TECHNISCHE
UNIVERSITÄT
MÜNCHEN

INSTITUT FÜR INFORMATIK

Diplomarbeit

Konzept zur Einbettung eines bestehenden Netzdokumentationssystems in eine integrierte Managementumgebung

Jürgen Prusseit

TECHNISCHE
UNIVERSITÄT
MÜNCHEN

INSTITUT FÜR INFORMATIK

Diplomarbeit

Konzept zur Einbettung eines bestehenden Netzdokumentationssystems in eine integrierte Managementumgebung

Bearbeiter	: Jürgen Prusseit
Aufgabensteller	: Prof. Dr. Heinz-Gerd Hegering
Betreuer	: Dr. Sebastian Abeck
	: Dipl.-Inform. Kirsten Heiler
Abgabedatum	: 15. August 1995

Diese Arbeit ist aus einer Kooperation der BMW AG mit dem Münchner Netzmanagement Team (MNM-Team) entstanden, das sich unter der Leitung von Prof. Dr. H.-G. Hegering aus Wissenschaftlern der beiden Münchner Universitäten und des Leibniz Rechenzentrum der Bayerischen Akademie der Wissenschaften zusammensetzt.

An dieser Stelle möchte ich mich bei meinen Betreuern, Frau Kirsten Heiler und Herrn Dr. Sebastian Abeck und bei BMW Herrn Thomas Schmidl bedanken, die mich bei dieser Arbeit betreut und mit ihrer Kritik unterstützt haben.

Ein weiterer Dank geht an Dieter Bertram und Karl Ewald für die kritische Durchsicht und Kommentierung meines Manuskripts und den damit verbundenen Anregungen.

Ehrenwörtliche Erklärung

Ich versichere, daß ich diese Diplomarbeit selbständig verfaßt und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 15. August 1995

....
(Unterschrift des Kandidaten)

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation

Aus der heutigen Infrastruktur eines Unternehmens sind Computersysteme nicht mehr wegzudenken. Besonders die Industrie hat den Nutzen früh erkannt und betriebseigene Netze installiert. Im Rahmen des Down-Sizings begann die Migration von den Mainframes zu Workstationnetzen, die jedoch nicht nur Vorteile brachte. Hatte man bei einem Großrechner nur ein, wenn auch großes und monolithisches System zu betreuen, so wurden die Aufgaben der Systembetreuer durch die Installation vieler untereinander kommunizierender Systeme – meist verschiedener Hersteller – immer umfangreicher. Doch nicht nur die Systeme wurden aufgeteilt. Auch Anwendungen und Daten befinden sich jetzt verteilt im Netz und beide werden immer größer im Ressourcenverbrauch. Damit erhöhen sich wiederum die Anforderungen an den Durchsatz des Netzes. Die Stabilität desselben wird ein immer größerer Faktor im Betriebsablauf.

Die Aufgabe, das Netz zu überwachen, wird in dem noch relativ neuen Zweig der Informatik, dem Netzmanagement [HeAb93], wahrgenommen.

Grundlage für das Netzmanagement sind die Informationen, die durch sogenannte „Agenten" in den aktiven Netzkomponenten dem Werkzeug „Netzmanagemensystem" zur Verfügung gestellt werden. Dieses sammelt und analysiert die Informationen. Doch nicht alle Komponenten haben einen Agenten, teils weil sie älterer Bauart sind, für die es keine Agenten gibt, teils weil sie mit der entsprechenden Erweiterung (noch) nicht ausgestattet sind. Somit sind nicht alle Komponenten des Netzes mit einem Netzmanagementsystem erfaßbar. Passive Komponenten, wie z.B. Kabel und Verteilerschänke, liegen auch nicht im Einzugsbereich dieses Systems.

Ein zusätzliches Werkzeug, das Netzdokumentationssystem, wurde schon früher zur Dokumentation der Netzinfrastruktur eingeführt und stellt so auch heute häufig noch das einzige Managementsystem dar, das einen Überblick über *alle* Netzkomponenten bietet, da das beste Netzmanagementsystem – aufgrund fehlender Voraussetzungen im Netz – versagt, alle Komponenten automatisch zu erfassen und als Netzstruktur darzustellen. Die Wandlung zu einem voll managebaren Netz ist nur unter hohem finanziellen Aufwand möglich – der Nutzen, der sich daraus ergibt, scheint sich z.Z. allerdings noch nicht zu rechnen.

In der Umgebung der Managementwerkzeuge erkennt man, daß diese heute meist eigenständige, proprietäre Anwendungen sind, die nur einen kleinen Teilbereich, oft sogar nur spezielle Produkte einzelner Hersteller abdecken. Ein Beispiel ist *CiscoWorks* der Firma Cisco, das nur für den Einsatz mit der firmeneigenen Routerpalette vorgesehen ist. Der Systemadministrator muß sich für jedes neue Produkt mit dem dafür „maßgeschneiderten" sog. Elementmanagementsystem beschäftigen.

So entstehen Probleme, die von der Nicht-Beherrschbarkeit der funktionellen Details jedes Produktes bis zur Mehrfachhaltung gleicher Daten in den Werkzeugen mit den bekannten Folgen der Redundanz und daraus folgenden Inkonsistenz reichen.

In einer integrierten Managementumgebung wird nun versucht, dieser Probleme durch Kommunikation auf verschiedenen Integrationsebenen Herr zu werden. Das noch entfernte Ziel ist, zu einer integrierten Managementlösung zu kommen, in der durch einen einheitlichen Ansatz eine heterogene Netz- und Systemumgebung administriert werden kann. In diesem Umfeld ist die vorliegende Arbeit als ein Baustein mit angesiedelt.

1.2 Aufgabenstellung und Ergebnisse

Das Konzept dieser Arbeit resultiert in einer Zweiteilung. Im ersten Teil wird eine Methodik erarbeitet, anhand der man die für die Einbettung des Dokumentationssystems nötige Gesamtsicht auf die managementrelevanten Daten erhält. Durch das beschriebene Top-Down-Vorgehen bekommt man eine klassifizierte Darstellung der Daten, anhand der Defizite im betrieblichen Datenmodell aufgedeckt werden können. Mit der so gewonnenen Gesamtsicht kann die Integration isolierter und verteilter Managementsysteme besser in Angriff genommen werden.

Der zweite Teil behandelt ein konkretes Szenario in der Betreiberumgebung der *Bayerischen Motorenwerke AG (BMW)*. Von den bestehenden Defiziten der Kommunikation zwischen dem betriebseigenen Netzdokumentationssystem Cinema und der Managementplattform NetView for AIX von IBM werden konkrete Integrationsanforderungen abgeleitet und spezielle Lösungsmöglichkeiten vorgestellt. Ein wichtiges Ergebnis dieses Teils ist, daß erstmalig ein automatischer Konsistenzvergleich der Daten aus Cinema mit den über NetView gesammelten Komponentendaten erfolgen kann. Auf der Seite des Netzmanagementsystems sind jetzt zusätzliche Komponentendaten verfügbar. Mit dieser Arbeit wurde damit die Integration bestehender Systeme anhand von konkreten Problemstellungen vorangetrieben.

1.3 Überblick über die Arbeit

Nach der Beschreibung des Vorgehens in Kapitel 2, wird Methodik für die Entwicklung eines Datenbestandes wird in Kapitel 3 beschrieben. Diese wird ergänzt durch die beiden Beispiele in Kapitel 4, in denen erst ein exemplarischer Datenbestand aufgebaut und dann mit dem Szenario Fehlermeldeverfahren die Anwendbarkeit dieses Datenbestandes überprüft wird.

In Kapitel 5 findet eine Differenzierung der Daten und Aufgaben eines Netzdokumentationssystems und eines Netzmanagementsystems im allgemeinen statt. Abgeschlossen wird es durch die Beschreibung verschiedener Integrationstechniken, die im nächsten Kapitel bei den Lösungsansätzen Anwendung finden.

In Kapitel 6 werden die allgemein gehaltenen Beschreibungen des letzten Kapitels durch die Analyse der bei BMW eingesetzten Netzdokumentations- und Netzmanagementsysteme ergänzt. Abschließend werden konkrete Defizite und Vorschläge für deren Lösungen benannt.

Während die Anbindungsmöglichkeiten der Systeme auf ihre Mächtigkeit untersucht wurden, sind zwei Prototypen entstanden, die in Kapitel 7 beschrieben sind.

Abgeschlossen wird die Arbeit mit dem Ausblick in Kapitel 8.

Teil I

Methodik für die Entwicklung eines Datenbestandes

Kapitel 2

Vorgehensweise

Die Organisation der Daten ist wichtig bei der Betrachtung der Verbindungsmöglichkeiten von Management- und Dokumentationssystem. Können die Daten aus *beiden* Systemen nicht *ein* Gesamtmodell bilden, ist eine Anbindung nicht möglich.

Somit muß der Datenbestand schon bei der Konzeption unter einer einheitlichen Betrachtung erstellt werden. Ist schon in der Anfangsphase die Kommunikation zu anderen Datenbanken und Managementsystemen eingeplant, kann sich der manuelle Pflegeaufwand erheblich reduzieren. Alleine durch die Kombination aus Daten der Komponenten-MIBs und aus dem Netzmanagementsystem lassen sich die Einträge im Dokumentationssystem in weiten Teilen überprüfen und ergänzen.

Doch erst mit der Beachtung der von den anderen Informationsquellen vorgegebenen (meist statischen) Datenformate läßt sich die Kombinierung mit *geringem* Aufwand realisieren.[1]

Mit einem einheitlich gestalteten Datenbestand erreicht man eine gesamtheitliche Betrachtung und Strukturierung der Daten.

Während in Kapitel 5 eine Unterscheidung zwischen Netzdokumentation und Netzmanagementsystem vorgenommen wird, erfolgt hier die Einordnung und Einbettung aller Daten von Netz- und Informationssystemen in ein einheitliches Datenmodell. Mit diesem kann dann eine Gesamtbetrachtung auf die bestehenden Daten gewonnen werden, die als Grundlage für Integrationen dienen können.

Der Datenbestand kann auch beispielsweise als Grundlage für ein zu evalierendes

[1] Bei diesen Anbindungen entstehen auch organisatorische und rechtliche Aspekte, die nicht zu vernachlässigen sind. So ist es im Einzelfall abzuklären, inwieweit datenschutzrelevante Belange bei der Kombination mit personenrelevanter Daten berührt werden. Die Daten müssen somit besonders geschützt werden, da bei unbefugtem Gebrauch Informationen aggregiert werden können, die Personen schaden können. Doch da dies nicht Bestandteil dieser Arbeit ist, wird hierauf nicht näher eingegangen.

kommerzielles Dokumentationssystem dienen, oder als Basis für ein zu realisierendes eigenständiges System. Dabei kann der Datenbestand auch in einem verteilten Datenbanksystem sein, auch wenn im weiteren immer von einem System ausgegangen wird.

Eine hierarchische Darstellung zwischen den Anwendungen und der Datenbank wird zunächst motiviert und dargestellt und dann im folgenden zwei Methodiken beschrieben, anhand derer ein Vorgehen zum Erstellen eines Datenbestandes abgeleitet wird. Der erste Ansatz ist ein naiver Bottom-Up-Ansatz, der gewählt wurde um eine Worst-Case Betrachtung vorzustellen. An diesem Ansatz sind die Nachteile dieses Vorgehens gut nachzuvollziehen.

Im zweiten Teil wird dann ein strukturierter Top-Down-Ansatz gewählt, der bessere Eigenschaften bezüglich der Stabilität des Modells aufweist. Dieser wird dann im Gegensatz zum ersten Ansatz weiter vertieft.

2.1 Das Ebenenmodell

Die Gründe für die Datenhaltung in einem Datenbanksystem sind im Grunde dieselben, die auch die Problematik im integrierten Management bestimmen. Diese sind:

- Vermeidung der Verteiltheit von Daten:
  Jedes Anwendungsprogramm braucht zur Abarbeitung Daten. Für eine isolierte Einzelanwendung liegen diese Daten oft in einem eigenen Dateiformat vor. Eine Erweiterung der Daten oder die gemeinsame Nutzung mit einer anderen Anwendung kann nur unter hohem Programmieraufwand stattfinden. Mit einem Datenbanksystem ist über eine definierte Schnittstelle der Zugriff einfach und eine Erweiterung des Datenmodells ohne hohen Aufwand möglich.

- Vermeidung von Redundanz von Daten:
  Werden von Einzelprogrammen Daten, z.B. Adreßdaten von Personen, benötigt, müssen diese im isolierten Fall mehrfach gehalten werden. In einem Datenbanksystem sind diese Daten zentral und in einfacher Ausführung vorhanden. Durch die verschiedenen Sichten, die den Anwendungsprogrammen gewährt werden (Schicht „externe Modelle“ in Abb. 2.1, Erläuterung auf Seite 2.1) wird die richtige Kombination von Daten dem Anwendungsprogramm zur Verfügung gestellt.

- Vermeidung von Inkonsistenzen:
  Bei der Einzeldarstellung ist es nur unter hohem Aufwand zu vermeiden, daß identische Daten in verschiedenen Anwendungen auch identisch bleiben.

Bei einer zentralen Datenhaltung ist eine Änderung für alle Anwendungen gültig.

- Flexibler Gebrauch von Daten:
  Mit einer zentralen Datenhaltung ist es einfacher neue Anwendungen mit einem neuen View auf eine Kombination von vorhandenen und neuen Daten zu betreiben.

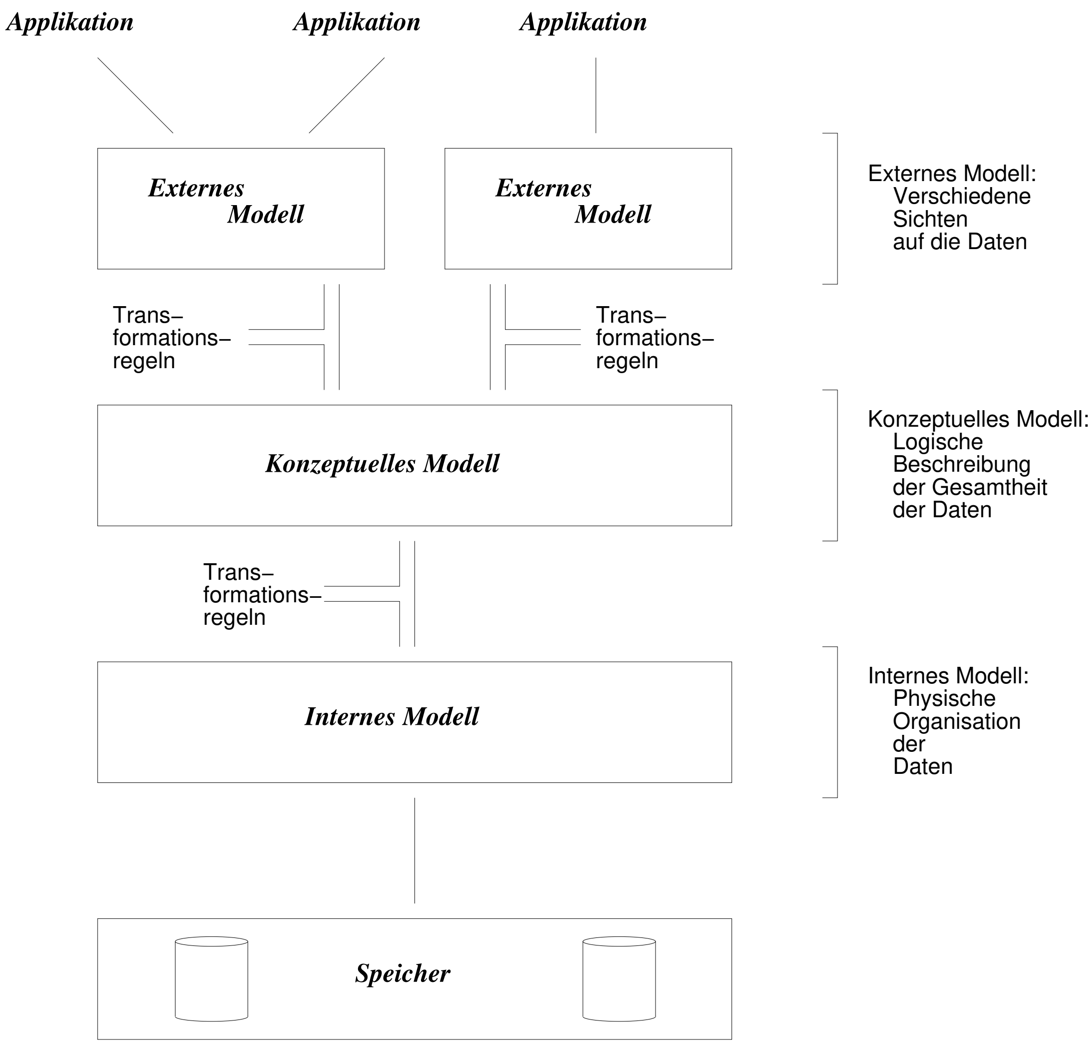

Abbildung 2.1: Architektur eines Datenbanksystems – Ebenenhierarchie

Mit der Trennung von Datenhaltung und Anwendung erreicht man somit eine physische Datenunabhängigkeit der Anwendung. Diese ist auch dann voll lauffähig, wenn eine Reorganisation der Daten in Hinsicht auf Zugriffsoptimierung stattgefunden hat.

Diese Trennung findet sich auch in dem folgenden Modell wieder. Eine begleitende Darstellung ist in Abbildung 2.1. In der Modellierung nach [ANS75] unterscheidet man drei Modelle – dem externen, konzeptionellen und internen Modell. Im folgenden sind diese drei Modelle näher erläutert:

- Externes Modell:
  Jeder Benutzer (egal, ob dies ein System oder ein Anwender ist) soll nur den Teil der Daten sehen, der für ihn von Bedeutung ist. Jede eigene Sicht auf die Daten werden in einem spezifischen externen Modell dargelegt. Auf ein Dokumentationssystem übertragen, kann man hier auch einzelne Aufgabenbereiche als Views betrachten.

- Konzeptionelles Modell:
  Das konzeptionelle Modell beschreibt die Gesamtansicht all der Daten, die in der Datenbank verwaltet werden sollen. Hier werden die Entities, Relationen und die Attributierung der Entities spezifiziert.

  Das konzeptuelle Modell beschreibt die Daten auf logischer Ebene, unabhängig von der tatsächlichen Realisierung in der Datenbank. Es beschreibt damit Bezugspunkte für alle Anwendungen und ist die Grundlage für die verschiedenen Sichten. Auf dieser Ebene ist auch die Kontrolle der Daten geregelt, indem die zulässigen Operationen, die Zugriffsrechte, Integritätsbedingung und Zuständigkeiten definiert werden.

  Das Modell ist ein (relativ) stabiler Bezugspunkt. Bei sorgfältiger Planung wird sich die Gesamtsicht weit weniger schnell ändern als die Views der Anwendungen.

- Internes Modell:
  In welcher Form die Daten, die durch das konzeptuelle Modell beschrieben wurden, im Speicher letztendlich abgelegt werden, ist in diesem Modell festgelegt. Hier erfolgt die Definition der physischen Datenorganisation, aus der sich die beschriebenen Entities und Relationen ableiten lassen. Das interne Modell enthält alle Informationen über den Aufbau der abgespeicherten Daten, deren Speicherorganisation, Zugriffspfade, usw.

  Von diesem Modell hängt im wesentlichen die Leistungsfähigkeit des Gesamtsystems ab. D.h. hier müssen alle Faktoren, die auf die Antwortzeit Einfluß haben, berücksichtigt werden. Beispielhaft dafür sind: Systemeigenschaften, wie Zugriffsgeschwindigkeiten, Speicherausbau; physische Verteilung der Daten bei verteilten Datenbanken; Optimierungsmöglichkeiten.

Der später erarbeitete Datenbestand ist im konzeptionellen Modell anzusiedeln, da gerade auf dem diesem die externen Modelle mit ihren speziellen Sichten aufbauen und die Stabilität des Modells Grundvoraussetzung ist.

Als letztes müssen in diesem Modell noch die Transformationsregeln betrachtet werden. Diese besagen, auf welche Art ein bestimmtes Objekt eines Modells aus einem oder mehreren Objekten eines tieferliegenden Modells gebildet werden soll.

Die Transformationsregeln externes/konzeptuelles Modell legen fest, auf welche Art die Entities und Relationen des externen Modells aus Entities und Relationen des konzeptuellen Modells zusammengesetzt werden. Ein Beispiel hierfür sind die verschiedenen Abbildungen von Personendaten auf die externen Modelle. So werden zwar alle Personendaten im konzeptuellen Modell modelliert, im externen werden dann aber je nach Anwendung nur z.B. Name und Telephonnummer, oder aber Name und Adresse zur Verfügung gestellt.

In den Transformationsregeln konzeptuelles/internes Modell werden die Zusammenhänge zwischen den physisch gespeicherten Datensätzen und den Entities und Relationen des konzeptuellen Modells festgelegt. Dabei sind hier Entscheidungen zu treffen, inwieweit oft benutzte Attribute einer Entity aufgrund der Zugriffsoptimierung getrennt von weniger benutzten gespeichert werden sollen.

2.2 Bottom-Up-Ansatz

In diesem Ansatz sind die Ausgangspunkte die in einem Unternehmen bereits bestehenden (meist flachen) Datenbestände, auf die die proprietären Werkzeuge aufsetzen. Diese Daten setzen sich aus den eingesetzten Element-Management-Systemen, Trouble-Ticket-Systemen, Buchhaltungssystemen, Netzdokumentationssystem, usw. zusammen.

Nach Auswahl der für den Gebrauch wichtigsten Daten werden diese nun in einen neuen Datenbestand importiert. Dabei findet man in der erstellten Datenbank nur Kopien der von den Werkzeugen gehaltenen Daten. Die Werkzeuge versorgen also parallel zu ihren eigenen Datenbeständen auch dieses neue System mit aktuellen Daten. Durch die Verbindung ähnlicher oder in logischer Verbindung stehender Datensätze können dann neue Strukturen aufgebaut werden. Dazu wird ein neues Werkzeug benötigt, das basierend auf den kombinierten Datenbeständen neue oder auch mächtigere Views ermöglicht (siehe Abbildung 2.2).

Die Vor- und Nachteile dieser Methode werden im folgenden erläutert.

Vorteile:

1. Mit diesem Ansatz bekommt man gegenüber den anderen Ansätzen relativ schnell Teillösungen. Dies ist darauf begründet, daß man nur die Schnittstellenbeschreibungen der Werkzeuge benötigt, um die Daten übernehmen zu können.

   Für die Implementierung brauchen nur die Werkzeuge betrachtet werden,

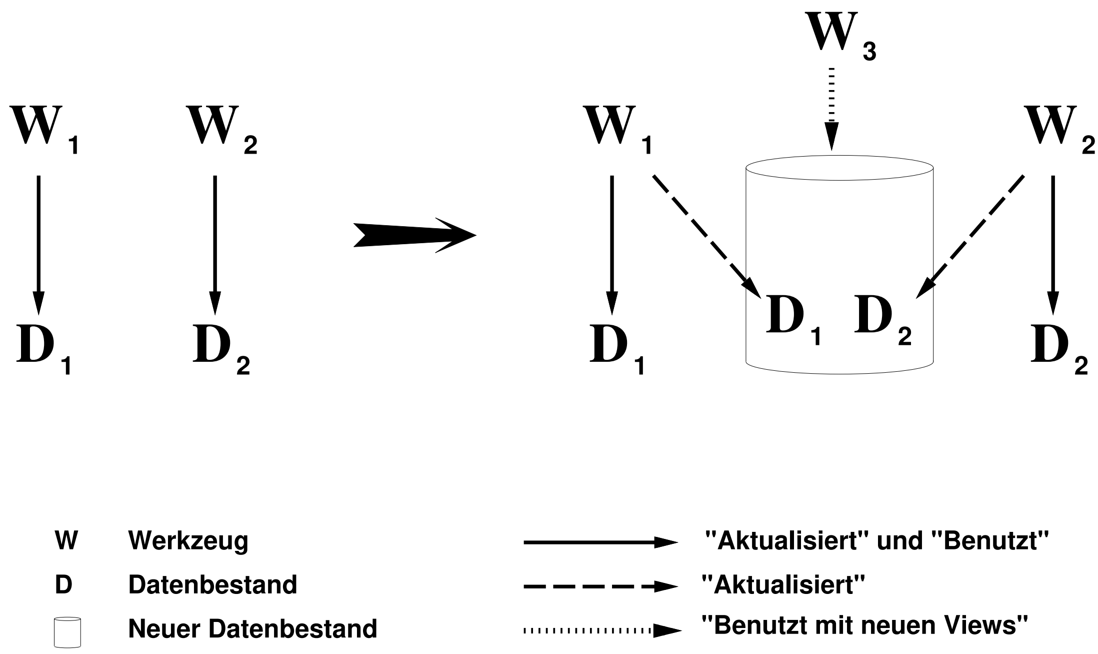

Abbildung 2.2: Aufbau des Datenbestandes nach Bottom-Up-Ansatz

die sich in den Datenbeständen gegeneinander ergänzen oder denen eine strategische Bedeutung für die Gesamtansicht der Daten zukommt.

