlt noch der effiziente Informationsaustausch zwischen den Programmen, so daß die selben Daten nicht mehrmals in unterschiedliche Programme eingegeben werden müssen. Eine positive Ausnahme für den Informationsaustausch ist die Zusammenarbeit mit einem Terminkalender. Diese gute Zusammenarbeit ist vorallem durch die gute Unterstützung des iCal-Formats von vielen Terminplanern ermöglicht worden.

7.2.4 Dokumentation für Entwickler

Das Programm verteilt die notwendigen Funktionen und Klassen in verschiedene Verzeichnisse, so daß jeder Entwickler sehr schnell die gesuchte Funktion auffinden kann. In jedem dieser Verzeichnisse liegen die vorgesehenen Dateien:

- actions Jeder Menüpunkt entspricht einer Aktion, welche eine Aufgabe interaktiv mit dem Benutzer durchführt. Dabei ist jede Aktion eigenständig und unabhängig von anderen Aktionen.
- includes Hier liegen zum einen die Datenbankkonfiguration und zum anderen enthalten sie einige Hilfsunktionen, die gemeinsam verwendet werden.
- install Im Installverzeichniss stehen die notwendigen Skripte zum installieren des Programmes.
- classes Im diesem Verzeichnis liegen in jeder Datei eine Klasse. Dabei entspricht der Dateiname auch dem Klassennamen. Diese Klassendateien müssen vor dem Instanziieren der Klasse eingebunden werden.
- help In diesem Verzeichniss stehen die Hilfedateien zu jeder Aktion. Der Dateiname der Hilfedatei ist ähnlich dem Dateinamen der betreffenden Aktion, nur anstelle der Extension .php benutzen die Hilfedateien die Extension .html .

Zentrale Steuerung Die zentrale Steuerung in *ConfDB* ist die Datei index.php , welche auch ein Framework für alle Aktionen bietet. Zuerst stellt es eine Datenbankverbindung her und überprüft dann, ob eine gültige Sitzung vorhanden ist. Falls keine gültige Sitzung aktiv ist, wird eine Authentifizierung des Benutzers verlangt. Bei einer gültigen Sitzung werden die Informationen über den Benutzer in einer globalen Variable abgelegt. Die Informationen in dieser Variable sind überprüft worden und somit ohne weitere Prüfung vertrauenswürdig [13].

Dann wird das Menü für diesen angemeldeten Benutzer ausgegeben. Dabei wird auch immer die Berechtigung überprüft, ob ein Benutzer diese Aktion auch ausführen darf. Wenn der Benutzer eine Aktion ausgewählt hat, wird nochmals die Berechtigung überprüft und gegebenenfalls diese Aktion auch ausgeführt. Den Namen der ausgewählten Funktion wird ebenfalls in einer globalen Variable abgelegt und der Inhalt dieser Variable ist ebenfalls vertrauenswürdig.

Während eine Aktion ausgeführt wird, wird die dazugehörige Hilfedatei dem Benutzer angezeigt. Der Inhalt der Hilfedatei beschreibt den Ablauf und die Bedienung der aktuell ausgeführten Aktion.

Weitere Details Im Anhang B auf Seite 111 sind noch weitere Details zur Implementierung von *ConfDB*, welche nur für Programmierer notwendig sind. Dazu gehört zum einen die Beschreibung des verwendeten Datenbankschemas und eine Übersicht über die verwendeten Klassen in UML.

7.3 Evaluierung

Bei einem genaueren Vergleich zwischen den Anforderungen und die realisierten Ziele stellt man fest, daß nicht alle Anforderungen für das Configuration und insbesondere für das Change Management umgesetzt worden sind. Aus diesem Grund soll jetzt kurz das Programms *ConfDB* evaluiert werden.

7.3.1 Configuration Management

Das Ziel von *ConfDB* ist die Unterstützung der beiden Prozesse Configuration und Change Management. Die höchste Priorität für dieses Programm war eine flexible CMDB für das Szenario. Jedoch darf die Flexibilität nicht zu einem komplexen Programm führen, damit auch eine kleine Personengruppe die CMDB effektiv nutzen kann.

Vereinfachungen Aus diesem Grund sind in der aktuellen Version nicht alle Anforderungen umgesetzt worden, damit das Programm auch langsam in das Szenario eingeführt werden kann. Jedoch sind diese Anforderungen nicht überflüssig, sondern können bei Bedarf später implementiert werden. Und diese Argumentation für eine Vereinfachung soll nun anhand von zwei Anforderungen demonstriert werden. Beide Anforderungen können die Produktivität des Configuration Managements erheblich steigern, aber andere Faktoren können die Realisierung dieser Ideen erschweren. Mit diesem Vorgehen werden deshalb zuerst die grundlegenden Prozessziele realisiert, um dann später weitere Optimierungen und Verbesserungen umzusetzen.

Agenten Eine sehr aufwendige Arbeit beim Configuration Management ist die Eingabe aller Configuration Items in die CMDB. Deshalb wäre eine eine automatisierte Datenerfassung über Agentenprogramme eine große Arbeitserleichterung (siehe dazu die Anforderung 28 auf Seite 61). Jedoch realisiert die aktuelle Version des Programms *ConfDB* diese Anforderung nicht. Ein Grund für diesen Mangel ist die geringe Anzahl von Computern mit sehr unterschiedlichen Konfigurationen. Und diese großen Unterschiede macht die Implementierung eines Agenten sehr schwierig, weil sie für jeden Computer speziell angepasst werden müssen.

[13] Vertrauenswürdig bedeutet, daß der Benutzer den Inhalt dieser Variable nicht unkontrolliert manipulieren kann. Diese Zusicherung ist für ein sicheres Programm notwendig.

Jedoch existieren schon Programme, die vorallem die Netzkonfiguration eines Rechnerraums sehr einfach und schnell erkennen können. Und weil beide Praktika sich mit Computernetzen beschäftigen, ist der Einsatz einer dieser Programme für das Szenario sehr empfehlswert. Eines dieser Programme ist *Nagios*, welche auch schon im Abschnitt 6.6.1 auf Seite 79 evaluiert wurde. Jedoch zeigt die Evaluierung auch, daß die gewonnenen Daten in *Nagios* nicht einfach in *ConfDB* importiert werden können, so daß dieses Programm nur ein Teil der Anforderungen an einen Agenten erfüllen können. Für einen vollständigen Agenten muß das Programm die gewonnen Daten auch korrekt in der CMDB ablegen können (siehe dazu auch die Anforderung 63 auf Seite 69).

Soll- und Ist-Zustand Mit dem Einsatz von Agenten wäre dann auch die Erfüllung der Anforderung mit dem Vergleich des Soll- und Ist-Zustands sehr einfach möglich (ist die Anforderung 29 auf Seite 62). Damit steckt viel Potential in der Realisierung von Agenten für weitere Effizienzsteigerung des Configuration Managements.

Je mehr Configuration Items in einer CMDB gespeichert werden, desto schwieriger sind Abweichungen der Realität mit den eingetragen Werten festzustellen. Damit aber die Informationen in der CMDB auf dem aktuellen Stand bleiben, müssen regelmäßige Überprüfungen des Datenbestandes durchgeführt werden (sieh dazu den Abschnitt 4.4.3 auf Seite 44). Aber der Vergleich des Soll- mit dem Ist-Zustand wäre auch eine gute Hilfe für die Diagnose von Störungen und Problemen, weil die gefundenen Unterschiede schneller auf mögliche Ursachen schliessen lässt.

Falls die Software den aktuellen Zustand der Configuration Items erkennen kann (siehe dazu die Anforderung 28 auf Seite 61), dann könnte sie diese Daten mit den Informationen in der CMDB vergleichen. Und die erkannten Unterschieden kann dann ein Benutzer überprüfen und korrigieren.

Jedoch sprechen zwei Gründe gegen die Implementierung dieser Anforderung: Der erste ist die fehlende Unterstützung von Agenten (siehe dazu den Abschnitt 7.3.1 auf der vorherigen Seite). Aber der wichtigere Grund ist die kleine Anzahl der Configuration Items im Szenario. Weil das Programm im Szenario nur wenige Rechner verwalten muß, übersteigt damit der Aufwand für die Implementierung der Agenten und den automatischen Abgleich des Soll- und Istzustands den möglichen Nutzen für den automatischen Vergleich.

Wenn also später die Überprüfung der CMDB oder die Diagnose wegen der hohen Anzahl von Configuration Items eine erhebliche Belastung darstellen soll, dann wäre die Realisierung dieser Anforderung unter Umständen sinnvoll.

Fazit Die beiden angesprochenen Punkte zeigen deutlich, welche Erweiterungen und Verbesserungen für das Programm *ConfDB* in der Zukunft noch möglich sind. Aber die Beschreibung der beiden Punkte zeigen auch die Herausforderungen für eine Realisierung dieser Ideen. Und weil das Programm *ConfDB* viele andere Anforderungen erfüllen kann, lohnt sich der Einsatz dieses Programmes im Szenario trotz der beiden angesprochenen Punkte.

