Konzeption einer Service-MIB – Analyse und Spezifikation dienstorientierter Managementinformation

Dissertation

an der

Fakultät für Mathematik, Informatik und Statistik
der
Ludwig-Maximilians-Universität München

vorgelegt von

Martin Sailer

Tag der Einreichung: 06. Juli 2007
Tag der mündlichen Prüfung: 26. Juli 2007

1. Berichterstatter:	**Prof. Dr. Heinz-Gerd Hegering**, Ludwig-Maximilians-Universität München
2. Berichterstatter:	**Prof. Dr. Gabrijela Dreo Rodosek**, Universität der Bundeswehr München

Das Ganze ist mehr als die Summe seiner Teile.
Aristoteles

Danksagung

Eine Dissertation geht zumeist aus einem Reifungsprozess hervor, bei dem verschiedene Ideen und Teillösungen ineinander übergehen und schließlich zu einem *großem Ganzen* verschmelzen. Dies trifft auch auf die vorliegende, im Rahmen meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl von Prof. Dr. Heinz-Gerd Hegering entstandene, Arbeit zu und deshalb möchte ich an dieser Stelle den Menschen danken, die diesen Prozess unterstützt und somit nachhaltig zum Gelingen meiner Dissertation beigetragen haben.

Mein besonderer Dank gilt meinem Doktorvater, Prof. Dr. Heinz-Gerd Hegering, der diese Dissertation von den ersten Ideen an begleitet und hervorragend betreut hat. Sehr herzlich möchte ich mich auch bei meiner Zweitgutachterin, Prof. Dr. Gabrijela Dreo Rodosek, für ihre konstruktiven Anmerkungen bedanken, die ebenfalls Einzug in diese Arbeit gefunden haben. Bei beiden Gutachtern möchte ich mich ausdrücklich für ihr persönliches Engagement und die unglaublich zügige Korrektur von Zwischenversionen bedanken.

Ferner möchte ich allen Kolleginnen und Kollegen des MNM-Teams danken. Zahlreiche Diskussionen und gemeinsam verfasste Artikel sind in die vorliegende Arbeit eingeflossen und haben zur Strukturierung meiner Ideen nachhaltig beigetragen. Darüberhinaus habe ich über die Jahre ein Arbeitsumfeld vorgefunden, das maßgeblich das Gelingen meiner Dissertation unterstützt hat.

Nicht zuletzt danke ich meinen Eltern, die mir durch ihre bedingungslose Unterstützung Zeit meines Lebens alle Wege geebnet haben. Besonderer Dank gilt Nadine; ohne ihren besonderen Rückhalt wäre diese Arbeit nicht möglich gewesen.

München, im Juli 2007

This work was supported in part by the EC IST-EMANICS Network of Excellence (#26854).

Zusammenfassung

In den letzten Jahren ließ sich ein starker Trend hin zum dienstorientierten Management verzeichnen. Betreiber von IT-Diensten gehen verstärkt dazu über, den Betrieb ihrer Infrastrukturen an technischen und organisatorischen Dienstmanagementkonzepten auszurichten und versuchen dadurch, den durch gesteigerte Kundenbedürfnisse und komplexere Diensterbringungsszenarien induzierten Rahmenbedingen zu begegnen. Mit der Einführung neuer Dienstmanagementsysteme wächst allerdings gleichzeitig der Bedarf nach Interoperabilität: Hierbei stellt insbesondere das Vorhandensein einer standardisierten Managementinformationsbasis (MIB) die entscheidende Prämisse für einen anwendungsübergreifenden Austausch bzw. eine Wiederbenutzung von Managementinformationen dar.