2. Die bereits bestehenden Vorgehensweisen in Arbeitsabläufen können beibehalten werden, da keine Änderung an der Funktionalität der Werkzeuge selbst vorgenommen wird.

3. Mit dem neuen Datenbestand ist ein werkzeugübergreifendes Agieren möglich, da alle Daten in einer Datenbank gehalten werden und somit eine Gesamtsicht auf die Daten möglich ist. Somit können bestehende Arbeitsabläufe durch neue Views auf die zusammengeführten Daten ergänzt und auf diese Weise optimiert bzw. vereinfacht werden.

4. Es können nun bisher nicht vorhandene Abhängigkeitsstrukturen nachgebildet werden. Zum Beispiel sind Fehlerfortpflanzungen erkennbar: Wenn vorher ein Fehler auf mehreren Element-Managementsystemen Folgefehler verursachte, können nun durch Verknüpfungen die Abhängigkeiten sichtbar gemacht und somit die Folgefehler ignoriert werden.

Nachteile:

1. Der Datenbestand ist unstrukturiert, da die Datenbetrachtung nicht unter dem Gesichtspunkt der Verwendbarkeit stattfindet, sondern einfach alle

übernommen werden. Dies bedeutet aber auch, daß auch bei gleicher Repräsentation die Daten mehrfach zu halten sind.

Bestehende Konsistenzprobleme werden so nicht gelöst, vielmehr muß das neue Werkzeug dies berücksichtigen.

2. Durch die naive Übernahme aller Daten wird der Datenbestand zu umfangreich und die Folge ist, daß überflüssige Daten gehalten werden, es somit zur Redundanz kommt und unnötig Ressourcen für die Speicherung verbraucht werden.

3. Zusätzlich zu den schon bestehenden Werkzeugen und Datenbanken muß jetzt auch noch eine neues System gepflegt werden. Der Pflegeaufwand für ein solches System ist bei Änderungen in der Werkzeugumgebung hoch, da dann einerseits die Schnittstellen und andererseits die Relationen der Einzeldatenbestände innerhalb der Datenbank angepaßt werden müssen.

4. Eine grundlegende Reorganisation der Daten ist schwierig, da durch die unterschiedlichen Sichtweisen der Werkzeuge die Darstellung innerhalb der Datenbank differieren. Es müßten also mehrere Datenmodelle in Einklang gebracht werden, um eine Reorganisation zu verwirklichen.

5. Der Arbeitsablauf wird weiterhin durch die Werkzeuge und deren Datenbestände bestimmt. Eine Optimierung des Arbeitsablaufes durch eine Anpassung der Werkzeuge und deren Daten an den optimierten Arbeitsablauf kann nicht stattfinden.

6. Die Datenbestände müssen mit jeder Ergänzung mit Daten eines weiteren Werkzeuges erweitert werden, da vorhandene Strukturen nicht unterstützt werden und somit ein neues Datenmodell mit aufgenommen und neue Verknüpfungen implementiert werden müssen.

Mit diesem Vorgehen erreicht man eine bessere Ausnutzung der vorhandenen Werkzeuge und schafft sich die Chance, neue Betrachtungsweisen auf diesen neuen Datenbestand zu gewinnen. Eine langfristige Lösung kann dieser Ansatz jedoch nicht anbieten, da der Aufwand der Pflege gegenüber den Nutzen unverhältnismäßig hoch ist.

Man nähert sich auch nicht dem integrierten Management an, da die proprietären Strukturen weiter beibehalten werden. Der einzig positive Integrationsaspekt besteht darin, daß vorher verteilte Datenbestände in einem einzigen vereint und auf diesem Aktionen (z.B. Viewbildung) ausgeführt werden können.

Kapitel 3

Top-Down-Ansatz

Der Top-Down-Ansatz liefert im Gegensatz zum Bottom-Up-Ansatz eine langfristige und stabile Lösung. Stabil heißt, daß auch mit Hinzunahme von neuen Aspekten das System nur minimal ergänzt werden muß.

In dem zu erstellenden Datenbestand werden nur die Daten aufgenommen, die für die Erbringung von Aufgaben, die später genauer benannt werden, relevant sind.

Die strukturierte Repräsentation der Daten wird durch ein einheitliches Vorgehen bei der Eruierung erreicht. Durch diese verwendungsbezogene Betrachtungsweise bekommt man eine gesamtheitliche Betrachtung und die Grundlage für ein einheitliches Modell der Daten.

Eine Methodik, die die Beziehungen zwischen zu erbringenden Diensten bis zu den Werkzeugen und deren Datenbeständen liefert, ist das Rahmenbetriebskonzept (RBK). Für eine Beschreibung diese Konzeptes sei auf die Beschreibungen von [HAWI94], [Weiß94] und [Bert95] verwiesen.

Die Begriffe Dienst, Aufgabe und Verfahren werden im folgenden im Sinne des RBK gebraucht.

Das RBK stellt eine Top-Down-Methodik vor, mit der man systematisch die Aufgaben isolieren kann. Anhand der Verfahrensbeschreibung bekommt man dann Antwort auf die hier in diesem Zusammenhang relevanten Fragen:

- Welche Informationen müssen zwischen den (Teil-)Verfahren ausgetauscht werden?
- In welchen Verfahren werden relevante Informationen gesammelt?
- In welchen Verfahren werden diese Informationen weiterverwendet?

Weiter unten liegende Ebenen sind in dieser Designphase noch nicht relevant, weil bis zu diesem Punkt die Betrachtung werkzeugunabhängig ist. Dies bewahrt auch davor, nicht relevante Daten zu beachten, da diese mit dieser Sichtweise aussortiert werden.

3.1 Aufgaben und Verfahren in der Dokumentationsumgebung

Dieser Abschnitt soll keine detaillierte Beschreibung der Aufgaben und Verfahren sein, sondern nur einen Überblick bieten, wo Einsatzbereiche eines Dokumentationssystems angesiedelt sind. Diese Beschreibung liefert den Hintergrund zum Verständnis des weiteren Vorgehen.

Die Aufgaben und Verfahren werden beispielhaft anhand von Szenarien gezeigt, von denen die wichtigsten im folgenden vorgestellt werden. Die Beschreibungen sind mit Sicht auf ein Dokumentationssystem ausgelegt und stellen somit die Anwendung dieses Systems in einer Betreiberumgebung dar.

Die Szenarien sind auf der Aufgabenebene des RBKs angesiedelt. Mit den Szenarien werden auch die für diesen Bereich typischen Verfahren genannt:

1. Aufgabenfeld: Installationsmanagement:
   Verfahren:
   - Planung neuer Installationen
   - Inventarisierung: Erfassen aller bestehenden und neuen Geräte und Bereitstellung der für die Bestandsführung relevanten Daten, wie z.B. Inventarnummer, Kaufpreis, Restwert, usw.
2. Aufgabenfeld: Konfigurationsmanagement:
   Verfahren:
   - Darstellung des Netzes
   - Adreßvergabe
   - Erfassen der Verbindungen und deren Verlauf (leitungsbezogen)
   - Dokumentation der Domänen (Subnetzbildungen, Zuordnung der Komponenten zu Organisationseinheiten, usw.)
   - Dokumentation der Konfiguration von Komponenten (Hardwareausstattung, Softwareausstattung)
   - Zuordnung von Benutzern/Systembetreuern zu den Komponenten; usw.
3. Aufgabenfeld: Fehlermanagement:
   Verfahren:

- Erfassen von Fehlern und deren Dokumentation
- Lokalisierung des Verantwortlichen
- logische Verknüpfung der behobenen Fehler an die Komponenten als Referenzlösungen

4. Aufgabenfeld: Accounting:
   Verfahren:
   - Zuordnung von Abrechnungsdaten zu Verbrauchern (sowohl Komponenten als auch Benutzer)
   - Überwachung von Kontingenten, Generieren von Abrechnungen, usw.
5. Aufgabenfeld: Umzugsmanagement:
   Verfahren:
   - Organisation eines Umzuges
   - Sicherstellung, daß beim Umzugsziel alle benötigten Installationen bereitgestellt werden (Netzanschlüsse, ausreichende Kapazitäten in der Netzinfrastruktur, wie Übertragungskapazitäten, usw.)
   - Zeitplanung; usw.
6. Aufgabenfeld: Qualitätsmanagement:
   Verfahren:
   - Erstellen von Langzeitstatistiken: Erfassen von verdichteten Statistikdaten, wie Ausfallhäufigkeit und Fehlerstatistiken (u.a. als Kaufentscheidung, hat die Komponente die gewünschte Qualität?), durchschnittliche Auslastung von (Teil-)Netzen (zur Bestimmung bzw. Vorbeugung von Engpässen), usw.

Da sich diese Szenarien jeweils aus mehreren Teilbereichen zusammensetzen, sind Überschneidungen der benötigten Informationen vorhanden (z.B. wird die Zuordnung der Komponenten zu Abteilungen sowohl in der Inventarisierung als auch im Konfigurationsmanagement benötigt). Doch die verschiedenen Szenarien blicken aus verschiedenen „Views“ auf diese Informationen, so kann die Abteilungszugehörigkeit von Komponenten bei der Inventarisierung als Grundlage für den Bestand einer Abteilung dienen, während beim Konfigurationsmanagement dies der Ausgangspunkt für die Bildung von Subnetzen sein kann.

So sind die Aufgaben also unterschiedlicher Natur, werden aber teilweise auf Basis von identischen (Teil-)Verfahren zur Informationsbeschaffung erbracht.

Beispiele für identische Verfahren, die immer wieder in verschiedenen Aufgaben mit unterschiedlicher Zielsetzung benutzt werden, sind:

- Identifizierung: Das Finden von Komponenten anhand eindeutiger oder zusammengesetzter Schlüssel
- Benennung: Eindeutige Vergabe von identifizierenden (Schlüssel-)Attributen
- Betreuer/Benutzer: Zuordnung von Administratoren und Anwendern zu den Komponenten
- Lokalisation: Zuordnung von Komponenten zu Standorten oder im weiteren Sinne betrachtet auch Domänen
- Darstellung der Komponenten und Verbindungen in der physischen Netzstruktur
- Wegeverfolgung: Über welche Komponenten und Verbindungen sind zwei (End-)Geräte verbunden?

Auf eine Zuordnung zu den vorher beschriebenen Aufgaben wurde verzichtet, da diese Verfahren elementar sind.

Um nun einen konkreten Datenbestand aus diesen Verfahren zu entwickeln, muß noch weiter differenziert werden. Dazu wird im folgenden eine Dreiteilung motiviert, anhand der dann die Klassifizierung vorgenommen wird.

Um den sukzessiven Aufbau des Datenbestandes zu ermöglichen, ist es erforderlich, verschiedene Ausbaustufen zu definieren. Anhand dieser Ausbaustufen ist es dann möglich, bedarfsgerecht die Umsetzung dieses Konzepts vorzunehmen. Aufgrund des umfassenden Modells ist eine spätere Erweiterungsmöglichkeit dann gewährleistet.

Um eine starke Polarisation zwischen den Gruppen zu erzwingen, wurden diese

- minimaler Datenbestand oder synonym Grunddatenbestand
- erweiterter Datenbestand

und

- nice to have

genannt. Im folgenden werden diese genauer erläutert.

3.2 Minimaler Datenbestand / Grunddatenbestand

Für diesen Datenbestand werden drei Prämissen definiert:

1. Die Mehrheit der Verfahren ist ohne diese Informationen nicht durchführbar.

2. Für im Unternehmen strategisch wichtige Aufgaben müssen die benötigten Informationen bereitgestellt werden. (Sind in einer Telephongesellschaft Abrechnungsdaten essentiell, so sind sie in einem lokalen, firmeninternen Netz von sekundärer Bedeutung.)

3. Ohne diese Information ist der Datenbestand unbrauchbar. (Ergeben sich aus den obigen Prämissen kein einheitliches und zusammenhängendes Modell, so wird es hier ergänzt. Beispielhaft sind hierfür noch nicht bestehende Verbindungen zwischen den Daten oder Daten, die für das Gesamtverständnis des Datenbestandes ergänzend nötig sind.)

Der minimale Bestand muß also die Informationen bieten, ohne die das System als solches nicht sinnvoll einsetzbar ist und anhand derer sich schon die wichtigsten (Teil-)Bereiche der in Abschnitt 3.1 angeführten Szenarien behandeln lassen. Als wichtigste Bereiche ergeben sich

- die Bestandsübersicht und
- die Darstellung des physischen Netzes,

da auf diese Bereiche alle anderen aufbauen.

Somit ergeben sich natürlicherweise folgende Fragen an das System:

Welche Komponenten gibt es im Netz? Über diese Frage werden die Informationen erreicht, die für Inventarisierungsfragen und Identifizierung wichtig sind.

Wo steht die Komponente? Der Standort ist hiermit identifiziert.

Wer hat mit dieser Komponente zu tun? Damit ist die Verknüpfung zu Benutzern und Betreuern impliziert.

Wie baut sich aus den Komponenten die physische Netzstruktur auf? So wird die Struktur des Netzes und der Verlauf der Verbindungen zwischen den Einzelkomponenten ermittelt.

Diese vier Fragestellungen, die gerade entwickelt wurden, werden im folgenden weiter vertieft und detailliert. Zuerst werden die Informationen beschrieben, die für die einzelnen Komponenten von Bedeutung sind. Die Erläuterungen zu der folgenden Tabelle befinden sich im Anschluß.[1]

[1]Die folgenden Tabellen sind so zu lesen, daß umfassende Punkte als Überbegriffe zu den detaillierteren Unterpunkten anzusehen sind. Anders betrachtet sind die Fragen um so konkreter, je weiter sie eingerückt sind.

Grunddatenbestand: Einzelkomponenten

(1) Suche alle Endgeräte und Netzkomponenten

Für jede einzelne Komponente

- Welche Information ist für diese Komponente generell wichtig?
  - (2) Was für ein Typ von Komponente ist es? (PC,Terminal, …)
  - (3) Welchen Herstellernamen hat die Komponente?
  - (4) Welchen eindeutigen Namen hat die Komponente im Netz?
  - (5) Welchen eindeutigen Namen hat die Komponente im Betrieb?
  - (6) Wer betreut diese Komponente?
  - (7) Wer benützt diese Komponente?
  - (8) Wo steht diese Komponente?

(1) Die folgenden Betrachtungen sollen für alle Endgeräte, aktive und passiven Netzkomponenten durchgeführt werden. Dabei sind bei den passiven Elementen nur die zu betrachten, denen eine strategische Bedeutung zukommt. Im ersten Ansatz sind dies nur Verteilerschränke für Kabelstränge.

(2) Um später Dataildaten, die nur für bestimmte Komponententypen interessant sind, adäquat zuweisen zu können, wird der Typ der Komponente bestimmt. Anhand dieser Zuweisung (gültige Zuweisungen sind in der später aufgeführten Tabelle „Komponententypen“ aufgeführt) entscheidet sich z.B., ob typische Router- oder PC-Informationen für diese Komponente erfaßt werden. Diese Daten fallen aber schon in den Bereich des erweiterten Datenbestandes.

(3) Hier wird der Name eingetragen, den die Komponente vom Hersteller bekommen hat. Bei managebaren Komponenten ist die idealerweise aus der `mib2` der Eintrag `sysDescr` zu übernehmen, um schon hier Übereinstimmungen zu schaffen.

Eigene Einträge sollten der Definition von `sysDescr` folgen, wobei zu beachten ist, daß die Software erst im erweiterten Datenbestand mit aufgenommen wird.

> „A textual description of the entity. This value should include the full name and version identification of the system's hardware type, software operating-system, and networking software.“ [RFC1213]

Für die Einträge müssen individuell Konventionen getroffen werden, nach denen man diese Einträge aufbaut.

(4) Hier werden die Adressen eingetragen, auf der die Komponente auf den verschiedenen Transportebenen angesprochen wird. Im IP-Bereich sind dies typischerweise die IP-Adresse, der Hostname und die MAC-Adresse. Bei Komponenten mit nur einem Port, normalerweise Endgeräte, können diese Daten aus der `mib2` übernommen werden:

`sysName`	Laut `mib2` der „fully-qualified domain name“
`ifPhysAddress`	MAC-Adresse des entsprechenden Interfaces

(5) Über diesen Eintrag wird die Identifizierung über die Inventarnummer festgelegt.

(6) Der Betreuer einer Komponente kann hier entweder als Referenz auf einen Personeneintrag, oder analog der Variable `sysContact` aus der `mib2` erfolgen:

> „The textual identification of the contact person for this managed node, together with information on how to contact this person.“[RFC1213]

Dabei soll im Grunddatenbestand bei Betreuerhierarchien nur der erste Ansprechpartner enthalten sein. Die weiteren Ansprechpartner sind dann im erweiterten Datenbestand einzuordnen.

So soll hier z.B. nur der Vor-Ort-Betreuer, nicht aber der Systemspezialist eingetragen sein.

(7) Dies ist nur dann sinnvoll auszufüllen, wenn es sich nicht um eine Netzkomponente sondern ein Endgerät handelt und dieses für eine bestimmte Person oder genau abgegrenzte Personengruppe dediziert ist.

(8) In diesem Feld soll der Standort derart beschrieben werden, das er eindeutig für den Betrieb ist. Dieser Eintrag ist auch in der `mib2` unter `sysLocation` zu finden.

Es ist zu beachten, daß die Daten der `mib2` auch in diese erst (zum Teil) eingetragen werden müssen. Hier ist im Vorfeld die Entscheidung zu treffen, welchem Ort der Datenhaltung die größere Priorität zugewiesen wird, und, wenn die Entscheidung zugunsten des Dokumentationssystem ausfällt, die Daten in den Komponenten-MIBs überhaupt gepflegt werden.

Von der Konzeption her ist es eleganter, Daten in den Komponenten als aktueller zu betrachten und sie dort zu pflegen, da viele Managementwerkzeuge nur auf die Komponenten selbst zugreifen können und so von Anfang an Inkonsistenzen vermieden werden können.

Durch ein regelmäßiges Update können die Daten des Dokumentationssystems mit denen der Komponenten abgeglichen werden. (vgl. hierzu auch die Beschreibungen in der BMW-Umgebung im Abschnitt 6.3.4)

Nachdem die Daten für die einzelnen Komponenten motiviert sind, muß jetzt aus diesen das Netz in seiner physischen Struktur modelliert werden. Dazu sind die bis jetzt isoliert betrachteten Komponenten anhand ihrere physischen Verbindungen untereinander zu verknüpfen. Um nicht nur die Verbindungen nachzubilden, werden auch Parameter, wie Übertragungsgeschwindigkeit und Anschlußart, mit erfaßt.

Die Erläuterungen zu der folgenden Tabelle befinden sich wieder im Anschluß.

Grunddatenbestand: Gruppierung von Komponenten

- (A) In welchem physischem Verbund sind die Komponenten zu sehen?
  - Mit wem ist die Komponente wie verbunden?
    - (B) Welche Anschlußarten/Ports/Geschwindigkeit stellt die Komponente bereit?
    - (C) Was für eine Art von Verbindung besteht?
      - Point-to-Point
      - Bus
    - (D) Über welchen Port sind andere Komponenten an diese Komponente angeschlossen ?
  - (E) Wo liegt die Verbindung?

(A) Es sollen hier vorerst nur physische Verbindungen aufgenommen werden, also die real existierenden, direkten Verbindungen zwischen zwei Komponenten.

Die Dokumentation logischer Verbindungen, wie Subnetzen und Domänen, erfolgt erst später. Diese können dann auf der physischen Struktur aufbauen, oder vollkommen unabhängig voneinander sein (Siehe „virtual Workgroups" bei ATM, wo nebeneinander stehende, miteinander verbundene Komponenten unterschiedlichen Subnetzen angehören können).

(B) Hier werden alle belegbaren Ports eingetragen, die die Komponente zur Verfügung stellt. Bei einer managebaren Komponente sollten diese Einträge in der Zählweise aufgeführt werden, wie sie auch in der Liste `IfEntry` der `mib2` eingetragen sind. Durch die Attributierung mit den Einträgen aus der `mib2` [RFC1213] ist der Port in seinen Eigenschaften hinreichend beschrieben:

`ifDescr`	A textual string containing information about the interface. This string should include the name of the manufacturer, the product name and the version of the hardware interface.
`ifType`	The type of interface, distinguished according to the physical/link protocol(s) immediately „below" the network layer in the protocol stack.
`ifSpeed`	An estimate of the interface's current bandwidth in bits per second. For interfaces which do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth.
`ifPhysAddress`	The interface's address at the protocol layer immediately „below" the network layer in the protocol stack. For interfaces which do not have such an address (e.g., a serial line), this object should contain an octet string of zero length.

(C) Mit der Information, um welchen Verbindungstyp es sich handelt, vereinfacht sich die Generierung einer graphischen Darstellung. Als Verbindungstypen gibt es nur „Bus" und „Point-to-Point", da dies die beiden einzigen Strukturen sind, die auf physischer Ebene bestehen. Eine Ringstruktur besteht aus einer geschlossenen Kette von Point-to-Point-Verbindungen.

(D) Über diese Frage bekommt man die Liste der angeschlossenen Komponenten. Diese enthält dann auch die korrespondierenden Ports, über die die Komponenten untereinander verbunden sind.

(E) Um bei Leitungsstörungen zu wissen, wo die Kabel verlegt sind, muß der Verlauf dokumentiert sein. Solange nur einzelne Kabel die Verbindung bilden, ist diese Dokumentation nicht zwingend, da die Übersicht vor Ort noch erhalten bleibt. Sind jedoch Kabel*stränge* verlegt, die sich womöglich auf einer Strecke vermischen und in anderen Kombinationen wieder teilen, so ist eine Dokumentation mit genauer Typbezeichnung und evtl. sogar eigenem Bezeichner unablässig.

3.3 Erweiterter Datenbestand

Im erweiterten Datenbestand werden zusätzlich die Daten mit aufgenommen, die aufgabenübergreifend von mehreren Verfahren benötigt werden. Hierzu werden weitere Datenbereiche erfaßt und die bestehenden weiter vertieft. Durch die Detaillierung können diese für weitere Einsatzbereiche an Bedeutung gewinnen.

Ein wesentlicher Punkt stellt jetzt die Bildung logischer Strukturen auf das physisch vorhandene Netz dar. Im erweiterten Datenbestand sind die verschiedenen Domänenarten dokumentiert, wie Subnetzbildung, Abrechnungsgebiete, Abteilungszugehörigkeiten, usw.

Die Verwaltung installierter Software stellt einen Teil des Systemmanagements dar. In folgenden Bereichen werden Informationen über Software benötigt, die die Aufnahme in den erweiterten Datenbestand rechtfertigen:

- Softwareverteilung: Dokumentation der erforderlichen Rahmenbedingungen
- Umzugsplanung: Bereitstellung derselben Umgebung am neuen Arbeitsplatz
- Konfiguration: Dokumentation der Installationen
- Fehlermanagement: Liefert mögliche Fehlerursachen, z.B. nicht aufeinander aufbauende Softwarepakete

Die folgenden Tabellen zeigen die Methodik zur Ermittlung der Daten für den erweiterten Datenbestand im Bereich der Komponente und der Software.

Erweiterter Datenbestand: Verfeinerung auf Komponentenebene
Für Komponententypen:
(a) Welche spezifischen Informationen sind für diesen Typ von Komponente von Bedeutung?
Für jede einzelne Komponente:
(b) In welchen Domänen befindet sich die Komponente?

(a) Wie bereits im Grunddatenbestand für die Einzelkomponenten angeführt, werden den einzelnen Komponententypen individuelle Attribute zugeordnet. Diese werden nämlich für bestimmte Verfahren benötigt. Diese sind im Konfigurationsbereich angesiedelt und können zur Fehlerbehebung mit herangezogen werden.

Beispielsweise werden u.a. CPU-Typ und RAM eines PCs benötigt, um zu entscheiden, ob ein bestimmtes Softwarepaket installiert werden kann. Im Fehlermanagement kann die ungenügende Ausstattung mit RAM einen Hinweis auf die Ursache für Fehlverhalten geben.

Die folgenden Komponententypen stellen eine Auswahl dar und müssen an die konkrete Betreiberumgebung angepaßt sein. Gemäß der Umgebung sind dann auch die Detaildaten zu modifizieren.