7.3.2 Change Management

Die Softwareunterstützung des Change Managements muß sehr eng mit der CMDB zusammenarbeiten, damit auch jede Änderung in der CMDB vollzogen wird. Sinnvollerweise realisiert das Programm *ConfDB* nach der Unterstützung des Configuration Management auch das Change Management.

Das Ziel des Change Managements ist die sichere Durchführung von Änderungen an der IT-Infrastruktur (siehe dazu den Abschnitt 4.5.1 auf Seite 45). Um dieses Ziel zu erreichen müssen zuerst alle Änderungen vor ihrer Umsetzung vollständig dokumentiert werden, um danach eine ausführliche Überprüfung durchführen zu können. Nur so können die negative Auswirkungen der Änderungen vermieden werden.

Weil jede Änderung an der IT-Infrastruktur auch eine Änderung an der CMDB erfordert, dokumentiert das Programm *ConfDB* ohne Ausnahme jede Änderung an den Configuration Items. Da häufig viele dieser Änderungen den selben Grund besitzen und auch gemeinsam organisiert werden müssen, kann das

Programm die einzelnen Änderungen an den Configuration Items zu einem Task zusammenfügen. Innerhalb dieses Tasks können dann die notwendigen Änderungen an den Configuration Items koordiniert und kontrolliert werden.

Jetzt stellt sich die Frage, ob die Beschreibung der Tasks auch den Anforderungen an eine RfC genugen (siehe dazu die Anforderungen 31 und 32 auf Seite 62). Man stellt fest, daß ein Task nur ein Teil der Informationen einer RfC enthält. Folgende Informationen fehlen oder sind nur teilweise vorhanden:

- **Abhängigkeiten** Das Programm kennt zwar die Abhängigkeiten zwischen den Configuration Items, aber diese sagen nichts aus, in welcher Reihenfolge die einzelnen Änderungen durchgeführt werden müssen.
- **Bedinungen während der Umsetzung** Die Situation während der Durchführung einer Änderung werden ebenfalls nicht explizit dokumentiert. Jedoch können wieder die Abhängigkeitsbeziehungen zwischen den Configuration Items einen Überblick über die potentielle Auswirkung geben.
- **Herkunft** Auch die Herkunft wird nicht innerhalb des Programms *ConfDB* dokumentiert. Jedoch besitzt jeder Task ein Kommentarfeld, so daß ein informelle Beschreibung der Herkunft im Programm *ConfDB* möglich ist.
- **Resourcen** Weil die notwendigen Resourcen sehr vielfältig sind und auch zum größten Teil auch nicht vom Programm *ConfDB* erfaßt sind, kann es auch nicht die Resourcen effektiv verwalten. Einer der wichtigsten Resourcen, welche im Programm nicht erfaßt ist, ist die verfügbare Arbeitszeit der Mitarbeiter.

Diese Informationen können häufig nicht automatisch berechnet werden, so daß die Personen selber die oben beschriebenen Informatione erarbeiten und eingeben muß. Damit die Arbeit der Administratoren nicht durch die strengen Regeln eingeschränkt werden, überlässt das Programm *ConfDB* die Entscheidung, in welchen Detailierungsgrad die Änderungen dokumentiert werden müssen. Vorallem hängt der notwendige Detailierungsgrad von der Komplexität der Änderung ab, so daß eine allgemeine und statische Vorgabe von einem Programm eher behindert als nützt. Aus diesen Gründen implementiert das Programm *ConfDB* nur einen kleinen Teil der Anforderungen einer RfC (siehe die Anforderung 32 auf Seite 62).

Für die Klassifizierung (siehe die Anforderung 34 auf Seite 63) der unterschiedlichen Änderungen bietet das Programm vorallem das Setzen einer Priorität für jede Änderung und für jeden Task. Und somit bleibt auch bei einer großen Zahl von Änderungen der Überblick vorhanden. Eine weitere Klassifizierung überläßt die Software dem Änderungskoordinator, in dem er seine Tasks in seinen Terminkalender importiert und dann dort verwalten kann.

Änderungen planen und koordinieren Die nächsten Schritte im Change Management Prozess ist die Planung, Genehmigung und Koordination der Änderung (siehe dazu die Anforderungen 35 und 37 auf Seite 63). Weil jeder dieser Aufgaben effektiver in einem Terminplanungsprogramm durchgeführt werden kann, bietet das Programm *ConfDB* den komfortablen Export der Tasks im iCal-Format an. Damit übernimmt das Programm *ConfDB* nicht die direkte Unterstützung dieser Tätigkeit, sondern überträgt die Verantwortung für die Unterstützung auf ein Terminplaner.

Ein besonderer Fall sind die eventuell notwendigen Genehmigungen (siehe die Anforderung 37 auf Seite 63), weil der Umfang und der Prozess für das Einholen einer Genehmigung sehr spezifisch für eine Organisation ist. Und im Szenario müssen nur in Ausnahmefällen die Änderungen genehmigt werden, und deshalb ist eine Softwareunterstützung nicht notwendig.

Änderungen testen Jedoch reicht eine ausführliche Planung und Koordination nicht aus, damit eine Änderung auch ein fehlerfreies Ergebniss liefert. Dazu bedarf es vorallem ausführliche Tests, so daß alle Nebenwirkungen erkannt werden (siehe die Anforderung 38 auf Seite 63). Jedoch bietet das Szenario nur geringe Kapazitäten für ausführliche Tests an, weil der Platz und die Resourcen dazu fehlen. Aus diesem Grund bietet die Software auch keine explizite Unterstützung von Tests an. Es können die notwendigen

Änderungen schon im voraus in der CMDB eingegeben werden. Und nach dem erfolgreichen Bestehen der Tests können dann diese neue Versionen aktiviert werden.

Jedoch bietet das Programm *ConfDB* keine Möglichkeit die Ergebnisse von Tests aufzunehmen. Und deshalb sollen die Testergebnisse in eigene Dokumente beschrieben werden, falls eine Änderung ausführlich getestet wurde.

Fazit Abschliessend stellt man fest, daß das Programm *ConfDB* selbst nur die wesentlichen Informationen einer Änderung verwaltet. Viele Tätigkeiten können besser durch ein Terminplanungsprogramm unterstützt werden und deshalb ist die Anbindung eines Terminplans so wichtig. Insgesamt kann also das Programm *ConfDB* die Grundlage für ein Change Management Prozess legen.

7.3.3 Release Management

Das Programm *ConfDB* ist nicht primär für das Release Management entwickelt worden. Aber es kann das Release Managements ein wenig bei ihrer Arbeit unterstützen.

Mit den Profilen können die beiden Policen DSL und DHS sehr einfach verwaltet werden (siehe dazu die beiden Anforderungen 27 und 45). Zusätzlich können die Releases ähnlich wie im Change Management kontrolliert werden, in dem die notwendigen Änderungen in der CMDB zu einem Task zusammengeführt werden. Damit kann das Programm *ConfDB* bei einem einfachen Release Management helfen.

Die besondere Anforderung für die DSL mit der automatischen Softwareinstallation können teilweise mit den Paketmanagementprogrammen der eingesetzten Linuxdistributionen und mit Hilfe von Shellskripten erfüllt werden (siehe die Anforderung 44 auf Seite 64). Der Einsatz von Skripten für die automatische Installation ist häufig auch die einfachste und schnellste Lösung des Problems.

Am Ende ist die Unterstützung des Release Managements nur minimal, weil im Szenario nur selten neue Releases eingeführt werden. Denn häufig erfordert die neue Releases auch eine Überarbeitung der Praktika, welches sehr viel Arbeit bedeutet. Und falls doch ein neues Release zusammengestellt wird, werden die nur in den vorlesungsfreien Zeiten umgesetzt, damit auch alle Probleme rechtzeitig vor dem Praktikumsbeginn erkannt und behoben sind.

7.3.4 Weitere Anforderungen

Nachdem das Programm *ConfDB* anhand den Anforderungen der einzelnen Prozesse untersucht worden ist, soll nun die prozessunabhängigen Anforderungen genauer betrachtet werden (siehe dazu den Abschnitt 5.3 auf Seite 65). Weil schon einige Anforderungen in der Besprechung des Programms erwähnt worden sind, werden hier nur die weniger beachteten Aspekte untersucht.

Berichte erstellen Die Evaluierung der Prozesse berücksichtigt bis jetzt nicht die Unterstützung zur Erstellung von Berichten für den Prozess-Manager. Zur Zeit kann das Programm *ConfDB* auch keine Berichte erstellen und somit können die dazugehörigen Anforderungen auch nicht erfüllt werden.