Um eine Integration des Dienstmanagements zu unterstützen, wird deshalb in dieser Arbeit eine dienstorientierte Informationsbasis (*Service-MIB*) konzipiert. Die Grundlage dafür bildet eine vierstufige Methodik: Zunächst wird innerhalb der *Analysephase* der Bedarf an dienstorientierter Managementinformation anhand mehrerer Gesichtspunkte ermittelt und somit der Frage nachgegangen, welche Informationen zur Erfüllung von Dienstmanagementaufgaben effektiv benötigt werden. Daran anschließend widmet sich die *Spezifikationsphase* der Modellierung und adäquaten Beschreibung von Dienstmanagementinformationen. Hierbei werden die vorab ermittelten Entitäten, Attribute und Beziehungen in objektorientierte Modelle überführt und mit Hilfe einer, in dieser Phase entwickelten, deklarativen Spezifikationssprache *(SISL)* formalisiert. Ein entscheidendes Kriterium für eine *Service-MIB* stellt ferner ihre Aktualität dar, d.h. in ihr enthaltene Informationen müssen den aktuellen Zustand des Dienstes reflektieren. Dies wird innerhalb der *Überwachungsphase* durch Einführung einer geeigneten Dienstüberwachung adressiert: Basierend auf *SISL*-Spezifikationen werden komponentenorientierte Managementinformation mit Hilfe eines Überwachungswerkzeugs (*SMONA*) aggregiert und zu Dienstmanagementinformationen verdichtet. Abschließend beschäftigt sich die *Nutzungsphase* mit Möglichkeiten zur Einbettung in bestehende Managementarchitekturen.

Abstract

Recently, there has been a strong trend towards IT Service Management. To cope with the complexity involved in IT provisioning and meet customers' demands, IT-Providers are increasingly adopting technical und organizational Service Management concepts. However, the introduction of new service management systems also presents a challenge in terms of integrated management. Facilitating the exchange of management information is widely been considered as the key to interoperability among different systems – an issue which has traditionally been approached by an Management Information Base (MIB). Despite the large of number of MIBs describing networks and systems, an adaption of this concept to service management is still missing.

To further integration efforts in IT Service Management, a management information base for service management (*Service-MIB*) is developed in this theses. It builds on a methodology consisting of four phases: Firstly, information requirements for characterizing a service are derived by analysing typical scenarios as well as service management processes (*derive phase*). Based on these requirements, services are represented as Managed Objects (MOs) and mapped to an object-oriented model (*specification phase*). This includes the devolopment of *SISL*, a language suitable for the representation of service attributes in dependence of management attributes of the underlying infrastructure.

Since a *Service-MIB* needs to reflect the actual state of a service, integration into monitoring systems needs to be facilitated (*monitoring phase*). This is addressed by an service monitoring architecture (*SMONA*) capable of aggregating data produced by existing tools according to the instructions given in the *define phase*. Finally, guidelines on scenario-specific adaption of the *Service-MIB* are presented *(usage phase*).

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Herausforderungen in der Informationsmodellierung . . 3
1.2 Untersuchte Fragestellungen 5
1.3 Vorgehensmodell dieser Arbeit 7

2 Anforderungsanalyse **11**

2.1 Informationsmodellierung im Management 11
2.1.1 Managementobjekte und MIBs 12
2.1.2 Modellierung von Managementinformation 13
2.1.3 Definition konkreter Managementinformation . . 16
2.2 Analyse von Managementszenarien 18
2.2.1 Web-Hosting Szenario 18
2.2.2 GRID Computing Szenario 23
2.3 Entwicklung des Anforderungskatalogs 27
2.3.1 Allgemeine Anforderungen 27
2.3.2 Modelleigenschaften 28
2.3.3 Definition konkreter Dienstmanagementinformation 29
2.3.4 Umsetzbarkeitsaspekte 33
2.3.5 Zusammenfassung und Vorstellung des Anforderungskatalogs 35

3 Analyse bestehender Ansätze **37**
3.1 Arbeiten von Standardisierungsgremien 37
3.1.1 Distributed Management Task Force 38
3.1.2 TeleManagement Forum 42
3.1.3 Internet Engineering Task Force 49
3.1.4 IT Infrastructure Library 53
3.2 Forschungsarbeiten 56
3.2.1 Modellierung und Beschreibung von Diensten . . 57
3.2.2 Abhängigkeitsmodellierung 63
3.2.3 Abbildung von Managementinformation 67
3.3 Beispiele kommerzieller Produkte 70
3.3.1 HP OpenView Service Quality Manager 70
3.3.2 Infovista 72
3.4 Gesamtbewertung bestehender Arbeiten 74