Komponententypen:

Komponententyp	Beispielhafte Detaildaten
Terminal:	Emulation, ...
X–Terminal:	RAM, SW/Color/Greyscale, ...
Workstation/PC:	RAM, CPU, OS, ...
Host:	RAM, CPU, OS, ...
Router:	Routing-Tabellen, ...
Bridge:	Filtereinstellungen, ...
Repeater/Sternkoppler:	keine
Verteilerschrank:	keine
Drucker	Sprache(n), RAM

(b) Jede Komponente ist Mitglied in einer oder mehreren Domänen. Diese können nach verschiedenen Aspekten definiert sein, beispielsweise Konfiguration (Subnetze, usw.), Abrechnung, Sicherheit.

Erweiterter Datenbestand: Software	
Für jedes Software-Paket:	
α	Wie heißt die Software?
β	In welcher Version liegt die Software vor?
γ	Welche minimalen Systemvoraussetzungen werden gefordert?
δ	Wieviele Lizenzen bestehen und wie lange sind diese gültig?
ϵ	Auf welchen Systemen ist die Software installiert?

α Hier wird die Produktbezeichnung und der Hersteller erfaßt.

β Da Software in verschiedenen Versionen vorliegen kann, die sich im Funktionsumfang deutlich unterscheiden, muß für jedes Paket auch die Version mitgeführt werden.

γ Hier werden die Rahmenbedingungen erfaßt, die zum korrekten Installation benötigt werden. Zu nennen sind: Benötigter Plattenplatz, minimale Speicheranforderungen, Betriebssystemversion, evtl. andere erforderliche Software.

δ Dieser Eintrag ist nur bei Software mit begrenzter Lizenzdauer oder limitierter Anzahl von Benutzern bzw. Installationen sinnvoll.

ϵ Diese Information liefert einen Überblick über die Installationen. Dies ist z.B. wichtig bei Versionsänderungen, um auch alle Instanzen zu erfassen und eine gleichförmige Softwarebasis im Unternehmen sicherzustellen.

3.4 Nice-To-Have Datenbestand

In diese Klasse werden die Informationen aufgenommen, die speziell für ein einzelnes Verfahren benötigt werden und somit sehr spezifisch sind.

Spezifisch auch in der Hinsicht, daß jetzt basierend auf dem minimalen und erweiterten Datenbestand spezielle Aspekte einzelner Aufgabenfelder weiter detailliert und unabhängig von anderen Feldern aufgebaut werden können. Aufgrund des Modells sind in den beiden anderen Datenbeständen dazu keine Änderungen vorzunehmen.

Da diese Erweiterungen jetzt sehr spezifisch und vielfältiger Natur sind, werden im folgenden nur beispielhafte Einzelaspekte behandelt.

- Erweiterungen im Bereich des Konfigurationsmanagement:
  Erweiterung um spezifische Systemkonfigurationsdaten, wie Serveranbindungen (z.B. Software-Server, DNS, Drucker-Server) oder von der Standardkonfiguration abweichende Software (z.B. spezielle Treibersoftware). Dabei kann dies der Name des Software-Pakets und abweichende Parameter, oder, wenn auch aufwendiger, ein „download"-fähiges Backup sein.

  Die Softwarebeschreibungen können um genauere Vertragsdaten erweitert werden, die z.B. genauerer die Servicepflichten des Hersteller bzw. der Firma, mit der ein Servicevertrag abgeschlossen wurde, beschreibt.

- Erweiterungen im Umfeld des Fehlermanagements:
  Um die Zeit der Fehlerbehebung zu verkürzen, können, um die Weiterleitung von Trouble-Tickets an nicht anwesende Mitarbeiter zu vermeiden, deren An- bzw Abwesenheit mit vermerkt werden.

  Sind die Lokalitäten des Betreibers groß, ist es auch von Vorteil für den Service von Netzkomponente nicht nur zu wissen, wo die Komponente steht, sondern auch wer für diesen Raum schlüsselberechtigt ist, oder auch wer der Hausmeister ist.

Weitere Beispiele lassen sich nach belieben hinzufügen.

Kapitel 4

Exemplarische Anwendung

Der folgende Abschnitt umfaßt drei Teile. Im ersten wird exemplarisch an einer bestehenden (Minimal-)Umgebung die Anwendbarkeit der Methodik zur Gewinnung des Grunddatenbestandes überprüft, dann exemplarisch ein minimales E-R-Modell vorgestellt und im dritten Teil dann die Daten anhand des Fehlermeldeverfahrens benutzt, um die Anwendbarkeit des Datenbestandes zu testen.

4.1 Aufbau eines Grunddatenbestandes

Anhand der in Abschnitt 3.2 beschriebenen Fragen, wird im folgenden exemplarisch ein Grunddatenbestand aufgestellt, da dieser das Fundament für die Erweiterungen ist. Gewählt wurde eine einfache, bestehende Umgebung, an der überprüft wurde, ob das Vorgehen folgende Eigenschaften erfüllt:

- Die Methodik ist anwendbar.
- Die Daten sind verfügbar.

Es wurden die Daten für vier Komponenten, zwei PCs, ein Hub und eine Workstation ermittelt. Dabei wurde streng nach erarbeiteten Fragen vorgegangen. Dabei ist zu den einzelnen Daten zu bemerken:

1. Bei der Typbezeichnung konnte nur bei dem Nixdorf-PC eine genaue Bezeichnung am Gerät abgelesen werden. Bei dem zweiten PC fehlt die Typbezeichnung, daher wurde dieser als „No Name“ eingeordnet. Bei der Workstation und dem Hub wurden diese Daten mit einem MIB-Browser aus der `mib2, sysDescr` ermittelt.
2. Der Hersteller war bis auf den „No Name“-PC am Gerät abzulesen.

3. Die Benennung im Netz war nur mit Rechnerhilfe zu lösen. Bei bekanntem IP-Hostname[1] wurde über den `host`-Befehl die IP-Adresse ermittelt. Die Ethernetadressen wurden entweder wieder über einen MIB-Browser abgefragt oder wurden nicht ermittelt.[2]

4. Die Inventarnummer konnte nur bei zwei Geräten festgestellt werden, da der Inventaraufkleber bei den anderen fehlte.

5. Als Betreuer wurden die Einträge der `mib2, sysContact` herangezogen. Für die PCs wurde als Betreuer der Leiter des Rechnernetzepraktikums Hr. Dr. Abeck eingesetzt.

6. Der Benutzereintrag ist nur der Vollständigkeit halber angegeben. Nützlich wäre dieser nur, wenn mit dieser Gruppe eine Referenz auf die Praktikanten verbunden wäre. Da diese aber zweimal pro Jahr wecheln, werden diese nicht erfaßt.

   Diese Einträge sind aber nicht Bestandteil des Grunddatenbestandes.

7. Der Standort wurde vom Türschild abgelesen.

Suche alle Endgeräte und Netzkomponenten: Folgende Geräte befinden sich im Praktikumsraum G525 am Institutsnetz (ohne Novell-Netz):

Typ:	PC, Modell 8810 M 55
Hersteller:	Nixdorf
Eindeutiger Name im Netz:	Hostname: "mars.informatik.tu-muenchen.de", IP: 131.159.12.11, Ethernet: XX XX XX XX XX XX
Eindeutiger Name im Betrieb:	08103
Betreuer dieser Komponente:	Dr. Abeck, 2105-2383
Benutzer dieser Komponente:	Praktikanten des Rechnernetzepraktikum
Standort dieser Komponente:	Barer Str. 40, G 525

[1] Dem Autor sind diese bekannt durch regelmäßigen Gebrauch

[2] Der Rechner athegering10 hat zwei Ethernetinterfaces, da dieser Rechner als Bridge eingesetzt wird.

Typ:	PC, No Name
Hersteller:	No Name
Eindeutiger Name im Netz:	Hostname: "athegering10.informatik.tu-muenchen.de", IP: 131.159.12.20, Ethernet XX XX XX XX XX XX, Ethernet XX XX XX XX XX XX
Eindeutiger Name im Betrieb:	08651
Betreuer dieser Komponente:	Dr. Abeck, 2105-2383
Benutzer dieser Komponente:	Praktikanten des Rechnernetzepraktikum
Standort dieser Komponente:	Barer Str. 40, G 525

Typ:	Sternkoppler, HP2868A Fiber-Optic Hub Plus, HW A.01.00
Hersteller:	Hewlett Packart
Eindeutiger Name im Netz:	Hostname: "hubhegering1.informatik.tu-muenchen.de", IP: 131.159.12.44, Ethernet: 08 00 09 32 25 03
Eindeutiger Name im Betrieb:	none
Betreuer dieser Komponente:	Dr. Abeck, 2105-2383
Benutzer dieser Komponente:	Praktikanten des Rechnernetzepraktikum
Standort dieser Komponente:	Barer Str. 40, G 525

Typ:	Workstation, 9000/700 2009709682
Hersteller:	Hewlett Packart
Eindeutiger Name im Netz:	Hostname: "hpheger3.informatik.tu-muenchen.de", IP: 131.159.12.41, Ethernet: 08 00 09 27 82 49
Eindeutiger Name im Betrieb:	none
Betreuer dieser Komponente:	Stefan Schwertner, 2105-8116
Benutzer dieser Komponente:	Praktikanten des Rechnernetzepraktikum
Standort dieser Komponente:	Barer Str. 40, G 525

Komponente 1	Port	Ü/u	→ Komponente 2	Port	Ü/u	über Verbindungsart	Verbindungstyp	Verbindungsverlauf
mars	Port 1	u	→ athegering10	Port 1	Ü	über LWL 10BaseF	Point-to-Point	G525
athegering10	Port 1	Ü	→ mars	Port 1	u	über LWL 10BaseF	Point-to-Point	G525
athegering10	Port 2	u	→ hubhegering1	Port 2	Ü	über LWL 10BaseF	Point-to-Point	G525
hubhegering1	Port 2	Ü	→ athegering10	Port 2	u	über LWL 10BaseF	Point-to-Point	G525
hubhegering1	Port 3	Ü	→ hpheger3	Port 1	u	über LWL 10BaseF	Point-to-Point	G525
hubhegering1	TRANS	u	→ Transceiver	Port 1	Ü	über AUI	Point-to-Point	G525 → Vorraum
hpheger3	Port 1	u	→ hubhegering1	Port 3	Ü	über LWL 10BaseF	Point-to-Point	G525
Transceiver	Port 1	Ü	→ hubhegering1	TRANS	u	über AUI	Point-to-Point	Vorraum → G525
Transceiver	Bus	u	→ BUS1		Ü	über 10Base5	Bus	Vorraum → ?

Verbindungen: (Beschreibung im Anschluß)

In dieser Tabelle wird der Verlauf der Verbindungen dokumentiert. Ausgehend von der ersten Komponente wird der Port identifiziert. Dieser wird dann als 'Ü'bergeordnet oder 'u'ntergeordnet eingestuft. Der Port der zweiten Komponente erhält die gegensätzliche Einstufung. Über diese Einstufung kann mit wenig Aufwand eine Wegfindung zwischen zwei Stationen implementiert werden, indem man von beiden Stationen die Hierarchie aufwärts verfolgt. Dies geht solange, bis auf beiden Wegen eine identische Station liegt, oder der Backbonebereich erreicht ist. In diesem Bereich muß dann mit anderen Techniken gearbeitet werden, wie zum Beispiel statischen Tabellen. Zusätzlich ist noch die Verbindungsart (hier: 10BaseF, AUI und 10Base5), der Verbindungstyp (hier Point-to-Point und Bus) und der Verbindungsverlauf erfaßt.

Bei dieser Erfassung traten keine Probleme auf, da die Daten aus der Installation und anhand der Beschriftung der Geräte eindeutig waren.

In Abbildung 4.1 ist die Installation noch einmal graphisch verdeutlicht.

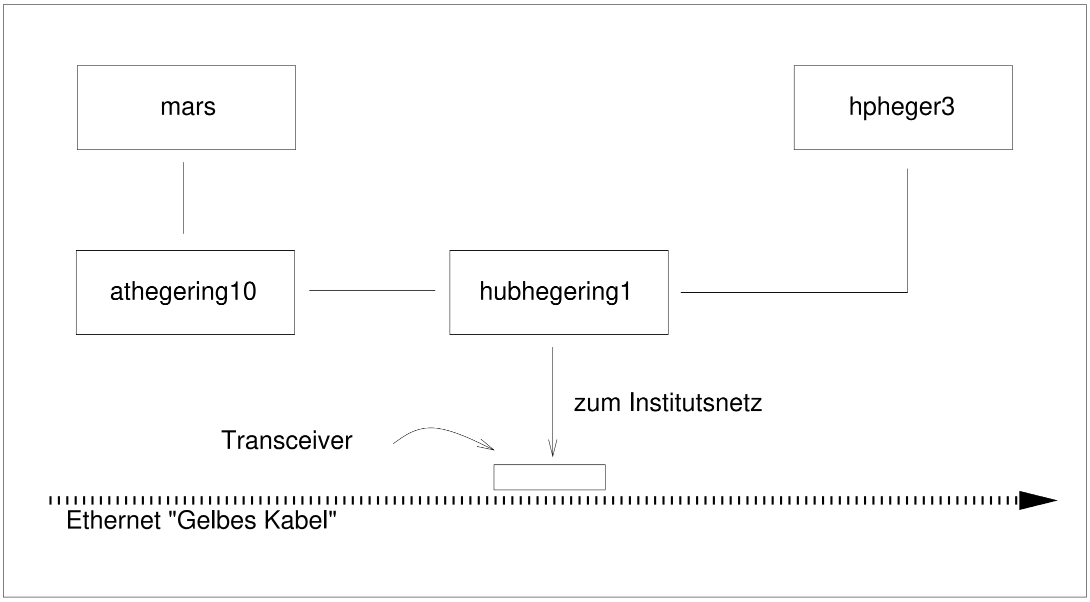

Abbildung 4.1: Im Beispiel erfaßte Komponenten

Zu Beachten ist, daß in dieser Umgebung keine Verteilerschänke als „passive" Knotenpunkte vorhanden sind.

4.2 Beispiel: E-R-Modell des Grunddatenbestandes

Um einen Überblick über die Grunddaten und eine mögliche Strukturierung in einem E-R-Modell vorzustellen, werden die eben gewonnen Daten einem E-R-Modell zugeordnet. Für eine Erklärung von E-R-Modellen sei auf [Kand93] und [Schla93] verwiesen.

Das zentrale Element ist die Entity „Komponente". Mit dieser sind alle anderen Entities verknüpft (Abbildung 4.2). Über die „benutzt" „administriert" Relationen findet die Verknüpfung zu der Entity „Person" statt. Mehrere Anwender können also die Komponente benutzten. Der Systembetreuer ist – nach Definition des Grunddatenbestandes – die erste Kontaktperson in der Betreuerhierarchie und somit eindeutig.

Über die „hat" Relationen werden die anderen Entities zugeordnet. Die „hat" Relation zwischen der „Komponente" und „Verbindung" ist zwar redundant, da auch der Weg über den „Port" gewählt werden kann, ist aber bei der Wegverfolgung von Nutzen, da der Weg direkt von einer Komponente zur nächsten Komponente führt. Im folgenden werden noch die Attribute der einzelen Entities im Überblick dargestellt:

Entity	Attribut
Komponente	Typ
	Hersteller
	Hostname
	IP-Adresse
	Inventarnummer
Person	Name
	Telephonnummer
Standort	Raumnummer
	Straße, Hausnummer
Verbindung	Verbindungsart
	Verbindungstyp
	Verbindungsverlauf
	ist-übergeordnet/ ist-untergeordnet
Port	Portnummer
	Ethernet-Adresse
	IP-Adresse

Dieses Modell ist nach den Daten des vorherigen Abschnitts gestaltet. Mit Ausbau um dem „erweiterten Datenbestand" müssen nur die Entity „Software" und einige weitere Attribute eingeführt werden. Das Grundmodell bleibt also nach der Vorgabe des Konzeptes unverändert und wird nur erweitert.

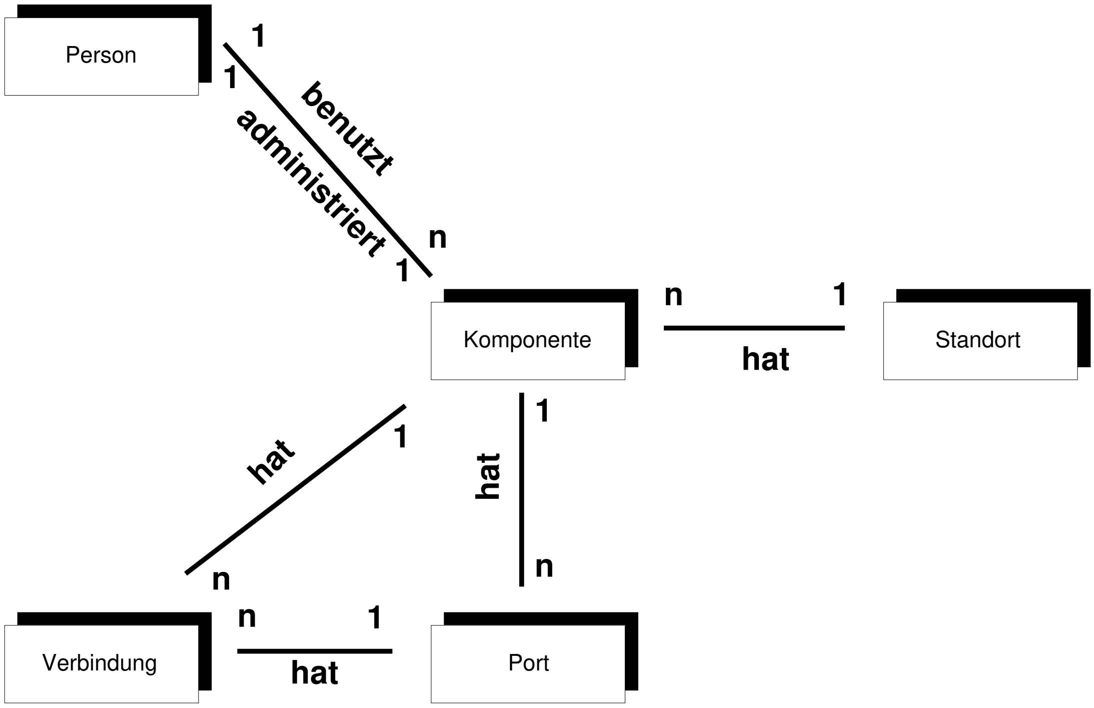

Abbildung 4.2: Beispiel: Grunddatenbestand als E-R-Modell

4.3 Anwendung des Grunddatenbestandes am Beispiel des Fehlermeldeverfahrens

In diesem Abschnitt wird anhand des Fehlermeldeverfahrens die Anwendbarkeit des beispielhaft angeführten Grunddatenbestands des letzten Abschnitts nachgewiesen.

Das Fehlermeldeverfahren ist im Bereich des Fehlermanagements angeordnet und beschreibt den Verlauf eines im Netz aufgetauchten Fehler bis zur seiner Behebung und Dokumentation. Ziel ist es genügend Informationen zu sammeln, um den Fehler einzukreisen und Strategien zum Lösen der Problematik zu finden.

Diese Beispiel wurde gewählt, da es einen typischer Ablauf in der Netzmanagementumgebung darstellt.

Im folgenden werden die verschiedenen Stufen des in Abbildung 4.3 schematisch skizzierten Fehlermeldeverfahrens erläutert, um anhand des Ablaufes die aus dem exemplarischen Datenbestand benötigten Informationen zu beschreiben. Zusätzlich ist in der Graphik die Zunahme an Information über den Fehler im zeitlichen Verlauf dargestellt.

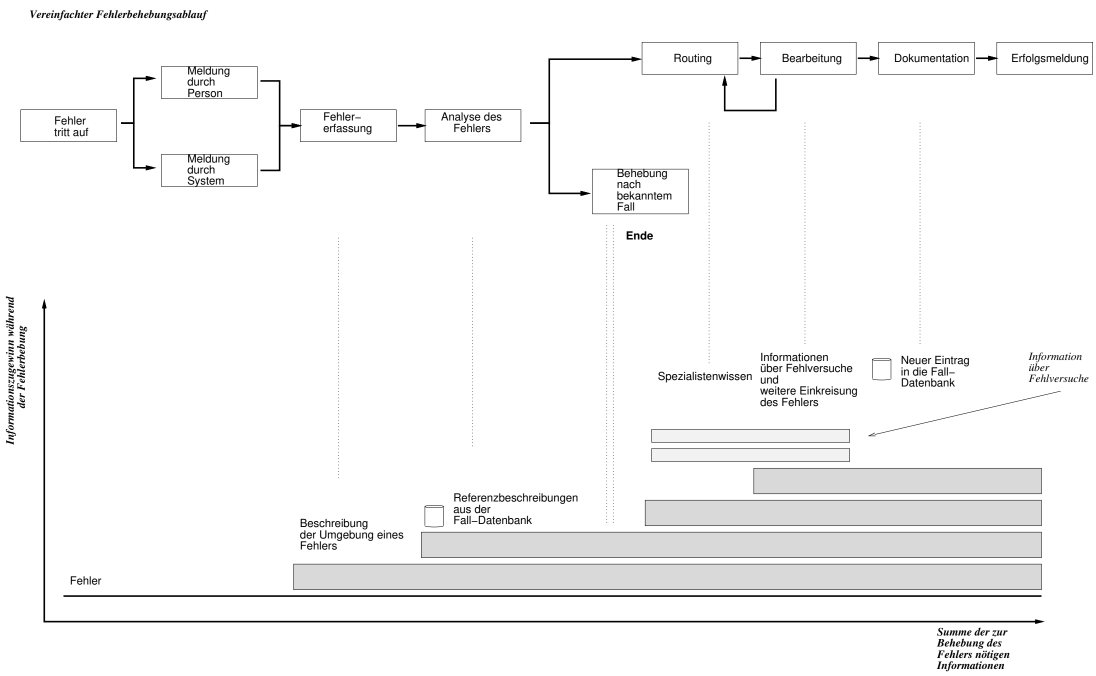

Abbildung 4.3: Schematischer Ablauf des Fehlermeldeverfahrens und zeitlicher Informationszuwachs

1. Fehler tritt auf:
   Ein Fehler stellt eine Störung im System oder ein anormales Verhalten dar. Dabei stellt der Fehler als solches zunächst eine äußerst informationslose Struktur dar.

   Zu diesem Zeitpunkt ist der Fehler noch nicht bekannt, d.h. es kann auch nicht reagiert werden.

2. Meldung durch Person/System:
   Man weiß zwar jetzt, daß der Fehler existiert, hat aber keine noch ausreichenden Informationen darüber, wie er zu lokalisieren und zu beheben ist.

   Ist der Melder ein Anwender, so kann jetzt schon automatisch aus dem Dokumentationssystem eine Referenz zu den Personendaten gezogen werden, um im weiteren Verlauf z.B. Rückfragen zu stellen oder ihn über die Fehlerbehebung zu informieren.

3. Fehlererfassung:
   Hier werden die Auswirkungen des Fehlers erfragt und in einem Trouble-Ticket dokumentiert.

   Dabei ist die eigentliche Ursache (z.B. ein loser Transceiver) meist zunächst nicht bekannt, sondern nur die Auswirkungen (Klage des Anwenders: „Bei mir geht nichts mehr ...“) und das Umfeld in dem der Fehler wirksam geworden ist (Arbeitumgebung ist ein PC, eine Workstation, usw).

4. Analyse des Fehlers:
   In diesem Schritt werden zusätzliche Daten gesammelt, die zur Behebung des Fehlers beitragen können. Hierzu werden die Daten aus dem Dokumentationssystem benötigt:

   - Konfigurationsdaten sammeln und auf deren Relevanz überprüfen. Diese Daten sind zwar im Beispiel nicht enthalten, werden aber im „erweiterten Datenbestand“ mit erfaßt.
   - Wegbestimmung und -verfolgung: Im Dokumentationssystem kann der physikalische Weg verfolgt werden und dann im Netzmanagmentsystem nachgeprüft werden, ob sich dieser Umgebung der Fehler auch mit ausgewirkt hat. Eventuell läßt sich hier mit diesen Informationen schon der Fehler beheben.

     Die Wegbestimmung läßt sich schon mit dem Grunddatenbestand realisieren.

   In der Falldatenbank kann nach ähnlichen Fehlersituationen gesucht werden, die zur Behebung beitragen können.

5. Behebung nach bekanntem Fall:
   Mit ähnlichen Beschreibungen aus einer Falldatenbank kann in diesem Schritt der Fehler behoben werden. Das Verfahren ist in diesem Teilast abgeschlossen.