Die größte Herausforderung stellt bestimmt die Anforderung 52 auf Seite 66 für benutzerdefinierte Berichte, weil dies nur über die Implementierung einer eigenen Abfragesprache möglich wäre. Diese Sprache benötigt dann einen einheitlichen Zugriff auf alle verfügbaren Informationen und sinnvolle Operationen auf diese Daten. Somit ist die Hürde für eine Implementierung sehr hoch und ob der Nutzen für das Szenario diesen Aufwand gerechtfertigt, ist nicht bekannt. Die umfangreichste Möglichkeit einen eigen Bericht zu erstellen ist der Einsatz einer Programmiersprache.

Eine vereinfachte Anforderung ist die Bereitstellung von fertigen Berichten. Dann kann auf den vollen Umfang einer Programmiersprache zurückgegriffen werden und so können auch sehr komplexe Berichte erstellt werden. Ob aber diese Berichte dann auch sinnvolle Informationen liefern können, bleibt ebenfalls fraglich. Beispielsweise können die Anzahl der Configuration Items oder die Anzahl der Änderungen keine Auskunft über die Effizienz der Prozesse geben.

Aus diesem Grund ist die Berichterstellung noch nicht implementiert. Aber mit der vollständigen Dokumentation des Datenbankschemas in der Abbildung B.1 auf Seite 112 können sehr einfach individuelle Programme zum Erstellen von Berichtes geschrieben werden. Der Aufwand für einen individuellen Bericht ist dabei relativ gering, so daß die Erstellung auch erst bei Bedarf schnell realisiert werden kann.

Kommunikation Ein wichtiger Punkt bei der Analyse des Szenarios war das Kommunikationsproblem (siehe dazu den Abschnitt 3.5.2 auf Seite 30). Deshalb ist die Realisierung der beiden Anforderungen 54 und 55 auf Seite 68 für das Szenario sehr wichtig.

Zuerst bietet das Programm die Auskunft über die Telefonnummer und der eMailadresse jedes Benutzers, so daß auch die Kommunikation zwischen den Benutzern gefördert wird. Desweiteren verdeutlicht die uneingeschränkte Veröffentlichung der Tasks die Wichtigkeit der Zusammenarbeit. Diese Veröffentlichung ist aber nur deshalb so einfach möglich, weil die Anzahl der Mitarbeiter und damit die Anzahl der Benutzer im Szenario sehr gering ist.

Und der Einsatz eines Terminkalenders ermöglicht auch die Erinnerung an die Umsetzung der eingegebenen Tasks (siehe die Anforderung 59 auf Seite 69). Der Zeitpunkt für den Alarm wird dabei automatische beim Export im iCal-Format mitgeliefert, damit auch eine pünktliche Erinnerung gewährleistet wird. Auch deshalb ist die enge Zusammenarbeit mit einem Terminkalender notwendig, weil eine Webanwendung nur sehr schwer den Benutzer pünktlich alarmieren kann.

Damit ermöglicht das Programm *ConfDB* die Zusammenarbeit der Benutzer, in dem es die notwendigen Informationen bereit stellt.

7.4 Fazit

Das Programm *ConfDB* ermöglicht eine effektive Verwaltung aller Configuration Items und erfüllt damit das gesteckte Ziel. Manche Anforderungen sind teilweise wegen den Besonderheiten des Szenarios nicht realisiert worden. Wenn später eine Implementierung sinnvoll erscheint, kann das Programm wie vorgeschlagen erweitert werden.

Der entwickelte Prototyp ist stabil genug, um die vorhandenen Funktionen sinnvoll zu nutzen. Damit soll der Einsatz innerhalb des Szenarios möglich sein, weil es auch die spezifischen Anforderungen des Szenarios erfüllen kann. Und auserhalb des Szenarios kann das Programm auch eingesetzt werden, wenn die angebotenen Fähigkeiten dort sinnvoll genutzt werden können.

8 Zusammenfassung und Ausblick

Die Optimierung von Arbeitsabläufen besitzt kein Ende, weil durch die schnellen Veränderungen in der Umwelt sich auch die Ziele entsprechend anpassen. Und somit ergeben sich immer wieder neue Möglichkeiten für weitere Optimierungen. Aus diesem Grund bietet diese Arbeit keinen optimalen Endzustand an, sondern erarbeitet Vorschläge für die Einführung von ITIL-konformen Prozessen. Jedoch erfordern auch diese Prozesse eine ständige Weiterentwicklung, weil nur so kann man sich an eine veränderte Situation anpassen.

8.1 Ergebnisse dieser Arbeit

Diese Arbeit zeigt eine Umsetzung der ITIL Empfehlungen am Beispiel eines Szenarios. Die Realisierung dieser Empfehlungen erfolgte über die Beschreibung der Prozesse und zeigte den möglichen Einsatz von Softwareprodukten für die Prozessunterstützung.

Aus den ITIL Empfehlungen sind dann Prozesse entwickelt worden, welche die Administration eines Rechnerraums optimieren können. Und damit sind die beschriebenen Prozesse auch eine ITIL-konforme Umsetzung der Empfehlungen. Obwohl die Prozesse für das Szenario entwickelt worden sind, bleiben sie so allgemein wie möglich. Dadurch können sie auch in andere Situationen sinnvoll eingeführt werden.

In den Prozessen werden dabei viele Informationen zusammengetragen, bewertet und verarbeitet. Und diese Informationsflut kann man am Besten mit den Einsatz von EDV bewältigen. Deshalb folgen aus den Prozessen auch die Anforderungen für eine Softwareunterstützung. Die Anforderungen bilden damit die Grundlage für Erarbeitung eines schlüssigen Softwarekonzepts für die Service Support Prozesse. Jedoch können die Prozesse nicht vollständig mit einer Software automatisiert werden, weil jeder Prozess eine Bewertung der vorliegenden Daten erfordert. Aus diesem Grund bleiben die Prozesse eine Empfehlung für die Mitarbeiter, um ihre Arbeit effizienter zu gestalten.

Das entwickelte Softwarekonzept besteht dann aus zwei Teilen: Zum ersten sollen vorhande Programme eingesetzt werden, wenn sie eine sinnvolle Unterstützung der Prozesse anbieten können. Dazu sind in der Evaluierung eine kleine Auswahl an Programmen untersucht worden, welche innerhalb des Szenarios eingesetzt werden können. Jedoch können diese Produkte nicht alle gestellten Anforderungen befriedigend erfüllen, so daß im zweiten Teil ein neues Programm mit dem Namen *ConfDB* für das Szenario entwickelt worden ist. Eine wichtige Aufgabe für dieses Programm war auch die Integration mit den anderen Softwareprodukten, damit die Bedienung der Programme so einfach wie möglich gestaltet werden kann.

Damit bietet das angebotene Softwarekonzept eine umfassende Unterstützung der vorgestellten Prozesse, welche auch die zentrale Aufgabe dieser Arbeit war. Und mit dieser Arbeit können dadurch konkrete Probleme innerhalb des Szenarios behoben werden. Zusätzlich zeigt sie auch an einem Bespiel das Vorgehen für die Einführung von ITIL Prozessen.

8.2 Ausblick

Auch diese Arbeit kann keine ideale Lösung aller Probleme anbieten, weil sich auch die Anforderungen weiter entwickeln. Somit bleibt die Herausforderung für eine weitere Optimierung der Arbeitsabläufe erhalten. Mit der schnellen Entwicklung der IT-Industrie bieten sich auch ständig neue Ideen, um diese Verbesserungen auch zu realisieren.

Ein neuer Trend in der Computerindustrie ermöglicht eine weitere Verbesserung des Informationsaustausches auch zwischen sehr unterschiedlichen Programmen. Die Grundlage für diesen Austausch ist das universelle Datenformat XML, welches ständig neue Bereiche in der EDV erobert. Ein verbesserter Informationsaustausch ermöglicht dann weitere Optimierungen der Arbeitsabläufe und damit auch des Managements.

Jedoch können heute die Informationsmenge von einem Menschen gar nicht mehr verarbeitet werden. Dadurch erfordert die steigende Informationsmenge auch eine intelligentere Verarbeitung. Die möglichen Ansätze bietet die Forschung im Bereich der Künstlichen Intelligenz an. Ein Beispiel für neue Möglichkeiten bei den Service Support Prozessen wäre die automatische Diagnose von Störungen und Problemen. Dabei berechnet die Software bei der Eingabe der Symptome aus der Beschreibung der IT-Infrastruktur die möglichen Ursachen[1].

Mit den beiden Beispielen soll gezeigt werden, daß mit neuen Ideen und Werkzeugen jederzeit das IT-Management weiter verbessert werden kann. Deshalb kann diese Arbeit nur ein weiterer Schritt in der Entwicklung der Supportprozesse sein.

[1] siehe zum Beispiel die Arbeit [Hell 01]

A Details zum Szenario

In diesem Anhang sind die ausführlichen Tabellen mit den technischen Details für das Szenario enthalten. Diese detailierten Informationen können einerseits für die Ausgangskonfiguration der CMDB benutzt werden oder als Nachschlagewerk für die Admnistration des Rechnerraums eingesetzt werden.