4 Lösungsansatz und weiteres Vorgehen **79**
4.1 Die Service-MIB Idee 79
4.2 Methodik zur Konzeption einer Service-MIB 83
4.2.1 Analysephase 85
4.2.2 Spezifikationsphase 85
4.2.3 Überwachungsphase 88
4.2.4 Nutzungsphase 90
4.3 Zusammenfassung und weiteres Vorgehen 90

5 Informationsanalyse dienstbezogener Managementinformation **93**
5.1 Vorgehen in der Informationsanalyse 94
5.2 Informationsanalyse von Dienstorientierungsaspekten . 96
5.2.1 Grundlegende Bestandteile eines Dienstes 97
5.2.2 Managementaufgaben entlang des Dienstlebenszyklus 105
5.2.3 Berücksichtigung von Organisationsgrenzen . . . 113
5.2.4 Zusammenfassung der Informationsanforderungen aus Dienstorientierungsaspekten 114
5.3 Ableitung des Informationsbedarfs aus Managementprozessen 119
5.3.1 Relevante Teile von Prozessbeschreibungen . . . 121
5.3.2 Incident-Management 125
5.3.3 Service-Level Management 136

5.3.4 Konkretisierung der Informationsanforderungen aus Managementprozessen . . . 142
5.4 Zusammenfassung . . . 147

6 Spezifikation dienstbezogener Managementinformation 151
6.1 Vorgehen in der Spezifikationsphase . . . 152
6.2 Definition grundlegender Entitäten . . . 154
6.2.1 Beschreibungsschema für Entitäten . . . 156
6.2.2 Die Wurzelentität ManagedServiceElement . . . 157
6.2.3 Dienst und Dienstspezifikation . . . 158
6.2.4 Service Access Point . . . 163
6.2.5 Quality-of-Service (QoS) . . . 167
6.2.6 Service Level Agreement . . . 173
6.2.7 Dienstnutzer und Dienstkunden . . . 177
6.2.8 Resource . . . 180
6.2.9 Abhängigkeitsbeziehungen zwischen Diensten und Ressourcen . . . 182
6.2.10 Basismodell der Service-MIB . . . 186
6.3 Erweiterung des Basismodells mit Dienstattributen . . . 188
6.3.1 Kategorien von Dienstattributen . . . 188
6.3.2 Beschreibungsschema für Dienstattribute . . . 191
6.3.3 Attribute in der Kategorie Service Fault . . . 192
6.3.4 Attribute in der Kategorie ServicePerformance . . . 199
6.4 Entwicklung einer Spezifikationssprache . . . 207
6.4.1 Grundlegende Sprachkonzepte . . . 210
6.4.2 Ausdrücke zur Referenzierung von Komponentenparametern . . . 211
6.4.3 Abbildungsvorschriften . . . 212
6.4.4 Benachrichtigungsoptionen . . . 213
6.5 Exemplarische Anwendung . . . 214
6.6 Zusammenfassung . . . 222

7 Überwachung und Benutzung von Managementinformationen 227
7.1 Werkzeugunterstützung für die Überwachungsphase . . . 228
7.1.1 Die Service Monitoring Architecture (SMONA) . . . 230
7.1.2 Anwendungsbeispiel für die Informationsverfeinerung . . . 235
7.2 Benutzungsphase . . . 237

7.2.1 Integration der Service-MIB in CIM/WBEM . . 238
7.2.2 Szenariospezifische Anpassung der Service-MIB . 242
7.3 Zusammenfassung und Gesamtbewertung des Ansatzes . 245

8 Zusammenfassung der Ergebnisse und Ausblick 249

A Entitäten 257
A.1 Wurzelentiät 257
A.2 Dienst und Dienstspezifikation 258
A.3 Service Access Point 263
A.4 Quality of Service 267
A.5 Service Level Agreement 274
A.6 Dienstnutzer und -kunden 277
A.7 Resource 280
A.8 Abhängigkeitsbeziehungen 281