6. Routing:
   Hier wird der nächste Bearbeiter für das Trouble Ticket gewählt. Ist dieser der Vor-Ort-Betreuer, kann er schon aus dem Grunddatenbestand über die gestörte Komponente ausgewählt werden.

7. Behebung:
   In diesem Schritt wird versucht aus den gesammelten Daten die Fehlerursache auszugrenzen. Dazu müssen Seiteneffekte beachtet werden und falls nötig weitere Daten gesammelt werden. Diese können sich evtl. durch die Wegverfolgung oder auch durch die Darstellung von Konfigurationsdaten ergeben.

Falls der Bearbeiter nicht zu einem Lösungsweg findet, kann ein Re-Routing stattfinden.

8. Dokumentation:
   Es ist wichtig, um auf das Wissen der Behebung von Fehlern auch später noch zugreifen zu können, die Beschreibung der Fehler so in das System einzubetten, daß anhand möglichst weniger Charakteristika eine Referenzbeschreibung identifiziert werden kann. Denn um zu gewährleisten, daß bei einem wiederholten Auftritt nicht wieder mit dem Wissensstand Null angefangen wird, muß die Fehlerbeschreibung und die Lösungsstrategie so in das Dokumentationssystem als Falldatenbank integriert werden, daß ein erneut auftretender ähnlicher Fall anhand von Charakteristika (gleiche Komponente, gleiche Gruppe, Kombination von Endgerät und Anwender, usw.) schnell als schon einmal gelöst gefunden werden kann.

   So wird es für die Fehlerbehandlung wichtig sein, im „nice-to-have" Datenbestand Referenzen aus der Falldatenbank zu den Komponenten in der Dokumentation zu ziehen, anhand derer für eine bestimmte Komponente oder Komponentengrupppe eine Vorauswahl spezieller Fälle getroffen werden kann.

9. Erfolgsmeldung:
   Der Melder bekommt die Information, daß der Fehler behoben wurde.

Dieser Ablauf zeigt, daß der Grunddatenbestand die Basisinformationen für die Einleitung weiterer Schritte bei der Fehlerbehebung liefert. Diese sind:

- Lokalisation von Komponenten
- Wegeverfolgung: Darstellung der Umgebung der Komponente
- Routing-Informationen an Vor-Ort-Betreuer

Nicht liefern kann der Grunddatenbestand genauere Konfigurationsdaten und Informationen über die auf der Komponente laufenden Software. Diese sind jedoch im „erweiterten Datenbestand" vorhanden.

Dies zeigt, daß der Grunddatenbestand in seiner Konzeption die nötigen Basisinformationen bietet, im erweiterten Stadium sogar ein tragfähiges System darstellt.

Die erarbeitete Methodik läßt sich gut in ein neu zu erstellendes Dokumentationssystem mit einbringen. In schon bestehenden Umgebungen ist die konsequente Anwendung mit einer Reorganisation bestehender Systeme verbunden, die jedoch sehr aufwendig ist.

Doch auch ohne Reorganisation ist das Ergebnis der Anwendung der Methodik nützlich. Es können so Defizite im bestehenden Datenbestand aufgedeckt werden, wie z.B. fehlende Korrelationen zwischen den Einzeldaten.

Typischer Weise sind die Daten nicht in einem einzigen Datenbestand enthalten, sondern anwendungsorientiert im Betrieb verteilt. Typische Datenbestände sind: Personaldatenbanken, Buchhaltungssysteme, Inventardatenbänke und in der Netzmanagementumgebung Netzdokumentationssysteme, Netzmanagementsysteme und Element-Managementsysteme.

Anhand der Beispiele des Netzmanagementsystems und der Netzdokumentation wird nun im nächsten Teil gezeigt, wie durch die Anbindung dieser beiden Systeme eine Steigerung der Effektivität sowohl in der Anwendung als auch in der Pflege erreicht werden kann.

Teil II

Netzdokumentation und Netzmanagementsystem

Kapitel 5

Differenzierung zwischen Netzdokumentation und Netzmanagementsystem

Die Verteiltheit der managementrelevanten Informationen stellt ein großes Hemmnis bei der Verwirklichung des integierten Management dar. Die Informationen sind auf verschiendene, oft sogar flache, Datenbanken verteilt und werden meist mittels proprietären Werkzeugen aufgebaut und benutzt. Eine Betrachtung der Informationen aus einer einheitlichen Gesamtsicht ist daher nicht möglich. Unter dem Druck der Anwender sind jetzt jedoch die Hersteller gezwungen, nach außen dokumentierte Schnittstellen bereitzustellen, über die eine, wenn auch teilweise rudimentäre, Kommunikation zwischen den Werkzeugen ermöglicht werden kann.

Doch um eine Anbindung zu realisieren, müssen zuerst einmal die Eigenschaften der Systeme analysiert werden, unter welchen Aspekten sie sich gegeneinander abgrenzen und wo sie sich überschneiden. Dies kann sowohl auf Datenebene als auch auf funktionaler Ebene betrachtet werden.

Dies wird im folgenden anhand der Beispiels des Netzdokumentations- und des Netzmanagementsystems verfolgt, da diese Systeme zentrale Werkzeuge in der Netzmanagementumgebung sind. Dann werden Integrationstechniken vorgestellt, die eine unterschiedliche „Tiefe“ der Anbindung, bis hin zur Integration, beschreiben. Abschließend werden dann an einer konkreten Betreiberumgebung Szenarien vorgestellt, die eine Anbindung erfordern.

NDS:	Netzdomentationssystem
NMS:	Netzmanagementsystem

5.1 Netzmanagement- und Netzdokumentationssysteme

Um eine klare Differenzierung zwischen den beiden Systemen zu treffen, werden diese im folgenden kurz vorgestellt und deren Aufgaben klassifiziert.

5.1.1 Netzmanagementsystem

Verallgemeinert beschäftigen sich Netzmanagementsysteme mit dem Sammeln und Auswerten von Daten aus dem Netz unter verschiedenen Aspekten. Die heutigen Systeme werden in folgenden Aufgabengebieten eingesetzt:

- Konfigurationsmanagement:
  Die wesentliche Grundlage eines Netzmanagementsystems ist die Übersicht über (managebare) Komponenten im Netz und ihren logischen Verbünden. Diese werden in sogenannten „Maps“ aufbereitet dargestellt und erlauben eine übersichtliche, meist nach Subnetzen strukturierte Darstellung. Auch kann über das Netz auf die Komponenten manipulierend zugegriffen werden, so daß Konfigurationsänderungen aus der Ferne vorgenommen werden kann. Eine wesentliche Funktionalität ist das „Autodiscovery“, die kontinuierlich nach neuen Komponenten im Netz sucht und diese dann in die bestehende „Maps“ einordnet.

- Fehlermanagement:
  In diesem Bereich wird das Netz dahingehend überwacht, daß Fehlermeldungen, wie sie von den Komponenten in Form von „Events“ gemeldet oder vom Managementwerkzeug selbst aktiv abgefragt werden, gesammelt und angezeigt werden. Zusätzlich gibt es meist Mechanismen, die es erlauben, Folgefehler, die auf einer einzigen Störung bzw. Fehlverhalten beruhen, herauszufiltern, so daß nur der auslösende Fehler angezeigt wird. Dieser kann dann, solange die fehlerhafte Komponente noch erreichbar ist und der Fehler aufgrund einer Mißkonfiguration beruht, mittels des Netzmanagementsystems behoben werden.

- Performancemangement:
  Die Überwachung der Auslastung des Netzes und das frühzeitige Anzeigen von Trends, die zu einer Überlastung des Netzes führen können, fallen auch in den Bereich des Netzmanagements. Kurzfristig werden dafür Werte aus den Netzkomponenten überwacht und bei Über- bzw. Unterschreitung Alarmmeldungen ausgegeben. Langfristig werden Daten gesammelt, die das Durchsatzverhalten dokumentieren. So werden Engpässe aufgedeckt, auf die dann entsprechend reagiert werden muß.

Netzmanagementsysteme stellen somit sehr komplexe Funktionalitäten und hoch dynamische Informationen bereit. Doch da diese Systeme nur die Daten darstellen, die sie über das Netz aus den Komponenten bekommen oder aus der Struktur der Anordung der Systeme gewinnen können, sind sie mit anderen Anforderungen überfordert. So können z.B. nicht managebare Komponenten nicht erkannt werden. Es bestehen zwar heute schon Bemühungen im Bereich des integrierten Managements, diese auch mit zu erfassen, doch sind dies heute noch klassische Aufgabenbereiche der Netzdokumentationssysteme.

5.1.2 Netzdokumentationssystem

Im Gegensatz zu Netzmanagementsystemen, die in den oben genannten Aufgabengebieten maßgerechte Lösungen anbieten, wirkt das Netzdokumentationssystem in den meisten Bereichen nur unterstützend und ergänzend mit Informationen, die andere Werkzeuge nicht zur Verfügung stellen, d.h. der Anwender wechselt zwischen den Systemen und kann durch die Kombination der Informationen sein Ergebnis ableiten.

Historisch sind Netzdokumentationssysteme entstanden aus der Notwendigkeit Informationen zu verwalten, die von den proprietären Netzmanagementwerkzeugen vernachlässigt wurden. Somit sind diese ersten Systeme aus der Bestandsführung entstanden. Der dadurch entstandene umfangreiche Datenbestand ist auf die Bedürfnisse der spezifischen Betreiberumgebung angepaßt.

Die Einsatzbereiche wurden schon in Abschnitt 3.1 anhand der Aufgaben und Verfahren in der Dokumentationsumgebung diskutiert. Doch im Gegensatz zu dort – wo eine Methodik für die Gewinnung eines netzübergreifenden Datenbestand vorgestellt wurde – sind hier existierende Netzdokumentationssysteme der Ausgangspunkt der Betrachtung. Diese dokumentieren heute hauptsächlich das Netz in seiner physischen (analog dem vorgestellten „minimalen Datenbestand“) und logischen Struktur (analog Teilen des „erweiterten Datenbestandes“).

Damit sind diese neben den Netzmanagementsysteme ein wertvoller Datenbestand auf den nicht verzichtet werden kann.

5.2 Datenbetrachtungen

Klassische Datenbereiche beider Systeme werden im folgenden dargestellt und in ein Aktualitätsschema eingeordnet. Zuerst wird eine Zuordnung der Daten zum Netzmanagementsystem getroffen. Im Anschluß werden typische Daten eines Netzdokumentationssystems beschrieben.

5.2.1 Welche Daten sind dem Netzmanagement zuzuordnen?

Die Aufgaben von Netzmanagementsystemen sind in erster Hinsicht im dynamischen Bereich der Netzadministration und Verwaltung zu suchen. Hierfür ist eine Viewbildung auf das Netz vonnöten, das dem Betreiber die Struktur sowohl in logischer als auch in physischer Hinsicht verdeutlicht. Dies wird durch die ständige Überwachung und Verarbeitung von Daten, die von Netzkomponenten bereitgestellt werden ermöglicht. Voraussetzung dafür ist, daß die Komponenten eine entsprechende Funktionalität aufweisen können. Diese liefern die Informationen normalerweise nur auf Anforderung des Managementsystems. Nur im Fehlerfall versenden diese sogenannte „Traps“, die dem Netzmanagementsystem Störungen oder Fehler unterschiedlicher Priorität mitteilen.

Typische Daten eines Netzmanagementsystemens sind somit:

- Konfigurationsdaten, wie IP-Adresse, Hostname, Subnet-Mask, Ausstattungsmerkmale, wie Interfacekarten, Hardware-Versionen
- Statusinformationen, wie Port aktiviert/deaktiviert, Komponente up/down, Lüfter an/aus, usw.
- Fehlerinformationen durch Traps, wie Übertragungsstörungen, Node up/ Node down, sowie durch Setzen von Schwellwerten, die entweder in der Komponente selber oder im Netzmanagementsystem gesetzt werden.
- Statistikinformationen über Counter, die z.B. die momentane Netzlast, Anzahl der Kollisionen usw. bereitstellen.

Zusätzlich zu diesen Daten, die rein komponentenbezogen sind, stellt ein Netzmanagementsystem zusätzlich die Funktionalität bereit, Zusammenhänge auf logischer Sicht darzustellen. Zum Beispiel automatische Erkennung von Subnetzbereichen in TCP/IP-Umgebung oder die hierarchische Struktur von SNA-Umgebungen.

Diese Daten stellen ein hochdynamisches und komplexes Gebilde dar. Die Daten unterliegen einer starkten Fluktuation, weswegen sie entweder nur auf Anforderung aktualisiert oder aber zyklisch abgefragt werden müssen.

5.2.2 Welche Daten sind der Netzdokumentation zuzuordnen?

Im Gegensatz zum Netzmanagementsystem werden in einem Netzdokumentationssystem Daten verwaltet, die eher statischer Natur sind und typischer Weise von einem Netzmanagementsystem nicht bereitgestellt werden.

Dabei ist die Trennung zwischen den Systemen historisch begründet und konzeptuell keinesfalls erwünscht, da zusammengehörige Daten künstlich getrennt wurden. Als Basis des Datenbestandes einer Netzdokumentation sind die Inventarisierungssysteme, aus denen diese auch weiterentwickelt wurden.

Neben der Inventarisierung haben sich folgende Aufgaben bzw. Informationsgruppen ergeben:

- Bereitstellung von detaillierten Verkabelungsdaten, wie Leitungsverlauf, passive Komponenten, die über ein Netzmanagementsystem nicht erfaßt werden können, sowie die Art der Verkabelung (Twisted Pair, UTP, STP, usw.)
- statische Netzinformationen, wie z.B. Informationen über die Konfiguration von Komponenten als Backup im Störungsfall,
- Verwaltung von eingesetzten und nicht eingesetzten Komponenten
- Erfassen von verdichteten Statistikdaten, wie Informationen über (längerfristige) Ausfallstatistiken.

Die Eigenschaft der Daten, daß sie meist nicht automatisch gesammelt werden können – ein Kabel, das seinen Verlauf automatisch meldet, ist dem Autor leider noch nicht bekannt – hat dazu geführt, das diese Systeme hauptsächlich manuell gepflegt werden müssen.

5.2.3 Übergänge zwischen den Systemen

Eine Trennung zwischen Netzdokumentations- und Netzmanagementsystemen läßt sich nicht klar vollziehen, da die Grenze je nach Betreiberumfeld und zeitlicher Entwicklung unterschiedlich verlaufen ist. Deswegen kann im folgenden nur tendentiell eine Klassifizierung vorgenommen werden.

Wie bereits erwähnt sind die nicht automatisch erfaßbaren Daten klar dem Netzdokumentationssystem zuzuordnen, diese sind manuell zu erfassen und zu pflegen. Der Hauptaspekt liegt für die Unterscheidung jedoch im Änderungszeitraum. Dies wird besonders durch die Graphik 5.1 verdeutlicht.

Ständig sich ändernde Daten sind der typische Einsatzbereich des Netzmanagementsystems, wie das Anzeigen des aktuellen Gerätestatus oder der Überwachung von Schwellwerten. Dazu im Gegensatz sind sich einmalig oder nie ändernde Daten, wie genaue Komponentendaten, Seriennummer, Inventarnummer der typische Bereich des Netzdokumentationssystem. Doch wie schon erwähnt existiert hier ein nicht genau abzugrenzender Überschneidungsbereich in dem beide Systeme Daten halten.

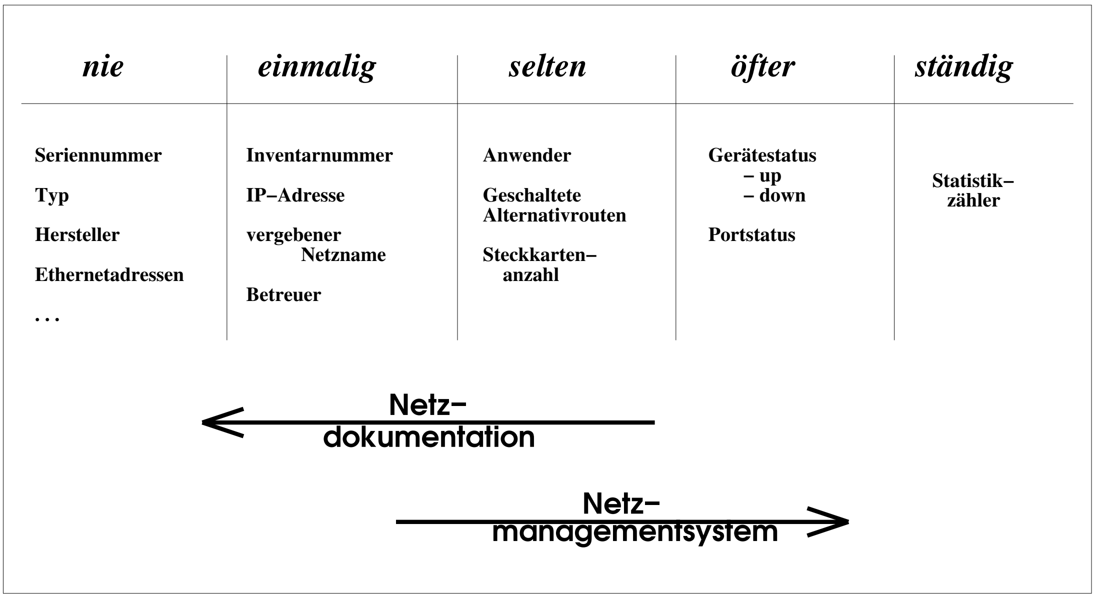

Abbildung 5.1: Abgrenzung der Systeme über den zeitlichen Aspekt

Die typischen Daten des Netzmanagement haben so einen eher einen überwachenden Charakter, die des Netzdokumetationssystem im Gegensatz dazu sind eher verwaltender Art sind.

Betrachtet man beispielsweise die Ports einer Komponete, so ist z.B. die Angabe des Status in Netzmanagement zu suchen, die genaue Typbezeichnung des Interfaces aber nur in der Netzdokumentation zu finden.

5.3 Problematik der getrennten Datenhaltung

Aus der Tatsache, daß Netzmanagementsystem und Netzdokumentation als getrennte Systeme entwickelt wurden, jedoch ihre Anwendungszwecke sich überschneiden, geht die Forderung hin zu einem integrierten Netzdokumentations- und Managemensystem.

Der Vorteil den ein Netzmanagementsystem mit seinen automatisch aktualisierten Daten bietet, soll mit den relativ statischen Daten der Netzdokumentation in Einklang gebracht werden.

Folgender Bedarf besteht für Daten einer Netzdokumentation:

- Konsistenzprüfung der enthaltenen Daten auf Korrektheit

- Automatischer Abgleich mit der sich ändernden Netzkonfiguration

Folgender Bedarf besteht für Daten eines Netzmanagementsystems:

- Integration von Informationen von nicht managebaren Komponenten
- Zugriff auf Daten, die die Informationen aus dem Netzmanagement ergänzend unterstützen

Mit diesen vier Aspekten kann schon verdeutlicht werden, daß es ein Ziel sein muß, Netzdokumentation und Netzmanagementsystem so stark aneinander zu binden, daß der Datenbestand beider logisch einen einzigen darstellt. Dies ist so zu verstehen, daß es keinen Unterschied geben sollte, ob die Daten in *einem* System in *einem* Datenbestand gehalten werden, oder sich durch die im nächsten Abschnitt 5.4 beschriebenen Integrationstechniken gegenseitig ergänzen.

Dies liegt auch im Sinne des integrierten Managements, da sich diese typischen Vorteile ergeben:

- Vermeidung von inkonsistenten und redundanten Daten:
  Durch den Abgleich der Daten untereinander können Inkonsistenzen aufgedeckt werden. Z.B. kann alleine durch die Abfrage der sich im Netz befindenden Komponenten durch das Netzmanagementsystem der Bestand in Netzdokumentationssystem überprüft werden. Zusätzlich mit anderen Maßnahmen kann so eine höhere Übereinstimmung der Abbildung der Daten in der Datenbasis mit der Realität erreicht werden.
- Geringerer Pflegeaufwand:
  Daten, die durch das Netz zur Verfügung gestellt werden, können automatisch in die Systeme übernommen werden, ohne daß sie von Hand eingegeben werden müssen. Die nur manuell einzugebenden Daten können mit Plausibilitätskontrollen, evtl. basierend auf automatisch übernommenen Daten, auf ihre Korrektheit überprüft werden.
- Bessere Anwendbarkeit beider Systeme:
  Durch die gegenseitige Ergänzung können neue Informationen, die für eine bestimmten Arbeitsablauf nötig sind, in einem Werkzeug dargestellt werden. Der „fliegende“ Wechsel zwischen den Systemen kann so vermieden werden.

Zusammenfassend erreicht man also eine hohe Korrektheit und Aktualität der Daten, die ohne die Verbindung beider Systeme nicht möglich wäre. Die negierten Vorteile des integrierten Systems sind die Nachteile der getrennten, nicht kommunizierenden Systeme.

5.4 Integrationstechniken

Im nächsten Kapitel werden Vorschläge zur Behebung von Defiziten in der Kommunikation zwischen Netzdokumentation und Netzmanagementsystem anhand konkreter Szenarios behandelt. Bei der Realisierung muß man entscheiden, auf welche Art der Integration sich die Implementierung abstützt [Abec94]. Man kann heute bei den auf dem Markt erhältlichen sogenannten „integrierten“ Managmentplattformen (z.B. Cabletron Spectum, Hewlett Packard OpenView, IBM NetView for AIX) unter drei Integrationstechniken wählen, die eine unterschiedliche Mächtigkeit haben. Die Integrationstechniken nutzen dabei die Schnittstellen der verschiedenen Bausteine eines Managementsystems. Um die Techniken den allgemein definierten Modulen einer Managementplattform zuzuordnen, seien diese kurz skizziert. Für eine ausführlichere Beschreibung sei auf [HeAb93] verwiesen:

- Oberflächenbaustein:
  Der Oberflächenbaustein stellt die Benutzeroberfläche zur Verfügung und bietet die Funktionalität der Map-Generierung.

- Managementanwendungen:
  Diese Anwendungen nützen die ihnen gebotene Umgebung der anderen Module um Einzelaspekte zu behandeln. Z.B. Mib-Browser, Event-Manager, Topologiemanager, usw.

- Kernsystem:
  Dieses Modul sorgt für die Kommunikation und Koordination der anderen Bausteine, d.h. Steuerung der Datenflüsse, Koordination der Anfragen der anderen Module, usw.

- Informationsbaustein:
  Dieser Baustein hält in strukturierter Form die managementrelevanten Informationen und stellt Dienste zum Kreieren und Verwalten von Managementobjekten bereit.

- Kommunikationsbaustein:
  Mit diesem Modul wird die Kommunikation zu entfernten zu managenden Komponenten oder auch zu anderen Managementstationen realisert. Es ist zuständig für die korrekte Protokollabwicklung (z.B. SNMP [Rose94], CMIP [ISO 9596]) und zur Bereitstellung der Angeforderten Informationen.

Dokumentierte Schnittstellen, die von „außen“, d.h. von externen Anwendungen angesprochen werden können, stellen der Oberflächen-, Informations- und Kommunikationsbaustein zur Verfügung.

5.4.1 Oberflächenintegration

Die Technik der Oberflächenintegration stellt die einfachste Form der Einbindung dar. Das einzige, was die beiden Anwendungen verbindet, ist eine gemeinsame Oberfläche über die Darstellung im Oberflächenbaustein des Managementsystems. Die Tatsache, daß der einzige zu sehende Integrationsaspekt darin liegt, den Aufruf eines anderen Programmes in die Oberfläche einzubinden, wird als „Oberflächenintegration“ bezeichnet.

Der Datenaustausch der Applikationen kommt dabei über die Übergabe der Aufrufenden, welches Objekt zu betrachten ist, meist nicht hinaus. Oft fehlt sogar dies, so daß aus der Managementplattform nur eine andere Applikation angestoßen wird.

Der Vorteil dieser Methode auf der Benutzerseite liegt in der (vordergründigen) Benutzung nur eines Werkzeuges, in dem die verschiedenen Teilapplikationen mit einem evtl. einheitlichen look-and-feel in einer einheitlicher Bedienweise eingebunden sein können. Auf der Implementierungsseite muß an der zu integrierenden Applikation nichts außer der Anpassung der Oberfläche verändert werden.

5.4.2 Datenintegration

Mit dieser Integrationsmethode wird der Informationsbaustein angesprochen. Es können so Daten aus einer zu managenden Komponente oder einer anderen Datenquelle dem Managementsystem verfügbar gemacht werden, das diese dann weiter verarbeitet und darstellt. Hierbei sind die zur Verfügung gestellten Daten ein Abbild der Informationsbasis, die von dem anderen Werkzeug gehalten werden.