A.1 Adressen und Namen

Eine wichtige Voraussetzung für die beiden Praktika ist die Vernetzung aller Computer im Rechnerraum. Und damit alle Computer auch im Netz eindeutig identifiziert werden können, wird jedem Rechner eine Adresse und einen Namen zugewiesen.

Weil die Praktika das Thema Netzwerkkonfiguration intensiv bearbeiten, ist die Netzkonfiguration im Rechnerraum komplexer als üblich. Ein schematischer Überblick über die Netzkonfiguration für den IP-Abschnitt im Rechnernetzpraktikum bietet die Abbildung A.1 auf der nächsten Seite an. Dieses Schema ist einem Diagramm[1] in der Anleitung[2] für die Teilnehmer des Rechnernetzpraktikums nachempfunden. In diesem Schema stellen die große Rechtecke die Computer mit ihren Namen dar. Und jeder Computer besitzt mindestens eine Netzwerkkarte, welche in den kleinen Quadraten an den Computer symbolisiert werden. In den Quadraten steht in der ersten Zeile der Gerätename und in der zweiten Zeile der DNS-Name für die zugewiesene IP-Adresse. Ein “ ˜ ”-Symbol wird dabei als Abkürzung für den Rechnernamen benutzt.

Desweiteren werden die verwendeten Adressen und Namen in den folgenden Tabellen dargestellt. Zuerst zeigt die Tabelle A.1 auf Seite 107 die verwendeten IP-Adressen und ihre Namen im IT-Sicherheitspraktikum. Jedoch verwendet das IT-Sicherheitspraktikum zwei verschiedene Netzkonfigurationen und beide Konfigurationen mit den Namen “Hub” und “Switch” werden in dieser Tabelle zusammen dargestellt. Falls in einer Zeile kein Name in einer Konfigurationsspalte steht, bedeutet dies, daß in dieser Konfiguration die betreffende IP-Adresse nicht verwendet wird.

Und im Rechnernetzpraktikum ist der Zusammenhang zwischen IP-Adresse und Rechnername besonders informativ, weil es auch einen Einblick in die Netztopologie gibt. Die Tabelle A.2 auf Seite 108 zeigt die DNS-Namen im Rechnernetzpraktikum. Zusätzlich zu den Namen der Computer besitzen auch die Subnetze jeweils einen Namen. Und zur besseren Unterscheidung werden diese in der Tabelle in Kapitälchen dargestellt.

Normalerweise reicht die Information über die IP-Adresse eines Rechners aus, um den gewünschten Rechner im Netzwerk anzusprechen. Jedoch sollen die Praktikumsteilnehmer die Konfiguration eines Netzwerks üben, so daß die eingestellten IP-Adressen der Rechner auch fehlerhaft sein kann. Aus diesem Grund ist es für die Administration enorm wichtig, auch die MAC-Adressen aller Netzwerkkarten zu kennen, so daß auch bei einer Fehlkonfiguration der IP-Adressen immer noch die beteiligten Rechner einer Netzwerkkommunikation identifiziert werden können. Deshalb stehen in der Tabelle A.3 auf Seite 109 die MAC-Adressen aller Netzwerkkarten.

[1]”Abbildung 2.4: Aufbau des Versuchsnetzes” auf Seite 37 in der Anleitung
[2]Anleitung: `http://wwwhegering.informatik.tu-muenchen.de/Praktika/ss04/rnp/tutorials-ip-student.pdf`

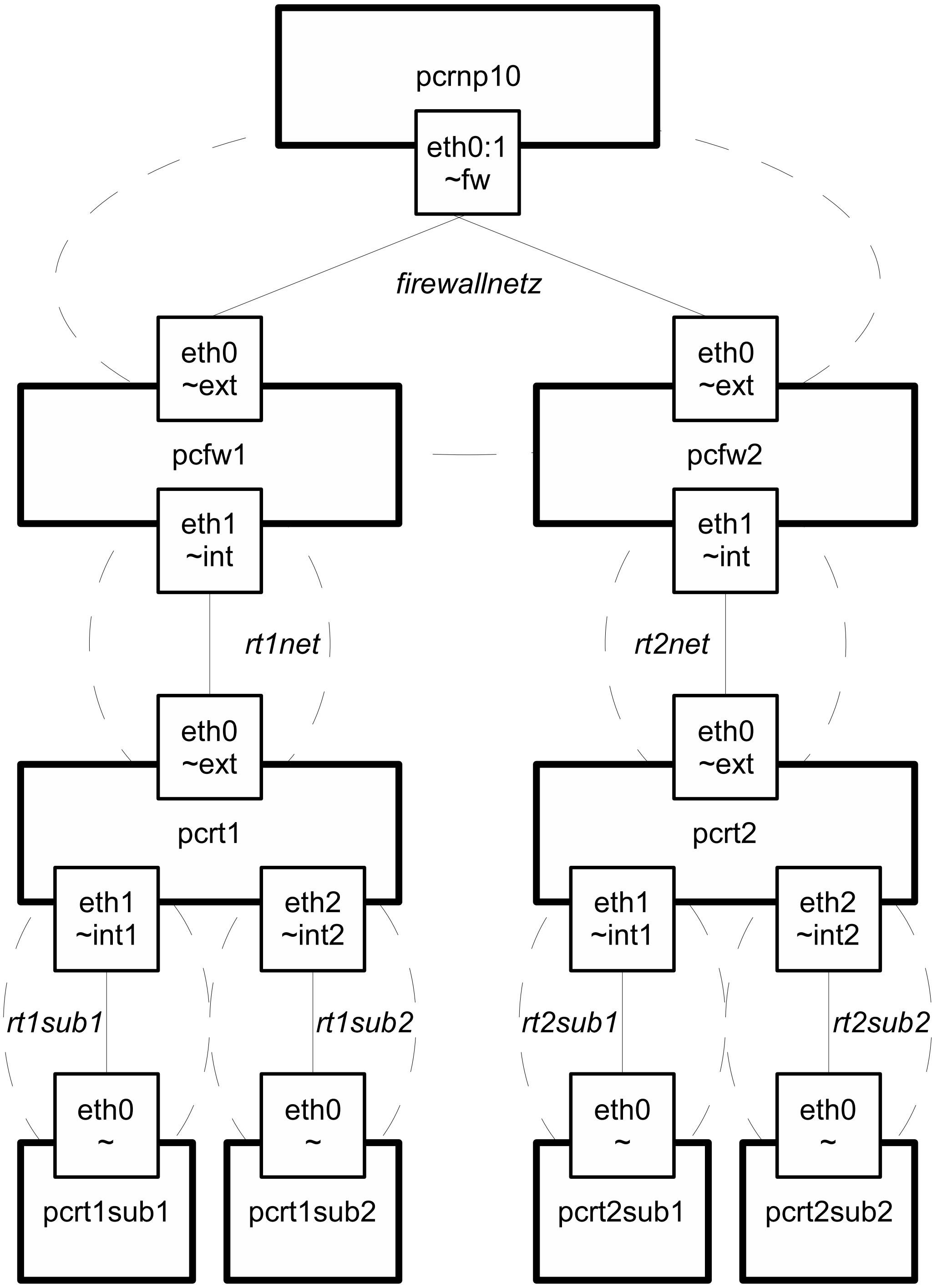

Abbildung A.1: Schema der Netzkonfiguration für den IP-Abschnitt im RNP

A.2 DHCP Konfigurationen

Ein Teil der Praktika fordert von den Teilnehmern die Durchführung der Netzkonfiguration. Deshalb werden auch viele Einstellungen nicht fest eingestellt, sondern müssen von den Teilnehmern zur Laufzeit

IP-Adresse	Hub	Switch
192.168.216.1	pcsec1	pcsec1
192.168.216.2	pcsec2	pcsec2
192.168.216.3	pcsec3	
192.168.216.4	pcsec4	
192.168.216.5	pcsec5	
192.168.216.23		pcsec3
192.168.216.24		pcsec4
192.168.216.45		pcsec5
192.168.216.46		pcsec6
192.168.216.67		pcsec7
192.168.216.68		pcsec8
192.168.216.109		pcsec9
192.168.216.110		pcsec10
192.168.216.156	pcsec6	
192.168.216.157	pcsec7	
192.168.216.158	pcsec8	
192.168.216.159	pcsec9	
192.168.216.160	pcsec10	
192.168.216.201		pcsec1-switch
192.168.216.203		pcsec3-switch
192.168.216.205	pcsec5-switch	pcsec5-switch
192.168.216.207		pcsec7-switch
192.168.216.209	pcsec9-switch	pcsec9-switch
192.168.216.252	hacktest	hacktest
192.168.216.253	test4all	test4all
192.168.216.254	secserver	secserver

Tabelle A.1: DNS Namen und Adressen im Sicherheitspraktikum

konfiguriert werden.