B SISL XML-Syntax 285

Abkürzungsverzeichnis 291

Abbildungsverzeichnis 295

Tabellenverzeichnis 298

Literaturverzeichnis 299

Index 315

Kapitel 1

Einleitung

In den letzten Jahren lässt sich eine ständige Zunahme der Anforderungen, die an Betreiber von IT-Diensten gestellt werden, beobachten. Als Ursachen dafür sind eine veränderte Erwartungshaltung von Seiten der Kunden, gepaart mit neuen, komplexeren Diensterbringungsszenarien anzusehen. Während Erstere im Zuge einer veränderten Wettbewerbssituation IT-Dienste zu immer geringeren Preisen, aber gleichzeitig festgelegter Qualität nachfragen, erfordern organisationsübergreifend realisierte und dynamisch komponierte IT-Dienste, wie sie im Rahmen von *Dynamic Service Provisioning* [DRL01] und GRID Computing [FK04] auftreten, ein gesteigertes Maß an Flexibilität und Adaptitvität.

Trend zu IT-Business-alignment

Weiterhin wird von IT-Abteilungen zunehmend erwartet, dass sie ihr Handeln an den Geschäftszielen des Unternehmens ausrichten [Gar04a]. Dieser, unter *IT-Business-alignment* subsummierte Trend verfolgt das Ziel, das symbiotische Zusammenspiel zwischen Geschäftsstrategie und Geschäftsstrukturen auf der einen Seite sowie IT-Strategie und deren Umsetzung auf der anderen zu verbessern. Für Betreiber von IT-Diensten erwächst daraus vor allem die Notwendigkeit, sich mit der Verzahnung der angebotenen Dienste mit dem Unternehmensgeschäft auseinanderzusetzen und Fragen nach beispielsweise der Auswirkung

von Ausfällen eines Dienstes auf vertragliche Vereinbarungen (*Service Level Agreement*) oder den Konsequenzen von Veränderungen in der Bereitstellung des Dienstes beantworten zu können.

Dienstmanagement gewinnt an Bedeutung

Als Konsequenz dieser Entwicklungen gewinnt das dienstorientierte Management an Bedeutung. Die charakteristische Eigenschaft dieses Paradigmenwechsels fußt darin, dass – im Gegensatz zur isolierten Betrachtung von Einzelkomponenten im komponentenorientierten Management – der Betrieb und die Bereitstellung von IT-Diensten ganzheitlich betrachtet wird. Ziel dabei ist es, eine *Dienstsicht* zu etablieren, die es Dienstbetreibern ermöglicht, ihre vorliegende IT-Infrastruktur im Kontext der damit erbrachten Dienste aufzufassen und im Einklang mit vertraglichen Vereinbarungen zu betreiben.

Managementinformationsbasis wird benötigt

Notwendige Voraussetzung zur Beantwortung von Dienstmanagementfragestellungen ist u.a. das Wissen um die an der Erbringung des Dienstes beteiligten Ressourcen und die dabei auftretenden Abhängigkeiten zwischen diesen [DR02] oder von Teildiensten. Derartige Informationen sind Teil einer *Managementinformationsbasis* (MIB), der Gesamtmenge der von einem offenen System nach außen zur Verfügung gestellten Managementinformationen. Die syntaktische und semantische Beschreibung der in einer MIB vorhandenen Informationen wird dabei durch ein Informationsmodell festgelegt und damit der Austausch von Managementinformationen zwischen verschiedenen Managementanwendungen ermöglicht. Allerdings wurde bisher für das Dienstmanagement kein gemeinsames Verständnis in Bezug auf die auszutauschenden Managementinformationen etabliert, was die Interoperabilität von Dienstmanagementwerkzeugen erheblich erschwert. Als Konsequenz wird von jeder Dienstmanagementanwendung eine eigene Dienstsicht hergestellt, gepflegt und verwaltet. Dabei entstehende Informationen weisen allerdings zumeist einen geringen Formalisierungsgrad aus, sind überdies in propietären Datenformaten gehalten und damit im Allgemeinen nicht wiederverwendbar. Ferner besteht in den seltensten Fällen ein Bezug zum Kunden bzw. zum Dienstleistungsvertrag.

Aus diesem Grund soll die vorliegende Arbeit einen Beitrag zur Definition adäquater Dienstmanagementinformation leisten und damit die Integration des Dienstmanagements unterstützen.