Als Beispiel sei hier [ANPR94] angeführt. In dieser Arbeit wurden die von einem proprietären Managementwerkzeug über eine V.24-Schnittstelle abgegebenen Managementinformationen in den Datenbaustein von *Cabletron Spectrum* übernommen und als Komponenteninformationen zur Verfügung gestellt.

5.4.3 Kommunikationsintegration

Bei der Kommunikationsintegration werden die Informationen einer Ressource oder Werkzeug über ein standardisiertes Managementprotokoll (SNMP, CMIP) abgefragt. Voraussetzung dafür ist, das in der zu managenden Komponente ein Agent mit einer standardisiert definierten MIB-Beschreibung vorhanden ist.

Der Vorteil dieser Methode ist, daß sie einerseits die konzeptuell überzeugenste ist, da sie einen integrierten Managementansatz am nächsten steht und andererseits auf der technischen Seite von den Managementplattformen gut unterstützt wird.

Spezialfall: Proxyintegration

Einen Spezialfall stellt die Proxyintegration dar. Dabei steht eine zusätzliche Anwendung, der Proxy, zwischen den beiden zu integrierenden Werkzeugen/Ressourcen, da diese nicht auf einer Ebene kommunizieren können. So kann der Proxy als Vermittler auf der einen Seite die Kommunikationsschnittstelle des Netzmanagementsystems bedienen und auf der anderen Seite Daten von der Datenschnittstelle einer Ressource abfragen und diese evtl. sogar schon vorverarbeiten. Mittels des Proxys können auch Ressourcen mit proprietären Schnittstellen dem Managementsystem über die Kommunikationsschnittstelle verfügbar gemacht werden. Für eine ausführlichere Diskussion der Proxy-Problematik sein auf [HeAb93] verwiesen.

Kapitel 6

Betrachtung bestehender Systeme

Im folgenden werden beispielhaft die in der Betreiberumgebung BMW eingesetzten Netzdokumentations- und Netzmanagementsysteme betrachtet. Es werden zuerst im Überblick die Systeme und ihre Schnittstellen beschrieben, dann anhand von Szenarien Möglichkeiten aufgezeigt, wie durch die Anbindung Defizite des einen Systems durch die Fähigkeiten des anderen behoben werden können.

6.1 Beispiel für ein Netzdokumentationssystem: Cinema

Das bei BMW eingesetzte Netzdokumentationssystem heißt Cinema[1]. Diese Eigenentwicklung startete 1986 mit dem Ziel, einen Überblick über die Netzinfrastruktur zu bekommen. Mit Cinema wurde ein zentrales Werkzeug bereitgestellt, das die interne Verwaltung einzelner Abteilungen ersetzte und mit einem großen System die Redundanz in der Netzdokumentation verringerte. Es wird heute unternehmensweit eingesetzt und bietet neben der Netzdokumentation noch zwei anderen Module an. Die anderen beiden sind ein Trouble-Ticket-System und ein Installations- bzw. Umzugsmanagement. Die Zielsetzungen, die in jedem Modul verfolgt wurden, sind in Abbildung 6.1 aufgelistet [Cine94]. Dabei sind die Modul-

[1] Das Akronym steht für:

- **C**omputer
- **In**tegrated
- **Ne**twork
- **M**anagement and
- **A**dministration.

trennungen nur funktioneller Natur, auf Datenebene greifen alle drei auf denselben Datenbestand zu. Somit stellt das Werkzeug in seinem Funktionsbereich ein integriertes System dar.

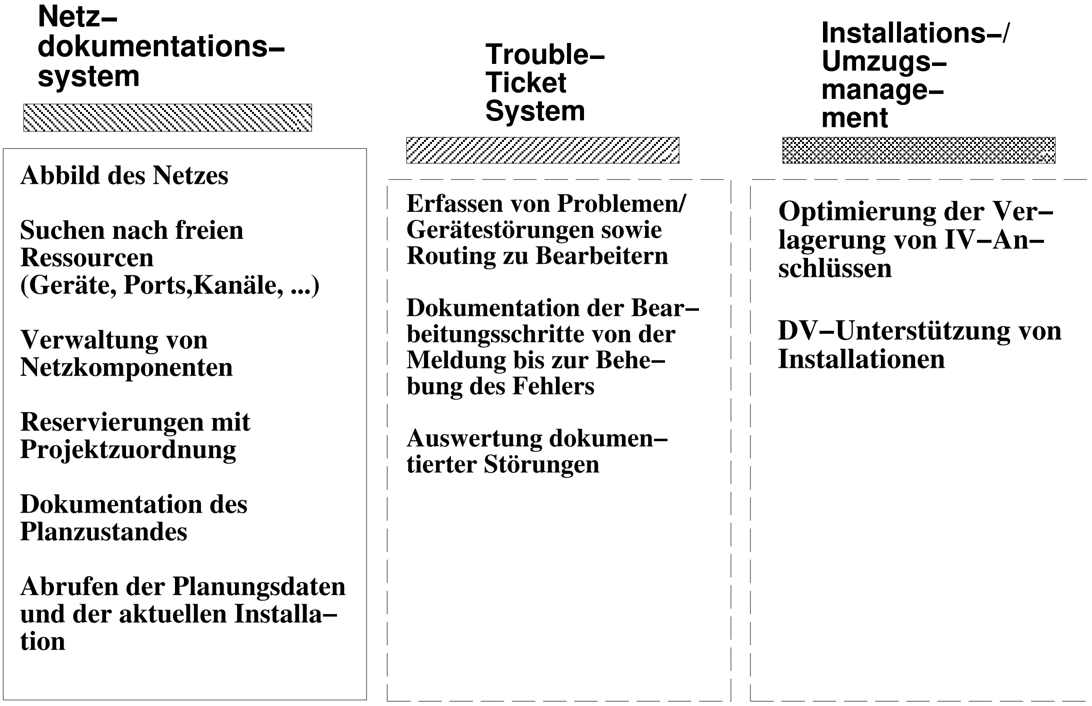

Abbildung 6.1: Die drei Module Cinemas und deren Zielsetzung

Cinema hat vielfältige Schnittstellen zu anderen Systemen, die jedoch meist lesend auf Cinema zugreifen. So stellt Cinema z.B. die grundlegenden Daten für das DNS-System der TCP/IP Umgebung bereit. Das ISYCON-System hat auch schreibenden Zugriff auf Cinema, die eingebrachten Daten haben jedoch keinen Einfluß auf die Netzdokumentation. Da hier nur die Anbindung zwischen der Netzdokumentation und dem Netzmanagementwerkzeug beschrieben wird, sei für eine Beschreibung der angebundenen Systeme auf [Weiß94] Kapitel 6.2 ff verwiesen.

Implementiert ist Cinema auf einem VM-Host[2] auf der Basis einer Oracle Datenbank, einem relationalen Datenbanksystem. Als Ausgabemedium wurden IBM 3270-Terminals gewählt. Diese werden heute noch benutzt oder aber unter anderen Betriebssystemen, z.B. unter dem *UNIX-X-Window-System*, emuliert.

Im folgenden werden das Trouble-Ticket- und das Umzugsmodul von Cinema nicht weiter betrachtet, sondern der Netzdokumentationsteil wird weiter vertieft. Eine

[2] Es laufen zur Zeit Bestrebungen von dem VM-Host auf ein anderes (moderneres) System zu wechseln. Genauere Informationen liegen z.Z. nicht vor.

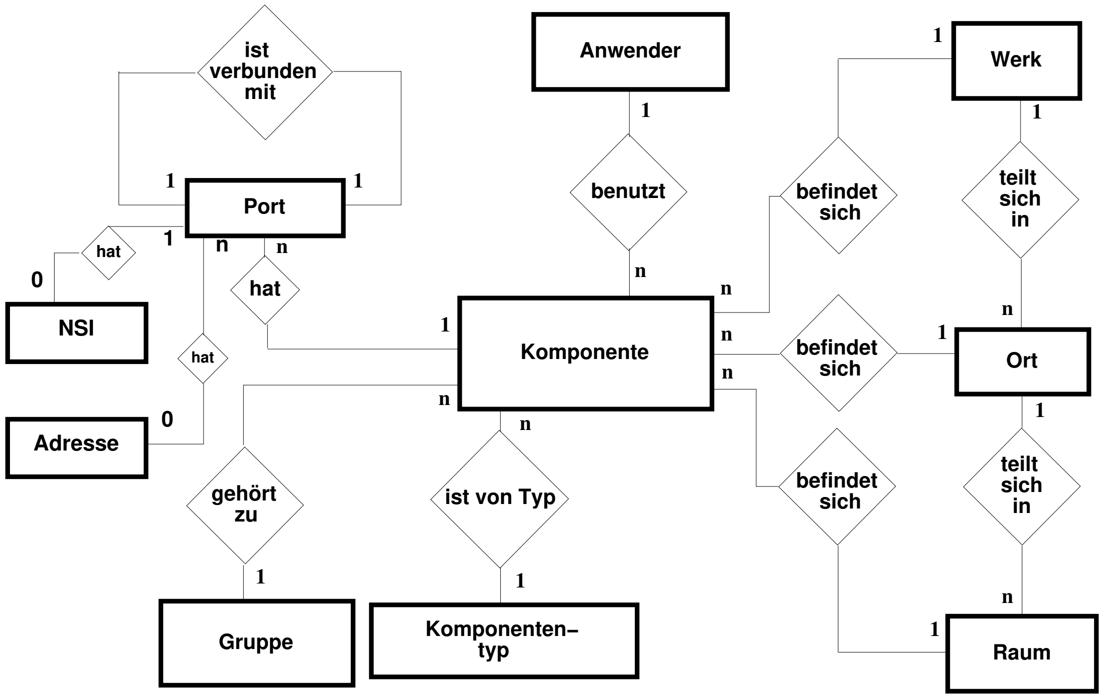

Abbildung 6.2: Entity-Relationship-Modell von Cinema

genaue Betrachtung des Trouble-Ticket-Moduls ist in [Bert95] zu finden.

Wie schon erwähnt läuft Cinema auf der Basis einer Oracle Datenbank. Das relationale Datenmodell wurde in der Diplomarbeit [Wied93] auf ein Entity-Relationship-Modell abgebildet. Da dieses Modell übersichtlicher und leichter verständlich ist, werden die in Cinema enthaltenen Daten anhand dieses Modells erläutert. In [Wied93] ist auch eine detaillierte Beschreibung der Tabellenstruktur gegeben, so daß hier versucht wird einen Überblick über die Daten zu geben.

Das E-R-Modell ist in Abbildung 6.2 abgebildet. Als zentrales Element ist die „Komponente" Mittelpunkt des Modells. Mehrere Komponenten können in einer weiteren Komponente über eine, in der Graphik nicht enhaltenen, Relation „eingebaut in" einer übergeordnete Komponente zugeordnet werden, wenn diese einen modularen Aufbau hat. Die wichtigsten Daten sind

- Hersteller, Typ und Modell,
- Seriennummer und Inventarnummer,
- Segment und Domäne,
- Eintrag, wer zuletzt an diesen Daten Änderungen vorgenommen hat.

Die Informationen über Anschlußports werden über eine „hat“ Relation mit der Komponente verknüpft. Hier sind Adreßinformationen die relevanten Attribute. Über die „ist verbunden mit“ Relation werden zwei Ports über eine Leitung verbunden. Über den zweiten Port kann die Komponente am anderen Ende der Leitung identifiziert werden. Die Entity „NSI“[3] bezeichnet die Verbindung mit einem NSI-Anschluß, wenn der Port mit einem Endanschluß verbunden ist. Die Adressen werden über die „Adresse“ mit dem Port verbunden, wenn dieser eine hat (z.B. haben serielle Anschlüsse keine Adresse).

Ein „Anwender“ wird zusätzlich zu seinem Namen mit der Abteilung und seiner Telefonnummer erfaßt. Mit den Entities „Werk“, „Ort“ und „Raum“ erfolgt eine eindeutige Zuordnung der Komponente an eine Lokalität. Ein Raum mit seiner Raumnummer ist somit eindeutig bestimmt durch seine Lage in einer „Etage“, „Gebäude“ und einem „Werk“. Mit der Entity „Ort“ sind Konsistenzinformationen abgelegt, die sicherstellen, daß die Angabe der Etage sich im gültigen Wertebereich befinden.

Über die Entity „Gruppe“ werden zusammengehörige Komponenten zusammengefaßt. Beispielsweise bilden alle Elemente eines CAD-Arbeitsplatzes eine Gruppe. So können die Komponenten schneller einander zugeordnet werden. Komponenten gleichen Typs, d.h. mit identischen Attributen „Hersteller“, „Typ“ und „Modell“, werden über die Entity „Komponententyp“ zusammengefaßt.

Zusätzlich existieren weitere Tabellen im relationalen Modell, die sich nicht in dieses E-R-Modell einordnen lassen. Dies sind u.a. Informationen über User-IDs und Logging-Informationen.

Einen Überblick über die Gestaltung der Oberfläche von Cinema kann man anhand der Abbildungen 6.6 bis 6.10 in Abschnitt 6.3.3 gewinnen.

Zusammenfassend sind also in Cinema die wichtigsten Daten erfaßt:

- Komponentendaten
- Anwenderdaten
- Standortdaten und
- Verkabelungsbeschreibungen.

Doch der große Nachteil von Cinema ist, daß der Datenbestand von Hand gepflegt werden muß. Es existiert keine Anbindungen zu Systemen, die Änderungen im Netz in Cinema mitprotokollieren würden. So wird der Anteil nicht korrekter Daten vom Betreiber selbst mit ca. 5% angegeben. Doch dieser Wert bezieht sich nur auf den Kernbereich des Netzes. Außerhalb dieses Bereiches ist dieser Wert wahrscheinlich höher.

[3] Network Service Interface: Über diesen Bezeichner werden die Netzanschlußdosen benannt.

Schnittstellenbeschreibung

Wenn man sich die Schnittstellen des Systems Cinema näher betrachtet, stellt sich heraus. daß die Möglichkeiten der Einflußnahme auf die SQL-Schnittstellen beschränkt sind (Abbildung 6.3). Die *PRO*REXX* Programme dienen nur dem Zweck der Oberflächengenerierung auf den 3270-Terminals, somit fallen diese für die Programmierung in der Unix-Umgebung aus. Dies stellt aber bezüglich der Schnittstellen insofern keinen Nachteil da, da der Kern von Cinema aus der Oracle-Datenbank und der darin programmierten Funktionalität besteht. Um auf die Datenbank zugreifen zu können, bietet Oracle zwei Möglichkeiten an. Beide basieren auf der standardisierten Sprache SQL.

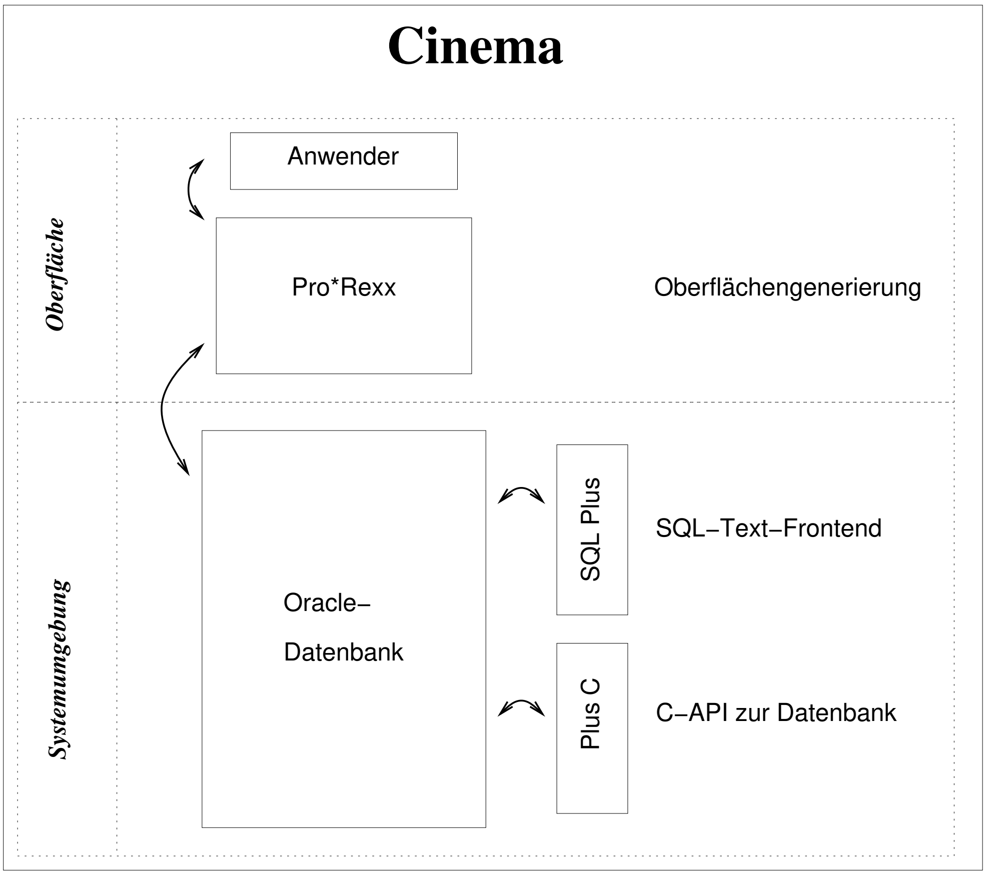

Abbildung 6.3: Schnittstellen von Cinema

- Die erste ist die inteaktive Eingabe in *SQL*PLUS*. Diese Schnittstelle bietet eine Shell-ähnliche, textorientierte Umgebung, auf die mit Batch-Files oder direkt zugegriffen werden kann. Der Vorteil liegt in der Einfachheit der

Abfragen, wie an dem Beispiel in Abschnitt 7.1 noch verdeutlicht wird. Die Ausgabe wird dabei in Tabellenform generiert. Der Nachteil besteht darin, daß die Einbindung in Programme schwer realisierbar ist. Um auf Einzelergebnisse zuzugreifen, bedarf es hier höherem Aufwand.

- Dies wird in der zweiten Möglichkeit dahingehend erleichtert, daß mit *PRO C* ein C-Präprozessor zur Verfügung gestellt wird, der es erlaubt, jetzt allerdings in komplizierterer Syntax, SQL-Statements in C-Programme direkt einzubinden. Da es in dieser Umgebung möglich ist, die Ausgaben direkt in Variablen abzulegen, ist die Weiterverwendung einfacher. Das Ergebnis sind dann kompaktere, allerdings schwerer lesbare, Programme.

Somit stehen von der Cinema-Seite der uneingeschränkte Zugriff auf die Datenbank zur Verfügung.

Beschränkt man sich bei der Darstellung nicht auf die Oberfläche von 3270-Terminals, so können auch noch die Tools Oracle Forms und Oracle Graphics benutzt werden, über die mit selbstgestalteten Oberflächen auf den Datenbestand zugegriffen werden kann.

6.2 Beispiel für eine Managementplattform: NetView

Die Managementplattform *NetView for AIX*[4] von IBM ist eine Weiterentwicklung des Systems *HP OpenView* von Hewlett Packard (HP). Seit dem Kauf des Source–Codes von HP entwickeln sich beide Produkte unabhängig voneinander. Die größten Veränderungen wurden am GUI[5] vorgenommen. Die Programmierschnittstellen wurden eins zu eins übernommen – die Aufrufe sind identisch mit denen von *HP OpenView*. Es wurde nur eine weitere Gruppe von GUI–Prozeduren hinzugefügt, die eine mächtige Schnittstelle zur Programmierung von Oberflächenkonstrukten bereitstellt.

Von NetView als einer integrierten Managementplattform zu sprechen, ist nur insoweit zu vertreten, als daß NetView ein Sammelsurium von Einzelprogrammen mit teilweise eigener Datenhaltung und nur rudimentärer Interkommunikation darstellt.

Der einführende Satz zu *Chapter 1* im „Administrators Guide“ trifft dies sehr treffend:

[4] Im folgenden immer mit der Kurzform „NetView“ bezeichnet

[5] Graphical User Interface

> „The NetView program uses many processes and databases to perform network management functions."[NvAd94]

Diese Programme arbeiten aber alle unter einer einheitlichen Oberfläche, so daß im Umgang mit dem System der Wechsel zwischen Einzelprogrammen nicht sofort auffällt.

Die in Abbildung 6.4 dargestellten Einzelprogramme stellen die Grobstruktur von NetView dar. Die im Hintergrund ablaufenden Deamons stellen die Informationen zur Verfügung, die dann graphisch angezeigt werden können. Die GUI wird vom Programm `ovw` bereitgestellt. Über die frei definierbare Menüstruktur können dann die `xnm*` Applikationen aufgerufen werden, die entweder weitere Subprozesse anstoßen, oder selber die gewünschten Funktionalitäten bieten.

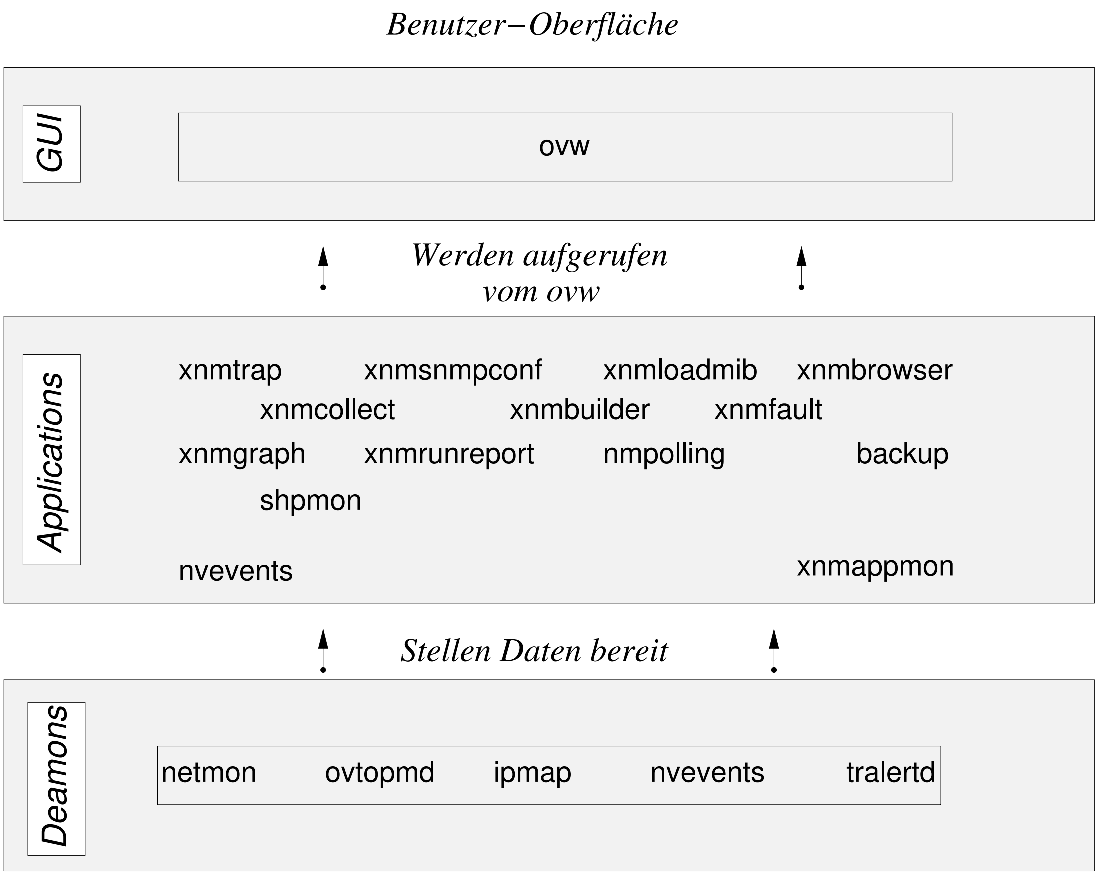

Abbildung 6.4: NetView: Übersicht der Einzelprogramme

Die genaue Beschreibung der Programme befindet sich in „Administrators Guide, Chapter 1", weswegen an dieser Stelle nur stellvertretend einzelne Applikationen aufgeführt sind, die typisch für den Aufbau von NetView sind.

`ovw` (Openview Window Manager)
: Stellt die graphische Oberfläche zur Verfügung, in die alle anderen interaktiven Programme eingebunden werden und ist verantwortlich für die Darstellung der Maps.

`ipmap`
: Diese Applikation läuft als Hintergrundprozeß zu `ovw` und sorgt für die Konsistenz der Abbildung der Maps und der Informationen aus der Datenbank.

  Andere Deamons laufen auch während der `ovw` nicht aktiv ist, und versorgen das Event–Log–File und die Topologie–Datenbank mit aktuellen Einträgen.