Jedoch werden für eine einfachere Softwarewartung bei jedem Einschalten des Computers das ausgewählte Betriebssystem über das Netzwerk geholt und gestartet. Um dies zu realisieren exisitiert eine minimale DHCP-Konfiguration, damit die Rechner beim Hochfahren temporär eine IP-Adresse und damit auch Zugriff auf ihr Betriebssystem erhalten. Diese Konfiguration ist im Rahmen eines Systementwicklungsprojekt erarbeitet worden (für die Details siehe [AlCh 03]).

Für jedes Praktikum existiert dabei eine eigen DHCP-Konfiguration, welche in den beiden Tabellen A.4 auf Seite 110 und A.5 auf Seite 110 vorgestellt werden.

IP-Adresse	Name	IP-Adresse	Name
192.168.215.0	RNPNETZ	192.168.215.80	RT1NET
192.168.215.1	vlanswitch1	192.168.215.81	pcrt1ext
192.168.215.2	vlanswitch2	192.168.215.94	pcfw1int
192.168.215.3	vlanswitch3		
192.168.215.4	hprnp4	192.168.215.96	RT1SUB1
192.168.215.5	secserv	192.168.215.97	pcrt1sub1
192.168.215.7	winrnp7	192.168.215.110	pcrt1int1
192.168.215.10	pcrnp10		
		192.168.215.112	RT1SUB2
192.168.215.16	EXTERNKOPPEL	192.168.215.113	pcrt1sub2
192.168.215.17	pcrnpext	192.168.215.126	pcrt1int2
192.168.215.30	ognog		
		192.168.215.128	FIREWALLNETZ
192.168.215.32	ATMETHER	192.168.215.129	pcrnp10fw
192.168.215.33	pcrnp10atm	192.168.215.131	pcfw1ext
192.168.215.34	swatm4	192.168.215.132	pcfw2ext
192.168.215.35	swatm5		
192.168.215.40	swatmether0	192.168.215.144	RT2NET
192.168.215.41	hprnp1	192.168.215.145	pcrt2ext
192.168.215.42	hprnp2	192.168.215.158	pcfw2int
192.168.215.43	pcatm3		
192.168.215.44	pcatm4	192.168.215.160	RT2SUB1
192.168.215.45	hprnp5	192.168.215.161	pcrt2sub1
192.168.215.46	hprnp6	192.168.215.174	pcrt2int1
192.168.215.48	ATMLANE	192.168.215.176	RT2SUB2
192.168.215.49	swlane4	192.168.215.177	pcrt2sub2
192.168.215.50	swlane5	192.168.215.190	pcrt2int2
192.168.215.51	hplane1		
192.168.215.52	hplane2	192.168.215.192	NM1
192.168.215.53	pclane3	192.168.215.193	pcrnp10nm1
192.168.215.54	pclane4	192.168.215.194	pcnm1prot
192.168.215.55	hplane5	192.168.215.195	pcnm1ov
192.168.215.56	hplane6	192.168.215.201	swnm1
192.168.215.64	ATMCLIP	192.168.215.208	NM2
192.168.215.65	swclip4	192.168.215.209	pcrnp10nm2
192.168.215.66	swclip5	192.168.215.210	pcnm2prot
192.168.215.71	hpclip1	192.168.215.211	pcnm2ov
192.168.215.72	hpclip2	192.168.215.212	swnm2
192.168.215.73	pcclip3		
192.168.215.74	pcclip4	192.168.215.224	BOOT
192.168.215.75	hpclip5	192.168.215.225	pcrnp10boot
192.168.215.76	hpclip6	192.168.215.233	bootsec3
		192.168.215.234	bootsec4
		192.168.215.235	bootsec5
		192.168.215.236	bootsec6

Tabelle A.2: DNS Namen und Adressen im Rechnernetzpraktikum

Rechner	MAC-Adresse	Hersteller
hprnp1	08:00:09:5f:c2:94	
hprnp2	08:00:09:c2:79:ac	
hprnp4	08:00:09:c2:d6:c8	
hprnp5	08:00:09:7a:ac:ec	
hprnp6	00:60:b0:a7:04:9d	
pcfw1	00:04:76:9b:ab:0f	3Com
	00:60:08:62:ed:16	3Com
pcfw2	00:04:76:a1:cf:ad	3Com
	00:a0:24:cb:85:c9	3Com
pcfwsub1	02:60:8c:7e:c1:36	
pcrnp10	00:04:75:75:10:90	3Com
	00:04:76:D1:B7:5B	3Com
	00:10:22:FD:4A:F0	3Com
	00:10:5A:31:39:14	3Com
pcrt1	00:04:76:9c:0f:86	3Com
	00:50:ba:1d:9d:fb	Via
	00:c1:26:0e:e7:e8	Realtek
pcrt1sub1	00:04:76:db:fa:28	3Com
	00:05:5d:dd:e0:35	Via
	00:c1:26:0e:78:ee	Realtek
pcrt1sub2	00:04:76:da:c1:11	3Com
	00:05:5d:dd:e6:5a	Via
pcrt2	00:01:02:f4:a3:11	3Com
	00:05:5d:dd:cf:c7	Via
	00:08:54:09:64:04	Realtek
pcsec10	00:02:b3:d7:63:30	Intel
	00:04:76:db:fa:32	3Com
	00:05:5d:a5:6c:1d	Via
pcsec1	00:04:76:db:fa:40	3Com
	00:50:ba:1d:8e:e7	Via
	00:c1:26:0f:0c:c6	Realtek
pcsec5	00:04:75:bf:f7:e5	3Com
	00:50:ba:1d:9d:f3	Via
pcsec6	00:04:76:db:fa:6a	3Com
	00:05:5d:dc:e1:02	Via
pcsec7	00:02:b3:d7:5d:0d	Intel
	00:04:76:da:c1:03	3Com
	00:50:ba:1d:9d:f7	Via
	00:c1:26:0e:79:22	Realtek
pcsec8	00:02:b3:d7:5b:9b	Intel
	00:04:76:da:c2:3f	3Com
	00:50:ba:1d:9e:00	Via
pcsub2	00:00:c0:30:5a:a4	

Tabelle A.3: MAC Adressen

Rechner	MAC-Adresse	IP-Adresse
swrnp1	00:20:48:1f:13:f1	
swrnp2	00:20:48:1f:14:fb	
pcatm4	00:04:76:da:c1:fd	192.168.215.44
pcatm3	00:04:76:db:fa:40	192.168.215.43
bootsec3	00:04:76:db:fa:28	192.168.215.233
bootsec4	00:04:76:da:c1:11	192.168.215.234
bootsec5	00:04:75:bf:f7:e5	192.168.215.235
bootsec6	00:04:76:db:fa:6a	192.168.215.236
swrnpatm	00:60:b0:70:ea:d0	
pcnm1ov	00:04:76:da:c2:3f	192.168.215.195
pcnm1prot	00:04:76:da:c1:03	192.168.215.194
pcnm2ov	00:04:76:db:fa:32	192.168.215.211
pcnm2prot	00:04:76:db:fa:37	192.168.215.210

Tabelle A.4: DHCP Konfiguration für das Rechnernetzpraktikum

Rechner	MAC-Adresse	IP-Adresse
bootsec1	00:04:76:db:fa:40	192.168.100.31
bootsec3	00:04:76:db:fa:28	192.168.100.33
bootsec5	00:04:75:bf:f7:e5	192.168.100.35
bootsec7	00:04:76:da:c1:03	192.168.100.37
bootsec9	00:04:76:db:fa:37	192.168.100.39

Tabelle A.5: DHCP Konfiguration für das IT-Sicherheitspraktikum

B Details zum Programm ConfDB

Hier werden die internen Details zum Programm ConfDB beschrieben. Diese Informationen können sehr nützlich für eine Weiterentwicklung oder für das Auffinden von Fehlern sein.

B.1 Datenbankschema

Eine ersten Übersicht über den Programmablauf erhält man am Besten durch eine Beschreibung des verwendeten Datenmodell. Dazu gehört bei dem Programm *ConfDB* vorallem die Beschreibung des Datenbankschemas.

Besonderheiten bei der Tabelle civalue Eine Besonderheit im Datenbankschema ist die Tabelle civalue, welche die Attributwerte aller Configuration Items enthält. Weil die Attribute eines Configuration Items unterschiedliche Datentypen besitzen können und dieser Datentyp erst bei der Laufzeit vom Anwender definiert wird, muß die Tabelle civalue alle möglichen Datentypen aufnehmen können. Diese Fähigkeit kann über zwei Wege realisiert werden:

- Es können alle mögliche Datentypen auf ein Datentyp bijektiv abgebildet werden. Als Zieldatentyp bietet sich dabei der Typ “String” an, weil die Daten in anderen Datentypen am einfachsten von und nach String konvertiert werden können.
- Für die wichtigsten Datentypen jeweils eine eigenes Datenbankattribut hinzufügen, so daß die Konvertierung zu den verfügbaren Datentypen einfacher wird. Diese Methode erlaubt es auch komplexere Abfragen an die Daten zu stellen, weil nicht jeder Datentyp alle Abfrageoperationen unterstützt.