1.1 Neue Herausforderungen in der Modellierung von Managementinformationen

Standardisierung im komponentenorientierten Management

Die Konzeption und Realisierung einer standardisierten und damit semantisch und syntaktisch festgelegten Managementinformationsbasis wurde in anderen Managementdisziplinen bereits etabliert. So wurde in den letzten 30 Jahren eine Reihe von Ansätzen für das Netz-, System- und Anwendungsmanagement im Rahmen von internationalen Standardisierungsarbeiten vorgestellt. Als prominente Vertreter gelten das von der Internet Engineering Task Force zusammen mit dem Internet Activity Board entwickelte Internet-Informationsmodell [RM90], das von der International Organization for Standardization vorgestellte ISO-Informationsmodell [ISO] oder neuere Ansätze wie das Common Information Model (CIM) [CIM06] und das vom TeleManagement Forum vorgestellte Shared Information Model (SID) [GB904a]. Insbesondere für das Internet- sowie ISO-Informationsmodell wurden eine Vielzahl von konkreten Managementinformationen in Form sogenannter Objektkataloge definiert.

Im Hinblick auf die Fülle existierender Arbeiten soll im Folgenden darauf eingegangen werden, welche neuen Herausforderungen aus der zunehmenden Dienstorientierung erwachsen und warum bisherige Ansätze diese nur partiell erfüllen können:

Festgelegte Dienstgüte wird notwendig

Berücksichtigung der geschäftlichen Perspektive Während in der Vergangenheit IT-Leistungen oftmals nach dem Best-Effort-Prinzip bereitgestellt wurden, wird zunehmend die Qualität des erbrachten Dienstes als wichtiges Differenzierungsmerkmal zwischen verschiedenen Providern angesehen. Aus Kundensicht scheint diese Entwicklung verständlich: Sollen für die Unternehmung geschäftskritische IT-Dienstleistungen an Dritte ausgelagert werden, erscheint eine festgelegte, garantierte Dienstgüte und deren vertragliche Absicherung in einer Dienstvereinbarung unerlässlich. Für eine Dienstmanagementinformationsbasis resultiert daraus vor allem die Notwendigkeit, entsprechende Konzepte (QoS, SLA) abzubilden und mit dem Dienst zu verknüpfen, was in bisherigen Managementmodellen nicht gegeben war.

Ausrichtung am Informationsbedarf von Dienstbetreibern Im Hinblick auf eine Verflechtung des IT-Managements mit den Geschäftszielen einer Unternehmung ist es weiterhin zwingend notwendig, die Festlegung von Dienstmanagementinformationen an der Frage auszurichten, welche Information von den Dienstbetreibern effektiv benötigt wird. Dieses Vorgehen steht im Gegensatz zu der Praxis im komponentorientierten Management, in der primär Managementinformationen definiert wurden, die sich durch einfache Instrumentierung von Komponenten gewinnen ließ [HAN99].

Abhängigkeiten zwischen Diensten und Komponenten Dienste werden üblicherweise durch eine Reihe von Netz- und Applikationskomponenten (*Dienstelemente*) realisiert und nutzen weiterhin Funktionalitäten von Subdiensten. Daraus resultierende Abhängigkeiten müssen in einer Dienstmanagementinformationsbasis abgebildet und beispielsweise durch ihren Typus (funktional, organisatorisch), ihre Ausprägung (zwischen Diensten oder zwischen Diensten und Komponenten), ihrer Kritikalität oder Lebensdauer charakterisiert werden. Dies wird insbesondere notwendig, um Kausalitäten in der Diensterbringungskette nachzuvollziehen: Gilt es beispielsweise, die Ursache für Störungen des Dienstes zu ermitteln, muss den entsprechenden Abhängigkeiten nachgegangen und durch Überprüfung der relevanten Dienstelemente der Fehler lokalisiert werden [DHHS06]. Soll zusätzlich noch der von einer Störung betroffene Kunde ermittelt werden, ist weiterhin Kenntnis des Dienstnutzungsprofils erforderlich [HSS05b].