`nvevents`
: Die Darstellung der Events aus dem Log–File `ovevent.log` in den Variationen als Karteikarte oder als Liste wird von diesem Programm übernommen.

  Dieses Programm wertet die Informationen aus dem Flat–File aus und stellt sie dar.

`xnmbrowser`
: `xnmbrowser` ist ein MIB-Browser, über den mittels SNMP der Internet–Registrierungsbaum abgefragt werden kann. Auch hier ist wieder zu betonen, daß dieses Programm auch unabhängig vom `ovw` laufen kann.

  Dieses Programm gehört zu einer Gruppe, die Informationen aus den Komponenten über SNMP darstellen können. Eine Verbindung zwischen den Programmen besteht nicht, außer daß sie gemeinsam auf die MIB–Database zugreifen, in der die Definitionen der MIBs archiviert sind.

  Weitere Beispiele für diese Gruppe sind: `xnmtrap`, `xnmsnmpconf`, `xnmloadmib`, `xnmbrowser`, `xnmcollect`, `xnmbuilder`, `xnmgraph`

Für die Programme, die weitere Einzelaspekte behandeln, sei auf das oben angeführte Manual verwiesen.

Schnittstellenbeschreibung

In NetView gibt insgesamt drei Schnittstellen, mit Hilfe derer aus dem System Information gewonnen oder mit ihm in Kommunikation getreten werden kann (vgl. Abb. 6.5). Die drei Schnittstellen lassen sich wie folgt beschreiben:

1. **Oberflächenschnittstelle:**
   Durch einfache Modifikation der sogenannten ARF's (Application Ressource Files) können externe Programme in die Oberflächenstruktur eingebunden werden. Dabei kann genau spezifiziert werden, unter welchem Menüpunkt

sich der Programmaufruf eingliedern soll. Zusätzlich können Aufrufparameter übergeben werden, die sich aus den in der Map selektierten Icons zusammensetzt. Diese können mit Hilfe von Zusatzregeln genau auf ihre Anzahl (Minimal- und Maximalanzahl) und Art (z.B. „IsCisco“, „IsNode“, „IsInterface“) festgelegt werden. Entsprechen die Attribute nicht dem gewünschten, so kann der Menüpunkt erst gar nicht ausgewählt werden.

Weiter kann auch noch festgelegt werden, ob das Programm nur einmal bei Start des NetView aufgerufen wird, oder ob es nach dem Aufruf nochmals gestartet werden darf.

Diese „Schnittstelle“ dient hauptsächlich zur Initiierung von NetView-eigenen Dienstprogrammen oder aber auch zum Aufruf sog. integrierter Produkte von Fremdanbietern (z.B. CiscoWorks), oder von eigenen erstellten Applikationen.

Allgemeiner formuliert ist es über diese Methode möglich, *jedes* beliebige Programm in die Oberfläche zu „integrieren“.

Vom Autor wurde hierüber ein Shell-Skript eingebunden, daß es ermöglicht, von einem selektierten Objekt der Map seinen Standort über Cinema mit Hilfe von SQL*PLUS zu ermitteln. Es wird im Abschnitt 7.1 genauer vorgestellt.

2. **Application Programming Interface (API):**
   Über diese Schnittstelle ist eine weitreichende Manipulation der Darstellung und Speicherung möglich. Man kann die Map per Programm ergänzen oder Objekte gezielt verändern. Ebenso kann transparent auf die NetView-Datenbank zugegriffen werden, allerdings nicht über SQL-Statements, sondern über C-Strukturen, die an NetView-Systemaufrufe übergeben werden. Welche Hintergrundprozesse angestoßen werden, bleibt dem Programmmierer der API verborgen. Durch die äußerst vielfältige Auswahl an Aufrufen können einfache Spezialanforderungen schnell implementiert werden.

   Vom Autor wurde in diesem Zusammenhang ein Prototyp erstellt, das alle NetView bekannten IP-Adressen aus dessen Datenbestand extrahiert (siehe Abschnitt 7.2). Im Zusammenspiel mit einem zweiten Programm, welches alle IP-Adressen aus Cinema ausgibt, konnte durch Mischen eine Tabelle erstellt werden, welche Aufschluß darüber gibt, inwieweit in Cinema Inkonsistenzen bezüglich inaktiver IP-Adressen oder nicht vergebener IP-Adressen bestehen. Dieser Prototyp wird in Abschnitt 7.1 genauer vorgestellt.

3. **Dienstprogramme:**
   Einige Programme, die über die Oberfläche eingebunden sind, können auch ohne NetView als Utilities aufgerufen werden. Diese wurden aber bisher nicht von Autor näher betrachtet, da sie der Anforderung einer echten „Anbindung“ beider Systeme nicht gerecht wird.

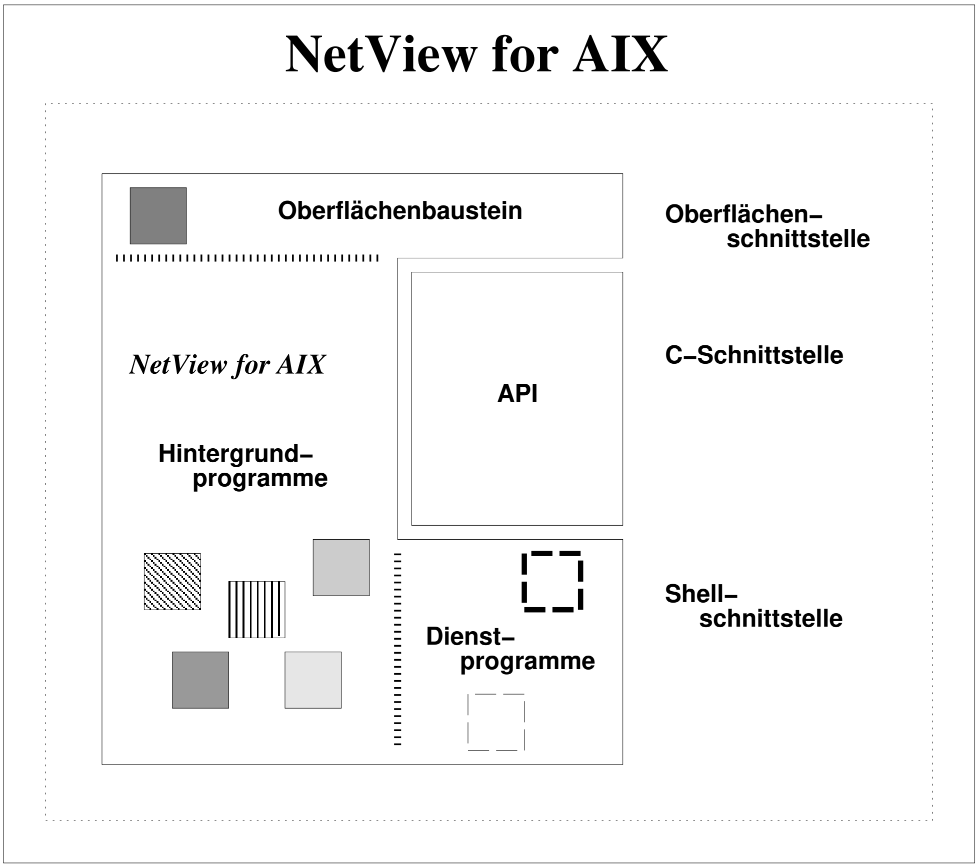

Abbildung 6.5: Schnittstellen von NetView

Generell gilt, daß es zur Herstellung einer Verbindung zwischen den Systemen immer externer Programme bedarf, die die Funktionalität für die Vermittlung bereitstellen. Es ist zwar von NetView vorgesehen, externe Datenbanken zu unterstützen, aber nicht, sich Informationen aus verschiedenen fremden Quellen zu beschaffen. Auf die externen Datenbanken greift NetView – nach derzeitigen Informationen – nur schreibend zu, indem das System dorthin Daten aus seinen flat-file-Datenbank kopiert. Änderungen in der Datenbank ändern somit nicht die NetView benutzten Daten.

6.3 i

Analyse der zwischen Cinema und NetView auszutauschenden Informationen Allgemein gibt es drei Arten von Datenflüssen, die zwischen der Netzdokumentation und dem Netzmanagementsystem möglich sind. Diese sind:

1. **Datenübertragung von Cinema nach NetView:**
   Daten werden anhand eines gemeinsamen Schlüssels in Cinema identifiziert und entweder in die Netzmanagementdatenbank mit aufgenommen oder an der Oberfläche präsentiert. Ein anderer Weg wäre, auch das Netzmanagementsystem mit seiner umfangreichen Funktionalität als Transitsystem von Daten aus der Netzdokumentation in einzelne Netzkomponenten zu benützen.

2. **Datenübertragung von NetView nach Cinema:**
   Die Identifikation erfolgt wie beim vorigen Fall, mit dem Unterschied, daß infolge des Maskensystems der 3270-Emulation man sich hauptsächlich auf die Modifikation vorhandener Tabelleneinträge beschränken wird.

3. **Kombination von Daten aus beiden Systemen:**
   Hiermit können die Stärken beider Systeme miteinander vereint werden, um aus der Verknüpfung bisher nicht erfaßbare Daten zu gewinnen. Als Beispiel sei auf die IP-Adreßproblematik hingewiesen, die in Abschnitt 6.3.5 genauer vorgestellt wird.

Die Grundlage für die Analyse der auszutauschenden Daten ist die Betrachtung der einzelnen, in den Systemen enthaltenen, Informationen.

Zuerst sollen daher die Informationen, die NetView zur Verfügung stellt, dargestellt werden und danach die von Cinema.

In einem Netzmanagementsystem gibt es naturgemäß zwei Arten von Daten, nämlich einerseits die Daten, die zur internen Verwaltung und graphischen Darstellung des Netzes unbedingt gebraucht werden, und andererseits die, die aktuell auf Anforderung hin von den Netzkomponenten ermittelt werden müssen.

Beispiele sind hierfür einerseits die IP-Adresse, IP-Name, Standort, letzter ermittelter Status und diverse Eigenschaften, wie „Ist-Ein-Router“, „Kann-SNMP“, „Ist-Von-Cisco“ uvam. Diese Informationen müssen immer (lokal) zur Verfügung stehen, da sie zur eindeutigen Identifizierung der Komponente dienen und auch dann zur Verfügung stehen müssen, wenn das Gerät nicht mehr ansprechbar ist.

Typische Vertreter der anderen Klasse sind Daten über die aktuelle Auslastung der Komponente, Fehlerraten, Daten also, die allgemein nur aktuell interessant sind (abgesehen natürlich für die Anfertigung von Statistiken, wo aber wiederum die Ansammlung von Momentanwerten interessiert).

Wenn die Netzkomponenten es zulassen, sind alle bisherigen Daten automatisch durch die „Autodiscovery“-Funktionalität abgedeckt, so daß für Konfiguration des ganzen Systems nur „Eckwerte“ manuell zu erfassen sind. Doch die eben erwähnte Einschränkung „wenn die Netzkomponenten dies zulassen“ stellt den größten Nachteil dar. Wenn nämlich eine Komponente nicht managebar ist, können darüber auch keine Informationen gewonnen werden und eine manuelle Konfiguration muß dann wieder manuell gepflegt werden.

Da wahrscheinlich auch in Zukunft Verteilerschränke, Netzanschlußdosen (NSI[6]), und Inventarnummern nicht managebar sind, kann man bei der Erstellung und Pflege eines Inventarisierungs- und Netzdokumentationssystemes auch in Zukunft nicht auf menschliche Pflege verzichten. So stellt dieses System auch die einzige Möglichkeit dar, *alle* aktiven und passiven Elemente des Netzes zu erfassen und physische Wege detailgenau zu verfolgen. Zusätzliche betriebswirtschaftliche Informationen, die z.B. zur Inventarisierung gebraucht werden, finden sich auch nur hier.

Um nun konkrete Anforderungen zu finden, wurden die Mitarbeiter verschiedener Arbeitsbereiche gefragt, wo sie Defizite im Umgang mit dem Dokumentationssystem und den Netzmanagementplattformen sehen. Da zur Zeit nur *Cabletron Spectrum* produktiv eingesetzt wird, bezogen sich die Anforderungen auch auf dieses System. Die Problemlösungen sind jedoch plattformunabhängig formuliert, so daß erst bei der Realisierung das System zum Tragen kommt.

Folgende Problemstellungen wurden dabei erörtert:

6.3.1 Verfolgung von Störungen

Um den Weg einer Störung bis zum Urheber verfolgen zu können, muß man die Informationen der einzelnen Geräte logisch – wie sie untereinander vernetzt sind – und physisch – wo sie stehen – kennen.

Da die Funktionalität, dies in einem Netzmanagementsystem zu verfolgen, z.Z. nicht gegeben ist, müssen die Pfade im Netzdokumentationssystem per Hand verfolgt werden. Dabei liegt es nicht daran, daß die Programme nicht in der Lage dazu wären, sondern der Grund ist, daß die Systeme nicht in diese Richtung konfiguriert sind.

Heute wird ein Weg von einem Router zu einem Endgerät wie folgt ermittelt: Ausgehend von einem Router, der im Netzmanagement vorhanden ist, wird in Cinema derselbe Router aufgerufen. Dann wird mittels der Interfaces die Verbindung und darauf das Gerät selbst ermittelt. Will man nun in einer kaskadierten Hub-Struktur die tiefer liegenden Geräte ermitteln, muß wieder der Weg `Gerät A -> Interface A -> Verbindung A zu B -> Interface B -> Gerät B` in Cinema nachvollzogen werden. Doch da man in Cinema maximal acht Schritte von einer Ausgangsmaske gehen kann, kann man oft nicht so weit die Struktur 'hinunter' verfolgen, wie es vonnöten wäre, um mittels dieser Suchmethode das Endgerät in der Kaskade zu finden.

[6]Network Service Interface, BMW eigene Bezeichnung

Ziel wäre es, mit einer Applikation in Cinema diese Pfade leichter zu ermitteln und die relevanten Informationen anzuzeigen. Von Vorteil wäre es, die Hubhierarchien zu visualisieren. Aber schon allein eine textuelle Anzeige wäre nicht nur als Interimslösung zu betrachten, da voraussichtlich nicht alle Hubs in naher Zukunft mit Agenten ausgestattet werden können (Kostengründe). Wären alle Hubs SNMP-fähig und würden die Funktionalität erfüllen Nachbargeräte zu erkennen, könnte man mit einem Blick auf die Managementplattform den Weg verfolgen und die Fehlerquellen über Schwellwerte ermitteln. Doch dies wird auch nach der Installation von Agenten nicht möglich sein, da diese Anforderungen vom Agenten (noch) nicht erbracht wird.

6.3.2 Aufnahme von MAC-Adressen in Cinema

Die Zuordnung MAC-Adressen zu Komponenten stellt einen wichtigen Punkt bei der Verfolgung von Fehlern dar. Die MAC-Adresse stellt die eindeutige Identifizierung einer Komponente auf der Schicht 2a des OSI-Modells dar. Alle Adressierungen in den höheren Schichten werden über Packete mit MAC-Adresse übertragen. Somit sind die Pakete, die letztendlich auf dem physischen Medium übertragen werden – sei es eine Koaxleitung oder ein Lichtwellenleiter – eindeutig über die MAC-Adresse in einem Segment dem Sender und Empfänger zuzuordnen.

Ein Szenario, das verdeutlicht, warum MAC-Adressen in Cinema aufgenommen werden sollen, wird im folgenden vorgestellt:

> Wenn ein Anwender eine Störung meldet, müssen zuerst mögliche Fehlerquellen ermittelt werden. Wenn sich herausstellt, daß die Ursache nicht an einem überlasteten Server oder an der Anwendung selber liegen könnte, sondern an einer Netzkomponente, wird anhand des Sniffers, einem Netzanalysator, festgestellt, welche Komponenten im Segment des Anwenders für diese Störungen in Frage kommen könnten. Da der Sniffer in jedem Fall MAC- und seltener IP-Adressen liefert, ist es unmöglich in Cinema anhand dieser Adresse den Standort des Gerätes zu ermitteln, da dort die IP-Adresse und nicht die MAC-Adresse erfaßt ist. Um dennoch den Standort zu bestimmen, geht man den Weg über die ARP-Tables bei IP-Geräten, oder über „dynamisches Einkreisen" der non-IP-Geräte, indem einzelne Segmente kurzzeitig abgetrennt werden und mit dem Sniffer ermittelt wird, ob die störende Komponente sich im abgetrennten Segment befindet.

Mit der Pflege der MAC-Adressen in Cinema kann man die störende Komponente sofort über eine einzige Abfrage ermitteln. Ein Problem bei der Betrachtung stellen die vielen Arten von Komponenten dar, die sich im Netz befinden. Es lassen sich folgende Typen unterscheiden:

- TCP/IP-Komponenten mit Agenten (z.B. Worstations, Router)
- TCP/IP-Komponenten ohne Agenten (z.B. PCs)
- DecNet-Komponenten mit IP-Zugang
- DecNet-Komponenten ohne IP-Zugang
- AppleTalk-Komponenten mit IP-Zugang
- AppleTalk-Komponenten ohne IP-Zugang

Die Ermittlung der zugehörigen MAC-Adresse fordert von der Managementplattform NetView verschiedene Verfahren des Autodiscovery. Bei TCP/IP-Komponenten mit Agenten und bei den Komponenten mit IP-Zugang ist eine Abfrage über SNMP möglich. Die Daten sind in der Standard-MIB `mib-2` enthalten, aber auch nach dem Autodiscovery als Kopie in der lokalen NetView-Datenbank. Bei TCP/IP-Komponenten ohne Agenten ist eine Abfrage der ARP-Tabellen des nächsten Routers vonnöten, in denen die Abbildung von IP- auf MAC-Adressen stattfindet, auch diese Komponenten werden in der lokalen Datenbank dokumentiert. Bei Komponenten ohne IP-Zugang gibt es für NetView keine Möglichkeit an die Adresse zu kommen. Eine Möglichkeit könnte sich noch ergeben, indem eventuell vorhandene Schnittstellen der proprietären Managementsysteme von DEC und Apple genutzt würden. Da der Autor keinen Zugriff auf diese Systeme hat und die Erfassung sich zunächst nur auf IP-Ebene bewegen soll, wurden keine näheren Untersuchungen vorgenommen.

NetView erkennt also alle zu erfassenden Komponenten im IP-Bereich, womit auch schon die Frage geklärt ist, mit welcher Methode die Daten erfaßt und zur Verfügung gestellt werden sollen. Da NetView in regelmäßigen, vom Benutzer einstellbaren, Intervallen einen Re-Scan des Netzes vornimmt, ist es auch gewährleistet, daß die NetView-Datenbank aktuell ist und Änderungen registriert werden. Allerdings ist die Belastung des Netzes mit Managementinformationen um so höher, je kürzer die Intervalle für den Re-Scan sind. Dies ist aber ein allgemeines Problem von Managementplattformen, und soll daher nicht weiter verfolgt werden.

Die MAC-Adressen können also aus der NetView-Datenbank abgerufen und dann in Cinema eingetragen werden. Hier bieten sich verschiedene Vorgangsweisen an:

- Bei nicht vorhandenen Einträgen wird automatisch „802.3“ im „System“-Feld und die MAC-Adresse im „Adresse“-Feld der Komponentenmaske eingetragen.
- Bei vorhandenen Einträgen, die sich geändert haben, können diese entweder sofort geändert und in einem Log-File mitgeloggt werden, oder erst gesammelt und nach Zustimmung geändert werden.

- Parallel zu den obigen Vorgängen kann ein Log-File mitgeführt werden, das den Ablauf des Programmes und seine Änderungen in Cinema dokumentiert. Auf diese Weise kann das Verhalten der Applikation kontrolliert und nachvollzogen werden.

Da die zu erstellende Applikation nach Erfahrungen des Autors mindestens 20 Minuten den NetView-Rechner unter Vollast beansprucht, sollte diese Applikation möglichst nachts oder am Wochenende laufen.

Eine Alternative, um die Last auf dem NetView-Rechner zu reduzieren, ist, daß nicht der Weg über die C-Schnittstelle von NetView gewählt wird, sondern direkt auf die von NetView unterstützte Datenbank zugegriffen wird.

Beide Systeme, Cinema und NetView, arbeiten bei BMW auf Basis von Oracle-Datenbanken. In einer Testumgebung laufen Cinema und NetView auf ein und derselben Datenbank.

Um einen Kompromiß zwischen den zu tolerierenden Inkonsistenzen der Adressen und dem Updatezeitraum zu treffen, ist ein Vorschlag anfangs den Zeitraum auf eine Woche festzusetzen und diesen dann gegebenenfalls nach oben oder unten zu korrigieren. Sollten größere Änderungen im Netz stattfinden, ist es nützlich den Prozeß auch auf Anforderung anzustoßen.

6.3.3 Aufnahme von Informationen in das Netzmanagementsystem

Oft sind die Informationen, die das Netzmanagementsystem zur Verfügung stellt nicht ausreichend. Informationen über passive Komponenten, wie z.B. Verteilerschränke, können nur über das Dokumentationsssystem abgerufen werden. (Vergl. Abbildung 6.10)

Andererseits enthält das Dokumentationssystem Daten, die auch in den Netzkomponenten konfiguriert werden können. So sind die Daten der `mib2` sysContact und sysLocation oft in den Komponenten nicht gepflegt, werden aber von jeder Managementplattform ausgelesen und dargestellt.

Daher stellt sich die Aufgabe, einen Weg zu finden, von *einem* Werkzeug, hier NetView, aus alle relevanten Daten darzustellen. Die Komponenten lassen sich aufgrund der Informationen, die sie selbst bereitstellen in vier Klassen einteilen:

- Komponenten mit SNMP-Agenten: Diese Komponenten liefern alle in ihren MIBs enthaltenen Daten, wobei es hier keine Unterscheidung zwischen TCP/IP und anderen Protokollen gibt, da diese auch eine eigene IP-Adresse für Managementzwecke haben.

- Komponenten ohne SNMP-Agenten (IP-Protokoll): Die sogenannten „Pingable Devices“ stellen keine Informationen bereit, von ihnen ist nur die IP-Adresse und MAC-Adresse vorhanden. Sie werden aber im Netzmanagementwerkzeug angezeigt.

  Komponenten im IP-Netz ohne IP-Adresse (z.B. nicht managebare Hubs) bleiben unentdeckt und somit unsichtbar.

- Komponenten ohne SNMP-Agenten (Nicht-IP-Protokoll): Diese Komponenten können aufgrund ihres fremden Protokolls nicht erkannt werden. Sie stellen folglich keine eigenen Informationen bereit und werden auch nicht angezeigt.

- Passive Komponenten: Da diese keine aktiven Teilnehmer am Netz sind, bleibt zu ihrer Dokumentation nur die Netzdokumentation. Beispiele sind hierfür Verteilerschränke, Kabelschächte, Kabel, usw.