Die zweite Methode ist für das Programm gewählt worden, weil das Suchen nach bestimmten Attributen einen sehr hohen Stellenwert besitzt. Mit der gewählten Lösung können beispielsweise Integerdaten auch mit Ungleichungen abgefragt werden, welches mit der ersten Lösung nur erschwert möglich ist.

Hier werden die einzelnen Attribute aus der Abbildung B.1 auf der nächsten Seite erläutert:

Tabelle	**ci**
Beschreibung	Hier werden die einzelnen Attribute der Tabellen beschrieben.
id	CI ID
name	Name des CIs
vendor	Name des Herstellers, welche diese CI produziert.
profile	Referenz auf die Tabelle profile, in der die CI zugeordnet ist.
schema	Dieses CI folgt diesem Schema. Referenz auf die Tabelle cischema.
activeversion	Zeigt auf die aktuelle und aktive Version dieses CIs hin.

Tabelle	**cischema**
Beschreibung	Ein Schema für ein CI ist ein Tupel von typisierten Attributen.
id	Schema ID
name	Name des Schemas.
comment	Beschreibung des Schemas.

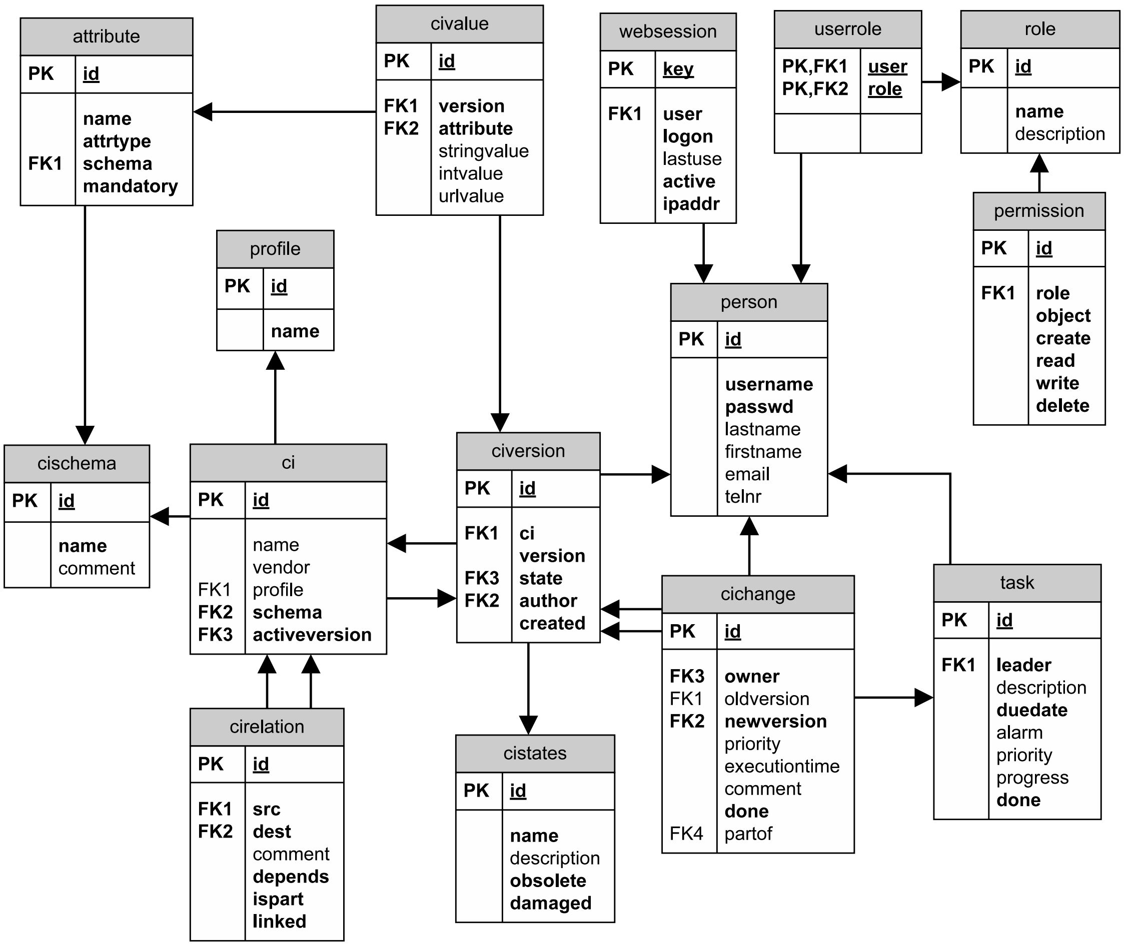

Abbildung B.1: Entity-Relationship-Modell für ConfDB

Tabelle	**attribute**
Beschreibung	Jedes Attribut ist Teil eines Schemas.
id	Attribute ID
name	Name des Attributes
attrtype	Typ dieses Attributes (Auswahl aus String, Integer, Boolean oder eine URL).
schema	Dieses Attribut ist Teil dieses Schemas. Referenz auf die Tabelle cischema.
mandatory	Ist dieses Attribute für die Beschreibung eines CIs erforderlich?

Tabelle	civalue
Beschreibung	In der Tabelle civalue sind die Attributswerte der CI-Instanzen enthalten.
id	Werte ID
version	Dieser Attributwert ist Teil folgender Version. Referenz auf die Tabelle civersion.
attribute	Der hier abgelegte Wert gehört zu folgendem Attribut. Referenz auf die Tabelle attribute.
stringvalue	Enthält den Attributwert, falls der Attributstyp entweder String oder URL ist.
intvalue	Enthält den Attributwert, falls der Attributstyp entweder Integer oder Boolean ist.
urlvalue	Enthalt die URL für den Attributsttyp URL.

Tabelle	civersion
Beschreibung	Hält alle Attributswerte einer Version eines bestimmten CIs zusammen.
id	Versions ID
ci	Diese Version gehört zu folgender CI. Referenz auf die Tabelle ci.
state	Die CI ist in dieser Version in folgendem Zustand. Referenz auf die Tabelle cistates.
author	Diese Version erstellte folgende Person. Referenz auf die Tabelle person.
created	Zeitpunkt der Erstellung dieser Version.

Tabelle	cirelation
Beschreibung	Beschreibt die Beziehung zwischen zwei CIs.
id	Relations ID
src	Anfangspunkt der Beziehung dieser Relation. Referenz auf die Tabelle ci.
dest	Endpunkt der Beziehung dieser Relation. Referenz auf die Tabelle ci.
comment	Beschreibung dieser Beziehung zweier CIs.
depends	Ist diese Beziehung eine Abhängigkeitsbeziehung?
ispart	Ist diese Beziehung eine “Teil von”-Beziehung?
linked	Sind die beiden CIs technisch miteinander verbunden?

Tabelle	cistates
Beschreibung	Beschreibt einen Zustand, in dem sich ein CI befinden kann.
id	Zustands ID
name	Name dieses Zustandes.
description	Beschreibung dieses Zustandes.
obsolete	Ein CI in diesem Zustand ist nicht mehr vorhanden oder nie mehr einsatzfähig.
damaged	Ein CI in diesem Zustand ist beschädigt und benötigt eine Reparatur.

Tabelle	cichange
Beschreibung	Beschreibt die Veränderung eines CIs von der aktuellen Version zu einer neuen Version.
id	Änderungs ID
owner	Die Person, welche diese Änderung durchgeführt hat. Ist für diese Veränderung verantwortlich. Referenz auf die Tabelle person.
oldversion	Die Ausgangsversion. Referenz auf die Tabelle civersion.
newversion	Die neue Version, welche durch diese Änderung erstellt wurde. Referenz auf die Tabelle civersion.
priority	Priorität dieser Änderung
executiontime	Zeitpunkt für die Durchführung dieser Änderung.
comment	Beschreibung dieser Veränderung.
done	Ist diese Änderung schon implementiert?
partof	Diese Änderung ist Teil folgender Aufgabe. Referenz auf die Tabelle task.

Tabelle	task
Beschreibung	Beschreibt eine Aufgabe, die es zu erledigen gilt. Ein Task kann mehrer Änderungen an CIs beinhalten.
id	Task ID
leader	Verantwortlicher für diese Aufgabe. Referenz auf die Tabelle person.
description	Beschreibung der Aufgabe
duedate	Termin für diese Aufgabe
alarm	Zeitpunkt für den Alarm zur Errinnerung an diese Aufgabe
priority	Priorität dieser Aufgabe
progress	Prozentualer Fortschritt dieser Aufgabe
done	Ist diese Aufgabe erledigt?

Weil die Software von mehreren Personen mit unterschiedlichen Aufgaben und Berechtigungen bedient werden sollen, stellt das Datenmodell auch Tabellen zur Beschreibung der Personen und ihren Berechtigungen zur Verfügung.