Die dem Managementwerkzeug zur Verfügung stehenden Informationen haben eine gemeinsame Eigenschaft: Sie sind komponentenbezogen. Verbindungsbezogene Informationen sind nicht vorhanden. Diese Informationen über die physischen Verbindungen sind nur in der Netzdokumentation, die logischen Verbindungen sind auch in der Datenbank des Netzmanagentsystems enthalten. Die folgenden drei Informationsquellen sind die, auf die möglichst einfach zugegriffen werden muß:

- Netzdokumentation
- Netzmanagemensystem
- Komponenten-MIBs

Als zentrales System wird die Managementplattform gewählt, um die Informationen anzuzeigen, das aus den verschiedenene Systemen entweder die Informationen nur anzeigt oder in die eigene Datenbank übernimmt. Zusätzlich zu der logischen Darstellung des Netzes sind folgende Daten der Komponenten relevant:

Übersicht der Daten einer Komponente: In der Maske „Standard IV-Anschluß“ von Cinema sind Daten über eine Komponentem Anschlußverkabelung, NSI-Nummer, Kostenstelle, Netzadressen, Betreuer, Inventarnummer, usw. übersichtlich dargestellt. Diese Daten sind zum größten Teil spezifisch für das Dokumentationssystem und nur von dort abzufragen. (Beispiel siehe Abbildung 6.6)

Übersicht aller Ports einer Komponente und Adreßzuweisung zu den Ports: In den Masken „Ports“ und „Adresse“ werden in Cinema die in einer

```
 ----- C I N E M A --- IV ------ Standard IV-Anschluss -----------------------
 Funktion: ___
--------------------- Netzwerkkomponente ------------------------------------
 Herst. LANNET____ Typ LE140XTN__   SN 9346105_____________  Port 08_________
-------Tertiaerverkabelung----------------------Dose------------------------
    VT 148_      Port 2A2_________ IDENT 02HW Typ DOSE______ Proj. __________
                                 ! Standort 1.4__ / 8.0__ - 2_ - S__ - 295___
--------------------- N S I -------------------------------------------------
   NSI 02HW - 1_  Kostenstelle 5267_ FI-21_____ A-Typ E Anschluss:___Ethernet

        System      Adresse                             !   SCHMIDL/QC26959
        TCP/IP____  160.50.13.30___                     !   FI-21/WKZ_
        IP-NAME___  NETMGR_________                     !   34078_____
        IP-ALIAS__  ZWIESEL________                     !   15-MAY-1995
        802.3_____  02608C2E497F___                     !   MUCTP___
--------------------- Endgeraet ---------------------------------------------
 Herst. IBM_______ Typ 7013______ Mod 550_______            Port AUI1________
 SerNr. 4490598_____________     InvNr. 457156  Projekt __________    Status I
Stammnr XXXXXX    Name SCHMIDL_________  Abt FI-21_____    Tel 382-34078______
Bemerk. ____________________________________________________________
-----------------------------------------------------------------------------
1-Komp. 2-Verk. 3-Grp. 4-Port 5-Adresse         10-Zusatz 11-Clear PF12-Exit
13-KOU 14-Gen. 15-IV  16-NSI 17-IVAS                     23-Druck PF24-Help
1 Adressen mit 1 Verkabelungen gelesen                                LEVEL 1
```

Zu sehen sind in dieser Maske die wichtigsten Informationen, die mit einem Endgerät assoziiert sind. Ausgehend von der letzten Netzkomponente LANNET LE140XTN wird die Verbindung über den Port 08 zum Endgerät hergestellt. Diese läuft erst durch den Verteilerschrank VT 148 über Port 2A2 bis zur Anschlußdose 02HW, die auch als NSI (Network Service Interface) bezeichnet wird. Die Abrechnung z.B. erfolgt über das NSI. Dann wird das Endgerät NETMGR in seiner Netz-Konfiguration und dem Benutzer beschrieben. Abgeschlossen wird die Maske durch die Typbezeichung und Inventarnummer des Endgeräts.

Abbildung 6.6: Beispiel: „Standard IV-Anschluß“ in Cinema

Komponente installierten Interfaces mit ihren Ports und zugehörigen Adressen dokumentiert (vgl. Abbildung 6.7 und Abbildung 6.9). Die Informationen „Hersteller“, „Typ“, „Seriennummer“, „Port“, „Protokollzugehörigkeit“ und korrespondierenden „Adresse“ sind standardisierte Informationen, die auch von der (managebaren) Komponente selbst zur Verfügung gestellt wird. Die korrespondierenden Werte in der `mib2` sind ifDescr (Hersteller, Typ), ifType (Protokollzugehörigkeit) und ifPhysAddress (Adresse), weitere Informationen sind dann komponentengruppenabhängig in den anderen Standard-MIBs zu suchen.

```
 ----- C I N E M A --- PO ------ Port ------------------------------------20>

 Funktion:                                         PI X   LI X   II X   GI X
                                                   PP     LP     IP
  Herst.: CISCO       Typ: 7000          Ser-Nr.:   07000YF
  System: DECNET      Adr: 3.52             Port:

F Port         Hersteller Typ        SerienNr/&Adresse    Port         VS Stan
--------------------------------------------------------------------------- ---
  CONSOLE      XYPLEX     1600       88837                3            PI 1.5/
  FDDI0/0
  SER1/0
  SER1/1       TIMEPLEX   LINK       2103-IVZ-3 GEB.16    1:10:2       PI 1.5/
  SER1/2       TELEKOM    DAG64K12   577/10080-A          X.21         PI 1.5/
  SER1/3
  SER2/0       TIMEPLEX   LINK       D2757-IVZ-3 GEB.16   1:15:2       PI 1.5/
  SER2/1       TIMEPLEX   LINK       2103-IVZ-3 GEB.16    1:16:1       PI 1.5/
  SER2/2       SYMPLEX    DATAMIZER4 55411114             PORT-C       PI 1.5/
  SER2/3       SYMPLEX    DATAMIZER4 55403127             PORT-D       PI 1.5/
  SER3/0       TIMEPLEX   LINK       D2757-IVZ-3 GEB.16   1:15:1       PI 1.5/
--------------------------------------------------------------------------- vvv
1-Komp. 2-Verk. 3-Grp. 4-Port 5-Adresse          10-Zusatz 11-Clear PF12-Exit
13-Endgeraet 14-Next_Adress                                 23-Druck PF24-Help
23 Ports  ; 19 Verbindungen ; 4 freie Ports                          LEVEL 2
```

Diese Maske beschreibt die Portbelegung eines Cisco Routers. Die Port-Spalte beschreibt Listet die eingeschobenen Interface-Karten (CONSOLE, FDDI0, SER1 bis SER3) mit ihren Ports (Index 0 bis 3) auf.

In den Zeilen sind die Endpunkte der Verbindung diese Ports aufgelistet.

So ist beispielsweise der Port SER1/1 mit einem TIMEPLEX an dessen Port 1:10:2 verbunden.

Abbildung 6.7: Beispiel: Port-Maske in Cinema

Verkabelungsinformationen bis zur nächsten aktiven Komponente: In der Maske „Verkabelung“ in Cinema werden alle (passiven) Zwischenkno-

```
 ----- C I N E M A --- PO ------ Port ------------------------------------20>

 Funktion:                                        PI X   LI X   II X   GI X
                                                  PP     LP     IP
  Herst.: CISCO       Typ: 7000          Ser-Nr.:   000232
  System: DECNET      Adr: 3.140            Port:

F Port         Hersteller Typ        SerienNr/&Adresse    Port         VS Stan
----------------------------------------------------------------------- ---
  ETH4/5       LANNET     LE140XTQ   9393019              07           PI 1.5/
  FDDI0
  FDDI0/0
  FDDI1/0      LANNET     LFD-104    9325178              1            PI 1.5/
  FDDI1/1      LANNET     LFD-104    9325178              2            PI 1.5/
  TR2/0        IBM        8228       23052GB              1            PI 1.5/
  TR2/0-V
  0            CISCO      CX-FIP-MM  00865872             BUS          PI 1.5/
  1            CISCO      CX-FIP-MM  01101373             BUS          PI 1.5/
  2            CISCO      CX-TRIP4   00673687             BUS          PI 1.5/
  3            CISCO      CX-EIP6    01316380             BUS          PI 1.5/
------------------------------------------------------------------------- vvv
1-Komp. 2-Verk. 3-Grp. 4-Port 5-Adresse          10-Zusatz 11-Clear PF12-Exit
13-Endgeraet 14-Next_Adress                               23-Druck PF24-Help
```

Zusätzlich zu den in Abbildung 6.7 beschriebenen Ports sind hier (allerdings von einem anderen Router) die Routerinterfaces 0-3 aufgelistet, die aber noch nicht alle Interfaces dieser Komponente sind. Sie sind symbolisch über einen Port BUS miteinander verbunden.

Abbildung 6.8: Beispiel: Router-Interfaces in Cinema

ten bis zur nächsten Komponente angezeigt. Diese Informationen sind wie im ersten Punkt spezifisch für ein Dokumentationssystem. (Beispiel siehe Abbildung 6.10)

Mittels einer Applikation, die über einen Menüpunkt aufgerufen wird, sollen nun die oben erwähnten Daten für eine ausgewählte Komponente in NetView angezeigt werden, ohne das System zu wechseln. Für die Anzeige der Komponentendaten und der Verkabelungsinformation muß dafür für den Anwender transparent eine Verbindung zu Cinema aufgebaut, die entsprechenden Abfragen abgesetzt und das

```
----- C I N E M A --- AD ------ Adresse ------------------------------------

Funktion:

 Herst.: CISCO       Typ: 7000            Ser-Nr.: 07000YF


F Port         System     Adresse
-------------------------------------------------------------------------- ---
  FDDI0/0      TCP/IP     160.50.1.13
  FDDI0/0      IP-NAME    BR7013
  SER1/0       TCP/IP     192.109.63.129
  SER1/0       IP-NAME    BR0095
  SER1/1       TCP/IP     192.109.63.177
  SER1/1       IP-NAME    BR164
  SER1/2       TCP/IP     192.109.63.145
  SER1/2       IP-NAME    BR0126
  SER1/3       TCP/IP     192.109.63.193
  SER1/3       IP-NAME    BR0262
  SER2/0       TCP/IP     192.109.63.81
-------------------------------------------------------------------------- vvv
1-Komp. 2-Verk. 3-Grp. 4-Port 5-Adresse          10-Zusatz 11-Clear PF12-Exit
               15-IV  16-NSI                               23-Druck PF24-Help
30 Adressen gelesen                                                 LEVEL 2
```

In dieser Maske ist die Adreßverteilung an einem Cisco Router dokumentiert. Die Definition der Port-Spalte ist analog Abb. 6.7. Die Spalte „System" definiert die Form, in der die Adresse in der „Adresse"-Spalte aufgeführt ist. Bei Mehrfachadressierungen, hier Hostname und IP-Adresse, ist derselbe Port mehrmals vorhanden.

Abbildung 6.9: Beispiel: Adreß-Maske in Cinema

Ergebnis in NetView angezeigt werden.

Dazu soll ein zusätzlicher Menüpunkt in NetView eingefügt werden, der dann anhängig von dem ausgewählten Icon Abfragen absetzt. Für einen ausgewählten Port sollen die Verkabelungsdaten angezeigt werden. Für eine Komponente die Übersicht der Komponentendaten oder, wenn die Komponente nur einen Anschluß am Netz hat, alternativ die Verkabelungsdaten.

In einer genauer zu klärenden Abschätzung ist eine Entscheidung zu treffen, in-

```
 ----- C I N E M A --- VK ------ Verkabelung ------------------------------20>

 Funktion:

F Hersteller Typ        Serien.Nr/&Adresse   S Port          VS Standort
------------------------------------------------------------------------- ---
  LANNET     LE140XTN   9346105              I 08            PI 1.4/8.0-2---INF
  QUANTE     VT         148                  I 2A2           II 1.4/8.0-2-S-343
* BMW        DOSE       02HW                 I 1             PI 1.4/8.0-2-S-295
  IBM        7013       4490598              I AUI1             1.4/8.0-2-S-295









------------------------------------------------------------------------- ---
1-Komp. 2-Verk. 3-Grp. 4-Port 5-Adresse          10-Zusatz 11-Clear PF12-Exit
               15-IV  16-NSI                               23-Druck PF24-Help
1 Port verkabelt;                                                    LEVEL 2
```

Diese Maske stellt die Port-zu-Port Verbindung des LANNET-Hubs zu der IBM-Workstation dar. Vom Port 08 des Hubs verläuft die Verbindung über Port 2A2 des Verteilerschranks zur Dose 02HW. Dort ist dann direkt die IBM angeschlossen.

Abbildung 6.10: Beispiel: Verkabelung-Maske in Cinema

wieweit die Anzeige der Ports „online“ aus Daten der Komponente bestehen soll, oder auf die in Cinema dokumentierten Daten zurückgegriffen werden soll. Die erste Variante ist vom Ansatz her die bessere, da die Daten in der Komponente nie falsch sein können, jedoch kann in einer Fehlersituation die Komponente nicht erreichbar sein und somit keine Informationen zur Verfügung stellen. Die zweite Variante ist mit weniger Aufwand verbunden, da die Verbindung zu Cinema sowieso aufgebaut werden muß. Man begibt sich jedoch in eine unnötige Abhängigkeit zu Cinema und nimmt mögliche Inkonsistenzen der dokumentierten und realen Daten in Kauf.

Es könnte jedoch sein, daß die Komponente nicht managebar ist, aber trotzdem eine der obigen Masken in Cinema besitzt. In diesen Konflikt mit der Bereitstellung der Informationen durch die Komponenten kommt man jedoch deswegen nicht, da eine Komponente solcher Art dann auch nicht in den Views von NetView zu sehen ist (Oder gibt es einen Router als „pingable device"?) und somit sowieso ein direkter Zugriff auf Cinema erfolgen muß.

6.3.4 Abgleich der Routerinterfaces mit Cinema

Eine zentrale Rolle im Netzbetrieb von BMW stellt das Cisco-FDDI-Router-Backbone dar. Es gibt ca. 70 Router mit je zwei bis zwölf Einschüben für Interfacekarten mit je ein bis vier Ports. Diese Interfacekarten werden bei Defekt oder Update ausgetauscht. Die Dokumentation dieser Aktionen erfolgt in Cinema manuell. Zur einfacheren Abwicklung soll dies nun rechnerunterstützt realisiert werden. Wichtige Fragen wie

- Wieviele Ports sind an diesem Router noch frei?
- Welche Router sind mit der neuesten Hardware ausgestatten?
- Welche Router arbeiten mit veralteter Hardware?

können nur mit aktuellen Daten beantwortet werden. Da die Router durchweg managebar sind, ist ein automatischer Konsistenzcheck möglich. Die durchzuführenden Aktionen sind in vier Gruppen einzuteilen:

Neueintragung: Bisher nicht im Router installierte Interfacekarten werden erkannt und in Cinema eingetragen.

Ergänzung: Änderungen an schon installierten Interfacekarten, wie zum Beispiel die benutzte Anzahl der Ports oder ausgefallene Ports, werden erkannt und in Cinema geändert.

Ersetzen: Der Austausch von Interfacekarten wird erkannt und die bestehenden Einträge in Cinema durch die neuen Daten ersetzt.

Löschen: Aus dem Router entfernte Interfacekarten werden erkannt und aus Cinema ausgetragen.

Zusätzlich können die Vorgänge aufgezeichnet werden und somit ein Überblick über den zeitlichen Ablauf des Programmes gewonnen werden. Außerdem hat man zusätzlich die Möglichkeit, über die Auswertung des Log-Files eine Übersicht über die Konfigurationsänderungen an den Routern zu gewinnen. Man kann dann z.B. aus diesen Daten ersehen, bei welchen Komponenten besonders häufig Änderungen stattfinden.

Eine offene Frage stellt noch die Abbildung der Daten aus den MIBs auf Cinema dar. Dies gilt es anhand konkreter Beispiele zu klären. Auch gilt es noch zu klären inwieweit Daten aus Cinema **nicht** in den MIBs zu finden sind. Nach einer ersten Untersuchung der MIB der Cisco-Router ist es zwar möglich die einzelnen Interfaces über die Interface-Group der `mib2` abzufragen, es ist aber anscheinend nicht möglich den genauen Typ von Einschubkarten abzufragen. Außerdem muß eine übereinstimmende Zählweise der Interface in Cinema und der Interface-Group der `mib2` vereinbart werden, so daß diese Zuordnung eindeutig ist.

Ebenfalls ist noch die Frage offen, ob die Informationen über die NetView-Schnittstellen oder über andere Wege (z.B. `perl` mit snmp-package; Einbindung externer Programme, wie `snmpget`) gewonnen werden können.

Das zu erstellende Programm sollte wegen der Wichtigkeit der Daten als `cron`-Job täglich ablaufen. Eine nennenswerte Belastung des Netzes, Cinema und NetView ist aufgrund der geringen Datenmenge nicht zu erwarten.

6.3.5 Konsistenzcheck der IP-Adressen im Netz und in Cinema

Weite Teile des BMW-Netzes werden auf TCP/IP-Basis betrieben. Für die eindeutige Zuordnung der IP-Adressen ist Cinema die Vergabeinstanz. Doch Cinema bekommt keine Informationen über die wieder freigegebenen Adressen, da einerseits keine Rückmeldung über unbenutzte Adressen erfolgt und andererseits zwischen Cinema und dem Netz keine Kommunikation über die tatsächlich genutzten Adressen erfolgt.

Dies hat dazu geführt, daß trotz ca. 3,9 Milliarden theoretisch möglicher Adressen, in Teilbereichen keine neuen IP-Adressen vergeben werden können. Somit ergibt sich die Anforderung eine Möglichkeit zu schaffen, die einen Konsistenzabgleich zwischen Cinema und NetView, als Instanz der tatsächlich genutzten Adressen, durchführt.

Dabei muß eine Liste aller in Cinema dokumentierten Adressen erstellt werden, die dann mit der Liste aller in NetView vorkommenden Adressen abgeglichen wird. In NetView kann auch eingestellt werden, ab wann eine nicht mehr erreichbare Komponente aus der Datenbank gelöscht werden soll. Ein Abgleich mit dem bestehenden Netz findet ständig statt. Somit ist die Dokumentation der Komponenten in NetView immer auf aktuellem Stand.

Es können beim Vergleich der Listen drei Fälle auftreten, auf die jeweils verschieden reagiert werden muß:

- **Adresse nur in NetView:** Es wird eine in Cinema freie Adresse schon im Netz genutzt. Somit kann es sein, daß auch die damit verbundene Komponente nicht dokumentiert ist, oder aber bei der Konfiguration die IP-Adresse falsch eingegeben wurde.

  Als Folge dessen muß die Adresse als belegt markiert werden und der Ursache nachgegangen werden.

- **Adresse nur in Cinema:** Hier *kann* es sich um eine nicht mehr benutzte Adresse handeln. Es muß zusätzlich eine Überprüfung stattfinden, ob in NetView das betreffende Subnetz überhaupt mit in den Autodiscoveryprozeß aufgenommen wurde und ob die Komponente über den Gültigkeitszeitraum von Einträgen in NetView hinaus nicht erreichbar war. Zusätzlich könnte die Komponente auch unter einer anderen Adresse in Netz aktiv sein.

  Nach diesen Abfragen kann die Komponente gelöscht werden und die Adresse neu vergeben werden.

Bem.: Bei diesen beiden Fällen ist darauf zu achten, daß eine Komponente auch mit mehreren Adressen angesprochen werden kann. So kann eine Maschine sowohl an einem Token-Ring als auch an einem Ethernet angeschlossen sein. Dies ist bei den Abfragen zu beachten.

- **Adresse in Cinema und NetView:** Dies sollte der Normalfall sein. Die Adresse ist dokumentiert und wird im Netz genutzt.

  Die einzige Unbekannte in diesem Fall ist, ob die mit der Adresse in Cinema dokumentierten Komponente auch tatsächlich so konfiguriert ist. Es könnten Vertauschungen von Adressen vorkommen. Dieser Nebenaspekt spielt aber keine Rolle in der Diskussion, ob eine Adresse benutzt wird.

Die Auswertung kann zusätzlich zu den oben angesprochenen Punkten nach einem organisatorischen Aspekt sortiert werden. Über die Zuordnung von Subnetzen zu Abteilungen kann gezielt diese Abteilung überprüft werden. Dies ist dann nützlich, wenn dieser Abteilung keine neuen Adressen zugeteilt werden können.

Um das Programm möglichst einfach zu halten, können für die Behandlung der IP-Adreß-Problematik im ersten Ansatz nur Listen generiert werden, die die oben angeführten Probleme der eindeutigen Zuordnung noch nicht behandeln, sondern nur eine Gegenüberstellung der in Cinema und NetView enthaltenen Adressen sind.

Auch hier stellt sich wieder die Frage, ob die Abfragen an NetView auf der Ebene der C-Schnittstelle oder auf der unterliegenden Oracle-Datenbank stattfinden.

Der erste Weg ist zwar eleganter in Hinblick auf die Implementierung, dafür der langsamere. Der zweite Weg ist der schnellere, da nicht der „Umweg“ über NetView gegangen wird. Für den ersten Weg wurde schon ein einfacher Prototyp erstellt, der die Gegenüberstellung der Adressen ohne Behandlung der Nebenaspekte erfüllt (siehe Abschnitt 7.1).

Der Abgleich sollte zyklisch ausgeführt werden, da auch NetView ständig Komponenten löscht, die über ein einstellbares Zeitlimit (z.Z. 6 Wochen) hinaus nicht mehr erreicht werden konnten. Zusätzlich sollte eine Möglichkeit bestehen, bei Bedarf auf Anforderung die Listen generieren zu können.

Kapitel 7

Prototyp-Implementierung

7.1 Anzeigen von Cinema-Daten in NetView

Um ein genaueres Bild der Mächtigkeiten der Schnittstellen zu bekommen und um ihre Einsatzfähigkeit zu testen, wurden aus zwei konkreten Anforderungen heraus Prototypen angefertigt, die die gewüschte Funktionalität erfüllen.

In den Netzkomponenten wird die mib2–System–Gruppe nicht administriert. Diese Informationen stehen jedoch in Cinema. Um auch im Netzmanagemensystem diese Daten einfach verfügbar zu haben, wurden in diesem Beispiel Daten aus Cinema wie Hersteller, Typ, Seriennummer, System, Adresse und Standort in einer Übersicht im Netzmanagemensystem angezeigt.

Die Spezifikation war dabei folgende:

- Der Aufruf erfolgt über einen Eintrag im `Misc`-Menü
- Der Aufruf gliedert sich in ein weiteres Submenü ein, unter dem schon ein anderes Beispiel von IBM selbst eingehängt wurde. (`Misc -> Print Map (ASCII) -> Print OVwSelection`)
- Die Anwendung selbst besteht aus einem `xterm`-Aufruf, das seinerseits ein Shell-Skript mit dem ersten auf der Map selektierten Objekt als Übergabeparameter ausführt.
- Das Shell-Skript gibt als erstes den Übergabeparameter aus. Dann wird über SQL*PLUS die Verbindung zur Cinema-Datenbank hergestellt. Der verwendete Query steht in einer weiteren Datei. Die Ausgabe wird im Verzeichnis `/tmp` abgelegt. Dann wird die Ausgabe im xterm-Fenster angezeigt und nach einer beliebigen Eingabe das Fenster wieder geschlossen.

Um die Einfachheit zu verdeutlichen, sei hier der Quelltext mit eingebunden:

- Dies ist das ARF-File, in dem definiert ist, wie die Menüstruktur aufzubauen ist:

```
Application "OVw API Example: Print Map"
{
    Description {
        "OVw API ASCII Print Map"
    }

    MenuBar <100> "Misc" _s
    {
        <100> "Print Map (ASCII)"       _P   f.menu "Print Map (ASCII)";
    }

    Menu "Print Map (ASCII)"
    {
        <100> "Print Map"               _m f.action "Print Map";
        <100> "Print Map (verbose)"     _v f.action "Print Map (verbose)";
        <100> "Print Fields"            _f f.action "Print Fields";
        <100> "Print Fields (verbose)"  _x f.action "Print Fields (verbose)";
        <100> "Print OVwSelection"      _x f.action "Print OVwSelection";
    }

    Action "Print Map"
    {
        MinSelected 0;
        Command "/home/uni/juergen/printmap/printmap > /tmp/map.print 2>&1";
    }

    ... weitere Action-Sektionen wurden geloescht ...

    Action "Print OVwSelection"
    {
        Command
           "xterm -e /home/uni/juergen/printenv/OVwSel ${OVwSelection1} 2>&1";
    }
}
```

- Dies ist die Datei `/home/uni/juergen/printenv/OVwSel`, in der das Shell-Skript enthalten ist.

```
!/bin/sh
echo ´Parameter: ´$*
sqlplus $CINEMA_UID:$CINEMA_PASSWD \
            @/home/uni/juergen/printenv/OVwSel.sql_sel $* > /dev/null
cat /tmp/cinema.data
line
```

- Dies ist die SQL-Abfrage, die aus dem Shell-Skript unter dem Namen `/home/uni/juergen/printenv/OVwSel.sql` angesprochen wird.

```
set feedback off
set verify off
set termout off
spool /tmp/cinema.data
select hersteller,typ,seriennr,system,adresse,werk,gebaeude,etage,segment,raum
from nd_komp_adr_j where adresse like substr(upper('&1'),1,instr('&1','.')-1)
;
spool off
exit
```

7.2 Gegenüberstellung der IP-Adresse aus NetView und Cinema

Das zweite Beispiel dient der Erstellung einer Übersichtstabelle aller in Cinema und NetView enthaltenen IP-Adressen. Dies Anforderung entstand aus der Problematik, daß die Domain Name Server sich aus Cinema täglich die vermeintlich aktuellen IP-Listen laden, in Cinema aber nur IP-Adressen vergeben, bei Entfernung der Geräte aber nicht mehr ausgetragen wurden. Um nun Inkonsistenzen zwischen den in Cinema enthaltenen Adressen und den wirklich benützten zu erkennen, dienen die erstellten Programme und als Kontrollinstanz ein Shell-Skript. Der Programmcode ist im Anhang A aufgeführt.

Das Skript dient der Ablaufsteuerung und der eigentlichen Erzeugung der Tabelle aus den beiden Einzeltabellen. Zuerst werden die beiden Programme aufgerufen, die aus dem jeweiligen System die IP-Tabellen extrahieren und in eigene Dateien schreiben. Nachdem beide Programme beendet sind, werden die Ausgaben durch geschicktes Mischen und Sortieren in die gewünschte Form gebracht. Aus dem Ergebnis kann nun in einer Gegenüberstellung ersehen werden, ob die Adresse nur in einem oder in beiden Systemen enthalten ist. Der prinzipielle Ablauf ist in Abbildung 7.1 beschrieben.

Doch das gelieferte Ergebnis ist nur teilweise aussagekräftig. Da in NetView *nicht alle* Netze erfaßt sind, sind auch nicht alle Geräte in der NetView–Datenbank enthalten. Somit trifft das Ergebnis nur auf all die Subnetze zu, die auch in NetView zu sehen sind. Ein weiterer Nachteil ist die lange Laufzeit von circa einer halben Stunde, in der dem NetView–Rechner alle zur Verfügung stehende Ressourcen abverlangt werden. Selbst der Mauszeiger bewegt sich nicht mehr flüssig. Die Anfrage an die Cinema-Datenbank fällt mit keiner Mehrbelastung auf, da hier das System auf derartige Anfragen ausgelegt ist.

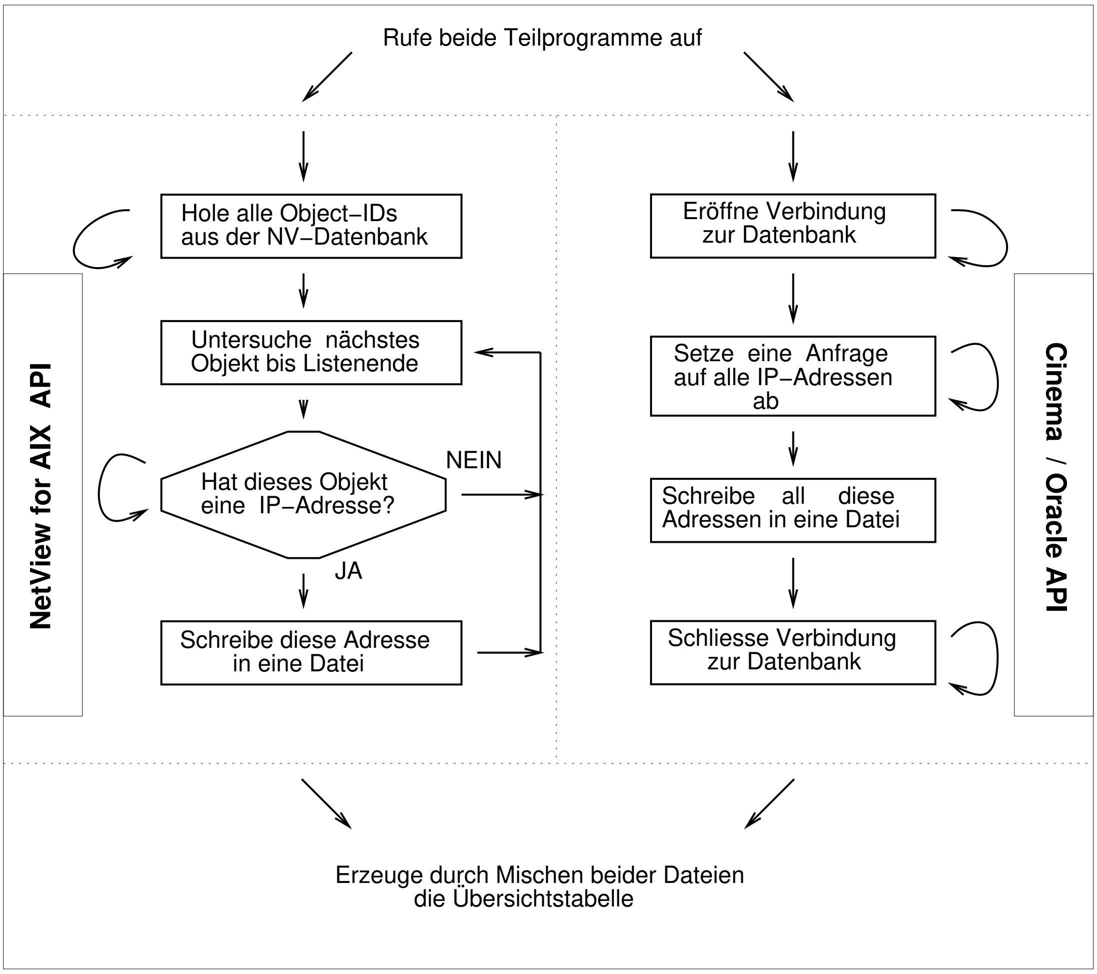

Abbildung 7.1: Ablaufdiagramm für die IP-Adreß-Problematik

Kapitel 8

Fazit und Ausblick

Die beiden Teile der Arbeit haben gezeigt, daß zwischen der Wirklichkeit und den bestehenden Anforderungen große Diskrepanz besteht.

Aus der Beschreibung der Defizite im zweiten Teil haben sich konkrete Implementierungsaspekte ableiten lassen, die es in Zukunft zu realisieren gilt, um damit die Qualität der bestehenden Werkzeuge zu erhöhen und somit eine Vereinfachung der Arbeitsabläufe zu erzielen.

Im Sinne des integrierten Managements sind Netzmanagementsystem und Netzdokumentation so tief zu verbinden, daß sie aus Benutzer- und Datensicht als ein integriertes System anzusehen sind. Ziel wird ein verteiltes, aber integriertes Management sein, das auf einer konsistenten gemeinsamen Datenbasis arbeitet. Die Methodik zur Realisierung eines solchen Datenbestandes liefert der erste Teil, der es mit seinem strukturierten Top-Down-Vorgehen dem Betreiber erlaubt, sich aus dem in manchen Bereichen herrschenden Überfluß an Daten die für seine Anforderungen relevanten herauszufiltern. Mit der Integration dieser beiden Systeme ist aber nur der erste Schritt getan: Weitere Systeme, die auch diesen Datenbestand als Grundlage benutzen, müssen in der Folge mit einbezogen werden. Dies können Trouble-Ticket-Systeme, Statistik-Systeme oder auch Accounting-Systeme sein.

Denn nur mit einem solchen konsistenten und umfassenden Datenbestand können dann globale Zusammenhänge erkannt werden und diese in den Arbeitsabläufen umgesetzt werden, wie z.B. automatische Fehlererkennung, -diagnose und -behebung oder aber automatische Fehlervermeidung und Früherkennung. Mit einer umfassenden Dokumentation der Konfiguration könnten sich dann neue Konfigurationen aus schon bestehenden ableiten lassen.

Doch ob dieses Stadium des Managements je erreicht wird, ist zur Zeit noch fraglich, da ein solches System von einem Hersteller oder Herstellerkonsortium verwirklicht werden müßte, die diesen immensen Aufwand finanziell zugunsten dieser Innovationen nicht scheuen und Spezialisten aus allen Bereichen der Informatik auf dieses Projekt vereinigen können. Es ist ebenso fraglich, ob ein solches integriertes Produkt gegen am Markt etablierte Insellösungen ausreichende Akzeptanz erzielen kann, um diesen Aufwand zu rechtfertigen. Die weitere Entwicklung wird Anworten auf diese Fragen geben.

Anhang A

Erstellte Programme

Im folgenden sind die einzelnen Programmteile aufgeführt, deren Gesamtablauf in 7.2 erläutert sind.

- `Check_IP_Adressen.reg`
  Über diese Registration-File in NetView ein Menüeintrag generiert, über den das Programm aufgerufen werden kann.
- `Check_IP_Adressen.skript`
  Dieses Skript ruft die beiden folgenden Programme auf und mischt deren Ergebnisse nach aufsteigenden IP-Adressen zusammen.
- `get_IP_Nv6k_fast.c`
  Diese Programm liest alle IP-Adressen über die C-API aus NetView aus.
- `get_IP_from_Cinema.pc`
  Dieses Plus-C-Programm nimmt Kontakt zur Cinema-Datenbank auf und liest alle IP-Adressen aus.
- `Makefile`
  Mit diesem Makefile wurden die lauffähigen Programm-Codes erstellt.

A.1 Check_IP_Adressen.reg

```
Application "Menu Item to Check IP Adresses in use or not used"
{
    Description {
    "Dieser Menuepunkt ruft ein Shellskript auf, das die IP-Adressen ",
    "aus der NetView-Datenbank und der Cinema-Datenbank holt,       ",
    "daraus dann eine Tabelle konstruiert, anhand der man dann      ",
    "feststellen kann, ob Inkonsistenzen in der Cinema-DB gegenueber",
```

```
    "dem realen Netz bestehen.                                        "
    }

    Version "1.0";

    Copyright {
      "(C) COPYRIGHT Juergen Prusseit, TUM, 1995 ",
      "    All Rights Reserved"
    }

    MenuBar <100> "Misc" _s
    {
        <100> "Check IP Adressen"       _P   f.menu "Check IP";
    }

    Menu "Check IP"
    {
        <100> "Check_IP_Adressen"              _g f.action "Check_IP_Adressen";
    }

    Action "Check_IP_Adressen"
    {
        Command "xterm -e /usr/bin/ksh
        /home/uni/juergen/Fertig/Check_IP_Adressen/Check_IP_Adressen.skript ";
    }
}
```

A.2 Check_IP_Adressen.skript

```
#!/usr/bin/ksh
#################################################################
#                                                               #
#  File: Check_IP_Adressen.skript                               #
#                                                               #
#  Steuere das Anlegen der Tabellen der IP-Adressen             #
#  aus Cinema und NetView und erzeuge dann daraus               #
#  die endgueltige Tabelle                                      #
#                                                               #
#  Juergen Prusseit, TUM , Maerz 1995                           #
#################################################################

# Fuers Debugging geben wir ab jetzt alle Zeilen aus
set -x

# Fuer ´get_IP_from_Cinema´ werden die beiden folgenden
# Variablen Definiert
CINEMA_UID=                     # gel"oscht, warum wohl ?
export CINEMA_UID
CINEMA_PWD=                     # gel"oscht, warum wohl ?
```

```
export CINEMA_PWD

# In die beiden folgendend Variablen werden die beiden
# Adress-Tabellen geschrieben
CINEMA_INPUT=/tmp/CINEMA_IP_Adressen
NETVIEW_INPUT=/tmp/NetView_IP_Adressen

# In den Tabellen sind die Adressen durch folgende
# Kennwoerter markiert
KENNWORT_CINEMA='CINEMA'
KENNWORT_NETVIEW='NetView'

# Hierhin wird das Gesamtergebnis geschieben
OUTPUT='/tmp/TABELLE_IP_NETVIEW_CINEMA'

# Hier stehen die Programmme
cd /home/uni/juergen/Fertig/Check_IP_Adressen

# Hole alle Adressen aus NetView for Aix
./get_IP_Nv6k_fast     >$NETVIEW_INPUT    &
# Hole alle Adressen aus Cinema
./get_IP_from_Cinema    >$CINEMA_INPUT    &

# Warte darauf, dass die beiden Programme beendet sind,
# bevor die eigentliche Tabellengenerierung beginnt
wait %1
wait %2

# Das folgende ist etwas umstaendlich, erspart aber einige Programmier-
# aufwand. (Warum soll man alles mehrfach programmieren ?)

# Join erwartet sortierte Dateien
sort $CINEMA_INPUT                            >/tmp/tmp_cine
sort $NETVIEW_INPUT                           >/tmp/tmp_nv6k

# join'e so, dass alle in  [ Cinema und Netview ]  und alle nur in [ Cinema ]
# enthaltenen Adressen in '/tmp/tmp_join' enthalten sind
join -a1 /tmp/tmp_cine /tmp/tmp_nv6k                    >/tmp/tmp_join

# join'e so, dass alle in  [ Cinema und Netview ]  und alle nur in [ NetView ]
# enthaltenen Adressen  an '/tmp/tmp_join' angehaengt werden.
join -a2 /tmp/tmp_cine /tmp/tmp_nv6k                    >>/tmp/tmp_join

# Sortiere aufsteigend nach der IP-Adresse, die durch den Separator '.'
# getrennt ist. (Wie nicht anders zu erwarten.
# Da alle in [ Cinema und Netview ] enthaltenen Adressen doppelt enthalten sind
# eliminiere diese mit uniq
sort -t\. -1 -n -2 -n -3 -n -4 -n /tmp/tmp_join | uniq   >/tmp/tmp_uniq

# Die Tabelle ist eigentlich fertig, allerdings sieht sie noch nicht sehr
# strukturiert aus. Also fuehre ich ein paar Tabulatoren ein, und sie sieht
```

```
# Perfekt aus :-))
sed -e "s/$KENNWORT_CINEMA $KENNWORT_NETVIEW/tmpstr/" \
                                                 /tmp/tmp_uniq     >/tmp/tmp_sed1
sed -e "s/NetView/    NetView/"                  /tmp/tmp_sed1    >/tmp/tmp_sed2
sed -e "s/CINEMA/            CINEMA/"            /tmp/tmp_sed2    >/tmp/tmp_sed3
sed -e "s/tmpstr/    NetView        CINEMA/"     /tmp/tmp_sed3    >$OUTPUT
```

A.3 get_IP_Nv6k_fast.c

```
/*
 ****************************************************************
 *                                                              *
 * File: get_IP_Nv6k_fast.c                                     *
 *                                                              *
 ****************************************************************
 *
 * Suche aus der Datenbank alle ObjectID´s
 * dann hole die IP-Adresse fuer diese ID,
 * wenn diese ungleich NULL ist, gebe sie
 * aus.
 *
 * Juergen Prusseit, TUM, Maerz 1995
 *
 */

#include <stdio.h>
#include <stdlib.h>

#include <OV/ovw.h>
#include <ctype.h>

void main()
{
    // dummy return wert
    int ret;

    // Zaehler
    int i,j,k,l ;

    // Hier wird die IP-Adresse eingehaengt
    char *ip_adr;

    // Hier stehen alle Object-ID´s drin
    OVwObjectIdList *OIDliste;

    // in f_id_ip steht fuer spaeter die FieldId der IP-Adresse
    OVwFieldId    f_id_ip;

    // Initialisiere OVw Zugriffe
```

```
    ret = OVwInit();
    if (ret != 0)
        { printf("Fehler: %s\n",OVwErrorMsg(OVwError())) ; exit(1); }

    // Initialisiere den Datenbankzugriff
    ret = OVwDbInit();
    if (ret != 0)
        { printf("Fehler: %s\n",OVwErrorMsg(OVwError())) ; exit(1); }

    // hole FieldId der IP-Adresse
    f_id_ip=OVwDbFieldNameToFieldId( "IP Address" );

    // Mit NULL als Uebergabe bekomme ich ALLE Object-ID´s
    OIDliste = OVwDbListObjectsByFieldValue( NULL );

    // Wenn die Liste nicht leer ist ...
    if (OIDliste != NULL )
    {
        // ... durchlaufe diese bis zum
        //  (OIDliste->count)-1 ...
        for (i=0 ; i < OIDliste->count ; i++ )
            {
                //  ... und hole die IP-Adresse ...
                ip_adr =
                     OVwDbGetFieldStringValue(OIDliste->object_ids[i],f_id_ip);

                // ... wenn sie vorhanden ist, gebe sie aus,
                // sonst gehe zum naechsten Feld
                if (ip_adr != NULL)
                    printf("%s\t\tNetView\n", ip_adr);

            }
    }
    else
    {
        printf("OIDliste leer!!\n");
        exit(1):
    }

}
```

A.4 get_IP_from_Cinema.pc

```
/*
 ***************************************************************
 *                                                             *
 * File: get_IP_from_Cinema.pc                                 *
 *                                                             *
 ***************************************************************
```

```
 *
 * Lese alle IP-Adressen aus Cinema und gebe sie aus
 *
 * Juergen Prusseit, TUM, Fr 28 Maerz 1995
 *
 */
#include <stdio.h>
#include <stdlib.h>

#define    SET_LEN(VAR)    (VAR.arr[VAR.len]='\0')
#define    GET_LEN(VAR)    (VAR.len=strlen(VAR.arr))

/***************************************************************************
Variablendefinitionen fuer Oracle.
Bei den mit 10000 vordefinierten Variablen werden 10000 DB-Saetze auf einmal
gelesen und anschliessen weiterverarbeitet.
***************************************************************************/

EXEC SQL BEGIN DECLARE SECTION;
    varchar    ip_adresse[10000][15];
    varchar    uid[100];
    varchar    pwd[100];
EXEC SQL END DECLARE SECTION;

EXEC SQL INCLUDE SQLCA;

main( argc, argv, envp)
int    argc;
char    *argv[];
char    *envp[];
{
        int i,num_ret;

/***************************************************************************
Der Datenbankconnect wird mit den Environment-Variablen CINEMA_UID und
CINEMA_PWD definiert.
***************************************************************************/

    strcpy (uid.arr, getenv("CINEMA_UID"));
    GET_LEN(uid);

    strcpy (pwd.arr, getenv("CINEMA_PWD"));
    GET_LEN(pwd);

    fprintf(stderr,"\nDownloading from DB: argc=%d\n", argc);
    fprintf(stderr,"Other:CINEMA_UID=%s -%d\n", uid.arr, uid.len);
    fprintf(stderr,"Other:CINEMA_PWD=%s -%d\n", pwd.arr, pwd.len);
    fflush(stderr);

        fprintf(stderr,"Trying to connect...");
```

```
        EXEC SQL WHENEVER SQLERROR GOTO errprint;
        EXEC SQL WHENEVER NOT FOUND CONTINUE;

    /* Connect to ORACLE */
        EXEC SQL CONNECT :uid IDENTIFIED BY :pwd ;
        fprintf(stderr, " connected.\n");

/***************************************************************************
Definieren des ersten SELECT-Statements
****************************************************************************/

    EXEC SQL DECLARE C1 CURSOR FOR
        select     ip_adresse
        from    dns_tcpip
        ;

    EXEC SQL OPEN C1;
    fprintf(stderr, " cursor C1 opened.\n");

/***************************************************************************
Definieren des Fehler-Exits. Der Exit wird angesprungen, wenn ein DB-Fehler
auftritt, oder wenn kein DB-Satz mehr vorhanden ist.
****************************************************************************/

    EXEC SQL WHENEVER NOT FOUND GOTO not_found_1;

        num_ret = 0;

    for (;;) {
/***************************************************************************
Lesen von jeweils 10000 DB-Saetzen in die lokalen Variablen, zuweisen der
aktuellen Laengen mit SET_LEN und ausgabe in ein File mit printf. In num_ret
steht die Anzahl der bereits gelesenen Adressen.
****************************************************************************/

        EXEC SQL FETCH C1 INTO :ip_adresse;

        fprintf(stderr,"sqlcode=%d(%60s)\n",
                sqlca.sqlcode,sqlca.sqlerrm.sqlerrmc);

        if ( sqlca.sqlcode != 0 ) break;

                for (i=0; i < (sqlca.sqlerrd[2] - num_ret); i++) {
          SET_LEN(ip_adresse[i]);
          printf("%-15s CINEMA\n",
            ip_adresse[i].arr
            );
        }
        num_ret = sqlca.sqlerrd[2];
    }
```

```
/*************************************************************************
Nachdem keine Datensaetze mehr vorhanden bzw. ein Oracle-Fehler aufgetreten
ist, wird hier weitergemacht. Lese alle noch nicht bearbeiteten Saetze in
die lokalen Variablen ein und schreibe diese in das File.
*************************************************************************/

not_found_1:
        if ((sqlca.sqlerrd[2] - num_ret) > 0)
                for (i=0; i < (sqlca.sqlerrd[2] - num_ret); i++) {
          SET_LEN(ip_adresse[i]);
          printf("%-15s CINEMA\n",
            ip_adresse[i].arr
            );
        }

    EXEC SQL CLOSE C1;

/*************************************************************************
Oracle-Connection beenden und Programm verlassen.
*************************************************************************/

    EXEC SQL COMMIT WORK RELEASE;
    exit(0);

errprint:
    /* We end up here if an error occurs */
    EXEC SQL WHENEVER SQLERROR CONTINUE;
    fprintf(stderr, "\n\n>>>>> Error during execution:\n");
    /* Print ORACLE error message and log off the database */
    fprintf(stderr, "%s\n",sqlca.sqlerrm.sqlerrmc);
    EXEC SQL ROLLBACK RELEASE;
    exit(1);
}
```

A.5 Makefile

```
################################################################
#                                                              #
#  File: Makefile                                              #
#                                                              #
#  Fuer die Compilierung der beiden Programme                  #
#              get_IP_from_Cinema                              #
#              get_IP_Nv6k_fast                                #
#                                                              #
#  Juergen Prusseit, TUM, Maerz 1995                           #
################################################################
CC = /usr/bin/xlc
CFLAGS=-g -qDBXEXTRA -qCPLUSCMT -qEXTCHK
LIBS=-lovw -lov -lntl
```

```
all:    get_IP_from_Cinema.c get_IP_from_Cinema get_IP_from_Nv6k


get_IP_from_Cinema.c: get_IP_from_Cinema.pc
    /usr/lpp/oracle/ora7016/bin/proc include=/usr/oracle/c/lib \
    ireclen=255 oreclen=255 iname=get_IP_from_Cinema.pc

get_IP_from_Cinema: get_IP_from_Cinema.o
    $(CC) $(CFLAGS) get_IP_from_Cinema.o  \
    /usr/lpp/oracle/ora7016/lib/libsql.a \
    /usr/lpp/oracle/ora7016/lib/libora.a \
    `cat /usr/lpp/oracle/ora7016/rdbms/lib/sysliblist` \
    /usr/lpp/oracle/ora7016/lib/osntab.o \
    /usr/lpp/oracle/ora7016/lib/libsqlnet.a \
    /usr/lpp/oracle/ora7016/lib/libcv6.a \
    /usr/lpp/oracle/ora7016/lib/libnlsrtl.a \
    /usr/lpp/oracle/ora7016/lib/libcore.a \
    -o get_IP_from_Cinema

get_IP_from_Nv6k: get_IP_Nv6k_fast.o
    $(CC) $(CFLAGS) get_IP_Nv6k_fast.o -o get_IP_Nv6k_fast $(LIBS)
```

Anhang B

Literaturverzeichnis

[Abec94] Abeck, S.: Integrationstechniken im Netzmanagement, TU München - Institut für Informatik, 1994

[ANPR94] Anastos, T.; Prusseit, J.: Management eines privaten X.25-Netzes auf der Basis der Managementplattform Spectrum, Fortgeschrittenenpraktikum, TU München - Institut für Informatik, Juni 1994

[ANS75] ANSI/X3/Sparc Study Group on Data Base Management Systems. Interim Report 75-02-08. FDT (Bulletin of ACM-SIGMOD) 7 (1975), Nr. 2

[Bert95] Bertram, D.: Analyse bestehender Verfahren des Problemmanagements im Hinblick auf deren Unterstützung durch ein Trouble Ticket System, Diplomarbeit, TU-München - Institut für Informatik, Februar 1995

[Cine94] BMW: Benutzerdokumentation CINEMA, BMW AG, München 1994

[HAWI94] Hegering, H.-G.; Abeck, S.; Wies, R.: Betriebskonzepte für ein betreibergerechtes integriertes Management von Informationsverarbeitungs-Ressourcen, 1995

[HeAb93] Hegering, H.-G.; Abeck, S.: Integriertes Netz- und Systemmanagement, Addison-Wesley, 1.Auflage, 1993

[ISO 9596] International Organisation for Standardization: Information Technology - Open Systems Interconnections (OSI) – Common Management Information Protocol Specification (CMIP), International standard 9596, 1991

[Kand93] Kandizia, P.: Theoretische Grundlagen relationaler Datenbanksysteme, BI-Wiss.-Verl., 1993

[NvAd94] IBM, NetView for AIX: Administrator's Guide Version 3, IBM SC31-7192-00, 1994

[RFC1213] McCloghrie, K.; Rose, M.T.: Management Information Base for network management of TCP/IP-based internets: MIB-II, 1991

[Rose94] Rose, M. T.: The Simple Book, Prentice Hall, 2nd ed, 1994

[Schla83] Schlageter, G.: Datenbanksysteme: Konzepte und Modelle, Teubner, 2. Auflage, 1983

[Weiß94] Weiß, H.-P.: Erstellung eines Konzepts zur Datenintegration im Netzmanagement und dessen Anwendung auf eine konkrete Netzbetreiber-Organisation. Diplomarbeit, TU-München - Institut für Informatik, November 1994

[Wied93] Wiedemann, K.: Integration eines Netz- und Fehlerdokumentationssystems in eine Managementplattform als Basis für ein Enterprise-Management. Diplomarbeit, TU-München - Institut für Informatik, August 1993

Abbildungsverzeichnis