Tabelle	person
Beschreibung	Beschreibt eine Person, welche Zugriff auf diese Software erhält.
id	Personen ID
username	Benutzername zum Anmelden an das System.
password	Passwort für die Authentifizierung.
lastname	Nachname der Person.
firstname	Vorname der Person.
email	eMailadresse für diese Person.
telnr	Telefonnummer

Tabelle	websession
Beschreibung	Eintrag für eine authentifizierte Sitzung.
key	Sitzungsschlüssel
user	Benutzer dieser Sitzung. Referenz auf die Tabelle person.
logon	Zeitpunkt der Anmeldung
lastuse	Zeitpunkt der letzten Aktivität dieser Sitzung.
active	Ist diese Sitzung noch vorhanden oder schon beendet?
ipaddr	IP Adresse des Benutzers.

Tabelle	role
Beschreibung	verfügbare Rollen für die Benutzer. Rollen enthalten die notwendigen Berechtigungen.
id	Rollen ID
name	Name der Rolle.
description	Beschreibung der Rolle.

Tabelle	userrole
Beschreibung	Welcher Benutzer besitzt welche Rollen?
user	Referenz auf die Tabelle person.
role	Referenz auf die Tabelle role.

Tabelle	permission
Beschreibung	Berechtigung einer Rolle über ein Objekt.
id	Berechtigungs ID
role	Beschreibt die Berechtigung dieser Rolle. Referenz auf die Tabelle role.
object	Das Objekt der Berechtigung.
create	Darf die Rolle dieses Objekt erstellen?
read	Darf die Rolle dieses Object lesen?
write	Darf die Rolle dieses Objekt verändern?
delete	Darf die Rolle dieses Objekt löschen?

B.2 UML Klassendiagramme

Zusätzlich zum Datenbankschema ist für das Verständnis des Programmablaufs auch die Struktur der verwendeten Klassen sehr sinnvoll. Und für die Darstellung von Klassen hat sich der UML-Standard[1] (vgl. auch dazu [BHK04]) durchgesetzt.

B.2.1 ConfDB

Zuerst sollen die zentralen Klassen für das Programm ConfDB dargestellt werden. Die Abbildung B.2 auf der nächsten Seite zeigt dabei eine vereinfachte Darstellung der Klassen, um einen Überblick über die Aufgaben der Klassen zu geben. Jedoch sind die Details der Implementierung entfernt, weil diese besser anhand des Quellcodes verfolgt werden können.

Hier sind noch die Kurzbeschreibungen der einzelnen Klassen:

- **ci** Diese Klasse beschreibt ein Configuration Item und verwaltet auch die verschiedenen Versionen eines Configuration Items.
- **schema** Und diese Klasse beschreibt ein Schema für ein Configuration Item. Alle Informationen über die Attribute und ihre Datentypen werden in dieser Klasse festgehalten und verwaltet.
- **attribute** Die einzelnen Attribute mit ihren Werten eines Configuration Items werden in dieser Klasse zusammengefasst. Damit wird der Zugriff auf ein Attributwert erheblich vereinfacht.
- **person** Die Klasse person fasst alle Informationen einer Person bzw. Benutzers zusammen. Sie wird auch zur Autentifizierung und Autorisierung der Benutzer benutzt.
- **cilist** Das Ergebnis einer Suche nach Configuration Items ist eine Liste dieser Items. Diese Klasse stellt eine Liste von Configuration Items, welche mit den vorgegebenen Methoden zur Suche erstellt werden können.

[1] UML-Standard: `http://www.uml.org/`

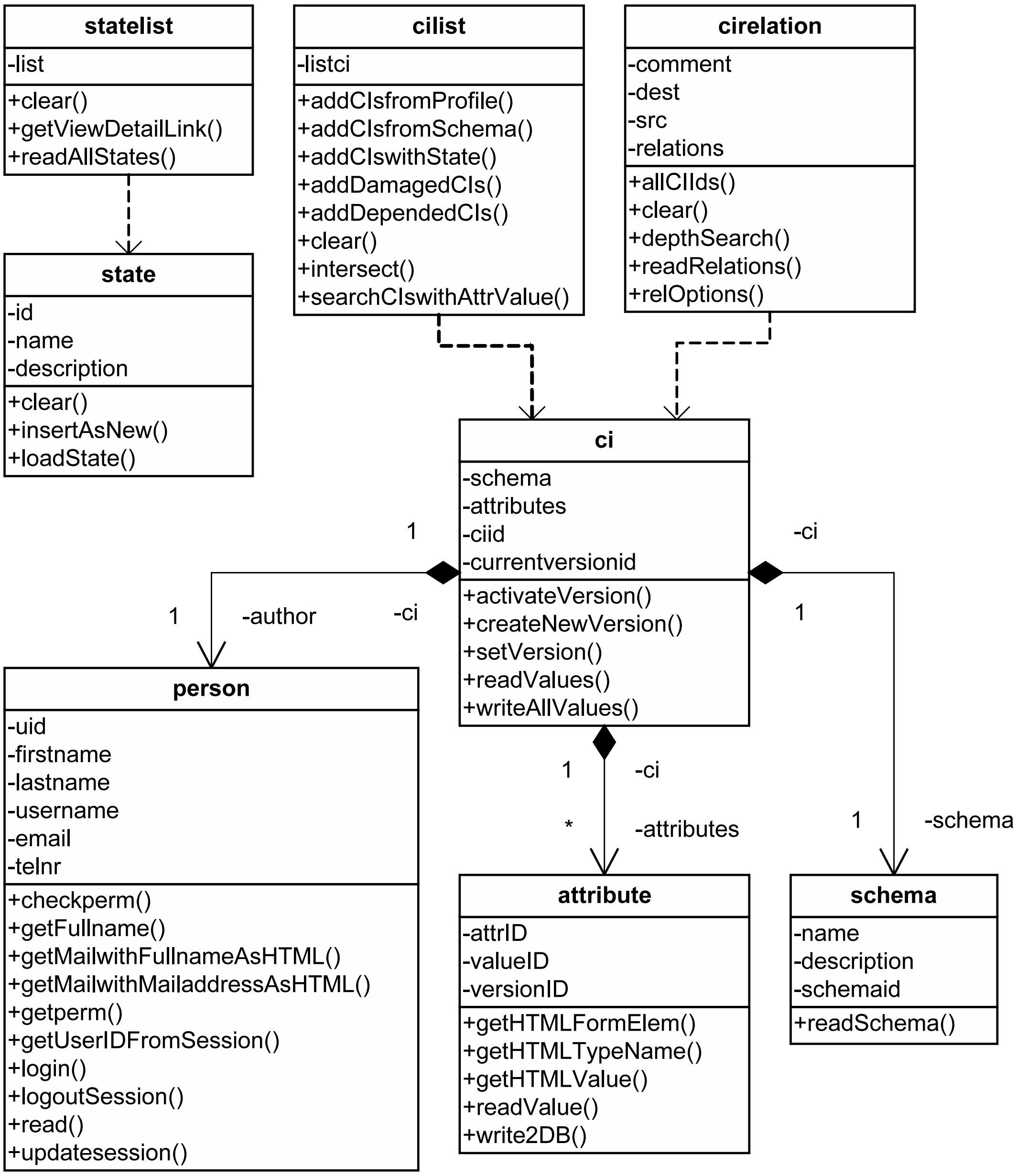

Abbildung B.2: UML-Diagramm der Klassen in ConfDB

- **cirelation** Die Beziehungen zwischen den Configuration Items stellt diese Klasse dar.

- **state** Jedes Configuration Item besitzt einen frei wählbaren Zustand. Alle Informationen über einen Zustand werden in dieser Klasse zusammengefasst.

- **statelist** Und diese Klasse liefert eine Liste aller Zustände.

B.2.2 iCal Export

Eine wichtige Anforderung an das Programm ConfDB war die Integration in die vorhandene Infrastruktur. Und weil das Management vorallem die Aktivitäten organisiert und plant, ist die Zusammenarbeit eines Terminplaners mit dem Programm ConfDB eine wichtige Funktion. Für den Datenaustausch zwischen den Terminkalenderprogrammen der verschiedenen Hersteller haben sich verschiedene Standards etabliert und der wichtigste davon ist das sogenannte “iCal Format”, welches in der RFC 2445[2] definiert wird.

Nach dieser Spezifikation ist der iCal-Export im Programm ConfDB implementiert worden. Die Implementierung verfügt aber nur über die notwendigen Klassen und Funktionen, so daß ein sinnvoller iCal-Export in ConfDB realisiert werden kann. Eine Übersicht über die verfügbaren Klassen bietet das Diagramm B.3 auf der nächsten Seite an.

Für die Einhaltung der Spezifikation ist die Klasse “component” zuständig, welche eine Komponente nach der iCal-Spezifikation implementiert. Alle anderen Klassen sind von dieser Grundklasse abgeleitet und gehorchen damit auch dem Standard.

Die Klasse “task” gehört aber im strengen Sinne nicht zum iCal-Standard, sondern bietet eine Fassade des iCal-Exports an. Diese Klasse bildet die Informationen eines Tasks, welches im Programm ConfDB benutzt wird, auf die beiden iCal-Komponenten “Event” und “ToDo” ab. Leider unterstützen nicht alle Terminplaner Events oder ToDos, so daß für eine breite Unterstützung aller Programme jeder Task in ConfDB als Event und auch als ToDo exportiert wird.

Damit der Anwender auch rechtzeitig an einen Task errinnert wird, benutzen beide Komponenten Event und ToDo eine Alarm-Komponente. In der vorliegenden Implementierung wird dieser Alarm durch das Versenden einer eMail an den Betroffenen realisiert. Falls eine andere Methode zur Erinnerung gewünscht wird, so können weitere Klassen nach der RFC 2445 implementiert werden. In Kapitel 4.6.6 der RFC werden folgende Möglichkeiten angeboten:

- **Audio** errinnert über einen frei wählbaren Ton an einen Termin.
- **Display** zeigt eine Nachricht für die Erinnerung.
- **eMail** versendet eine eMail an die betroffene Person.
- **Procedure** führt ein beliebiges Programm zur Erinnerung aus.

Fazit Die Implementierung des iCal-Exports ist trivial, und kann damit jederzeit sehr einfach an die eigenen Wünsche angepasst werden. Diese Einfachheit ist zum Teil auch dem iCal-Standard zuzurechnen, weil das Format sehr übersichtlich und verständlich aufgebaut und erklärt ist. Leider erfordert die unterschiedlichen Implementierungen, welche teilweise auch nicht vollständig den iCal-Standard unterstützen, ausführliche Tests, ob die Informationen auch korrekt interpretiert werden.

[2] RFC 2445: `http://www.faqs.org/rfcs/rfc2445.html`

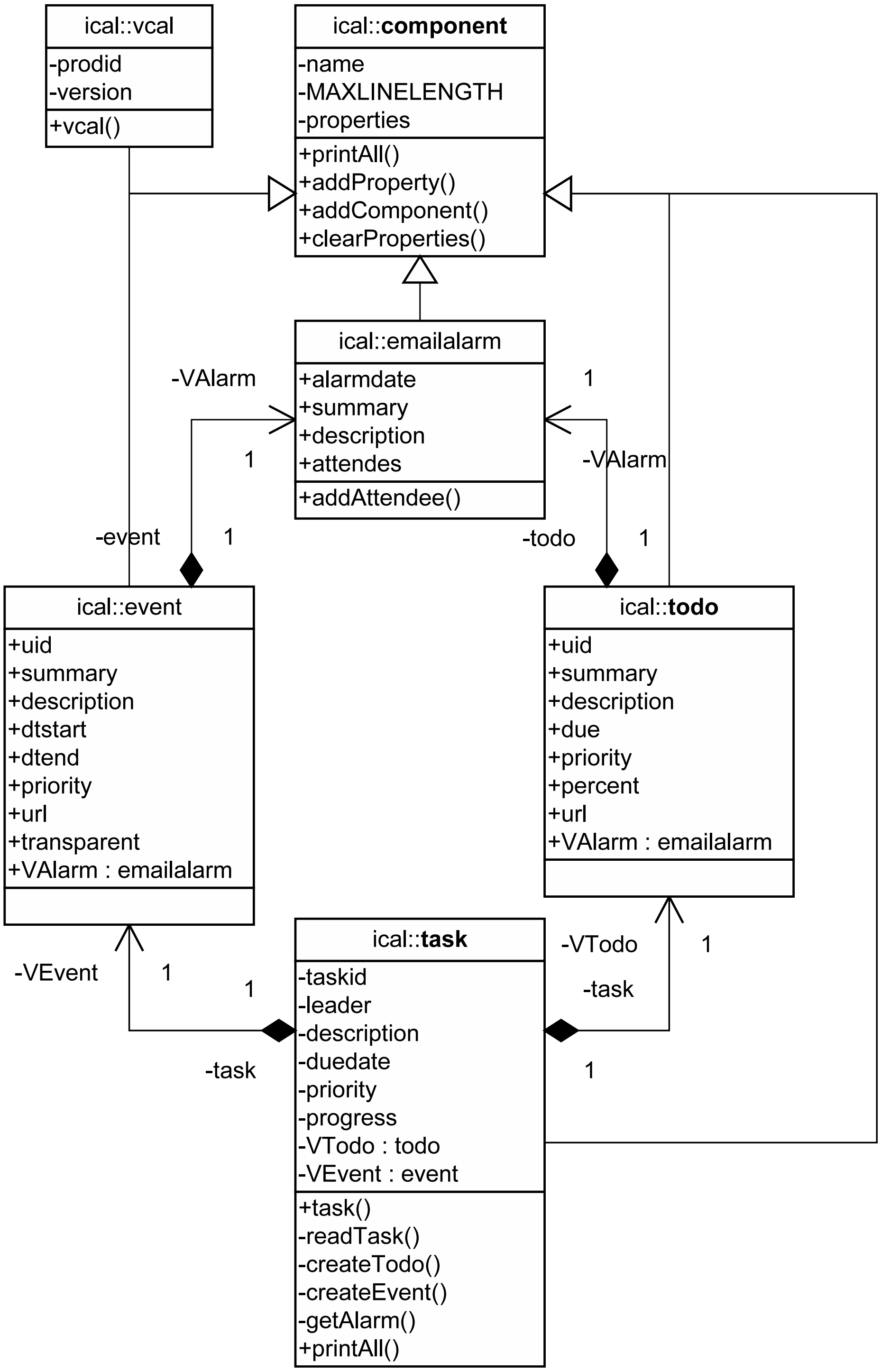

Abbildung B.3: UML-Diagramm der Klassen für den iCal-Export

Abkürzungsverzeichnis

AAL ATM Adaption Layer

ACID Atomicity, Consistency, Isolation and Durability

ATM Asynchronous Transfer Mode

BSC Balanced Scorecard

CAB Change Advisory Board

CCTA Central Computer and Telecommunications Agency

CI Configuration Item

CLIP Classical IP and ARP over ATM

CMDB Configuration Management Database

CSF Critical Success Factor

DHCP Dynamic Host Configuration Protocol

DHS Definitive Hardware Store

DSL Definitive Software Library

EDV Elektronische Datenverarbeitung

FAQ Frequently Asked Questions

FSC Forward Schedule of Changes

HiWi wissenschaftliche Hilfskraft

IP Internet Protocol

IT Informationstechnologie

ITIL IT Infrastructur Library

J2EE Java 2 Enterprise Edition

JDBC Java Database Connectivity

KPI Key Performance Indicator

LAMP Linux, Apache, MySQL, PHP

LANE LAN Emulation

LMU Ludwigs-Maximilians-Universität

MAC Media Access Control

MIME Multipurpose Internet Mail Extension

NEIS Nagios Extended Information System

OGC Office of Government Commerce

QoS Quality of Service

RfC Request for Change

TU Technische Universität

UML Unified Modeling Language

URL Universal Resource Locator

VLAN Virtual Local Area Network

WWW World Wide Web

Literaturverzeichnis

[AlCh 03] ALEXANDER ILIC und CHRISTIAN LIEB: *SEP: Konzeption und Realisierung einer sicheren Multiboot- und Backup-Infrastruktur*. Technische Universität München, 2 2003.

[BHK04] BORN, MARC, ECKHARDT HOLZ und OLAF KATH: *Softwareentwicklung mit UML 2*. Addison Wesley Verlag, 2004.

[BKP 02] BON, JAN VAN, GEORGES KEMMERLING und DICK PONDMAN: *IT Service Management; eine Einführung*. Van Haren Publishing, ISBN 90-806713-5-5, 2002.

[Hell 01] HELLER, U.: *Process-oriented Consistency-based Diagnosis*. Doktorarbeit, Technische Universität München, 2001.

[ITIL 00] OFFICE OF GOVERNMENT COMMERCE (OGC) (Herausgeber): *Service Support*. IT Infrastructure Library (ITIL). The Stationary Office, Norwich, UK, 2000.

[kaea 01] KEMPER, ALFONS und ANDRÉ EICKLER: *Datenbanksysteme, Eine Einführung*. Oldenbourg Verlag München Wien, 4 Auflage, 2001.

[PACL98] ALBITZ, PAUL und CRICKET LIU: *DNS and Bind*. O'Reilly and Associates, Inc, 3 Auflage, 1998.

[Pill 97] PILLER, FRANK T.: *Das Produktivitätsparadoxon der Informationstechnologie*. Technischer Bericht, 1997.

[PRW 03] ARNOLD PICOT, RALF REICHWALD und ROLF T. WIGAND: *Die grenzenlose Unternehmung*. Gabler Verlag, ISBN 3-409-52214-X, 5 Auflage, 2003.

[Sage 05] SAGER, F.: *Strategien und Konzepte für Configuration Management in Kooperation mit Microsoft Deutschland*. April 2005.