Weiterhin resultiert aus den geschilderten Abhängigkeiten, dass Eigenschaften eines Dienstes (z.B. Bearbeitungsgeschwindigkeit von Anfragen) untrennbar mit den Eigenschaften dienstrealisierender Komponenten (z.B. Durchsatz von Applikations- und Netzelementen) verbunden sind. Für das Management des Dienstes ist es deshalb von Bedeutung, diese Zusammenhänge zu dokumentieren, um sie so zur Berechnung von Dienstmetriken verwenden zu können. Hinsichtlich der Informationsmodellierung erwächst daraus vor allem die Notwendigkeit, eine Abbildung zwischen bestehenden komponentenorientierten MIBs und den Inhalten einer Dienstmanagementinformationsbasis zu schaffen.

Aktualisierung von Dienstmanagementinformationen Eine weitere Herausforderung betrifft die Aktualisierung einer Dienstmanagementinformationsbasis bzw. Überwachung von Dienstmanagementinformationen. Wie aus dem vorhergehenden Absatz ersichtlich, sollte dies unter Einbeziehung dienstrealisierender Komponenten bzw. der Verdichtung von Managementinformationen geschehen. Entsprechende Überwachungswerkzeuge müssen deshalb in der Lage sein, mit verschiedensten Datenquellen umzugehen und Aggregationsoperationen durchzuführen. Dabei liegt die Schwierigkeit vor allem darin, mit der im Dienstmanagementumfeld anzutreffenden hohen Dynamik Schritt halten zu können, was vor allem unter Berücksichtigung des Dienstlebenszyklus deutlich wird: Da die Diensttopologie und technische Realisierung, also die Zusammensetzung der an der Diensterbringung beteiligten verteilten Komponenten, üblicherweise in der Aushandlungsphase entsteht und oftmals während der Betriebsphase modifiziert wird, müssen Überwachungswerkzeuge flexibel auf diese Änderungen reagieren können.

Konfiguration von Überwachungswerkzeugen muss unterstützt werden

Es erscheint deshalb wünschenswert, dass Überwachungswerkzeuge ihre Konfiguration unter Einbeziehung einer Dienstmanagementinformationsbasis selbständig vornehmen können. Dadurch sollen Änderungen in der Dienstimplementierung antizipiert und ein höherer Integrationsgrad zwischen der Modellsicht auf Managementinformationen und Managementwerkzeugen erreicht werden. Dieser Aspekt wurde in bestehenden Ansätzen nur sehr eingeschränkt berücksichtigt und kann demensprechend als weitere Herausforderungen bei der Konzeption einer Dienstmanagementinformationsbasis bzw. der Festlegung dienstorientierter Managementinformationen aufgefasst werden.

1.2 Untersuchte Fragestellungen

Mit der vorliegenden Arbeit wird das Ziel verfolgt, eine Dienstmanagementinformationsbasis zu schaffen und damit eine Integration des Dienstmanagements zu unterstützen. Wie im letzten Abschnitt dargelegt wurde, gilt es dabei eine Reihe von Herausforderungen zu adressieren, die im Folgenden anhand von Teilfragestellungen detailliert werden.

- **Welcher Informationsbedarf von Seiten der Dienstbetreiber an dienstbezogener Managementinformation ist vorhanden?**
  Der Definition von dienstbezogener Managementinformation geht eine umfassende Informationsanalyse voraus. Dabei muss – dem im letzten Abschnitt erwähnten Top Down Ansatz bei der Definition von Dienstmanagementinformation folgend – zunächst geklärt werden, welche Anforderungen von Seiten der Dienstbetreiber an Managementinformation bestehen.

- **Wie kann Dienstmanagementinformation modelliert und spezifiziert werden?**
  Dies beinhaltet die Auswahl eines geeigneten Beschreibungsrahmens, der eine adäquate Modellierung und Spezifikation dienstbezogener Managementinformation erlaubt. Da die Erstellung eines neuen Informationsmodells nicht im Fokus dieser Arbeit liegt, gilt es insbesondere zu klären, welche bestehenden Ansätze verwendet bzw. erweitert werden können um diese Zielsetzung zu erreichen.

- **Wie kann eine Abbildung zwischen komponenten- und dienstorientierter Managementinformation geschaffen werden?**
  Wie bereits erwähnt stehen standardisierte Managementinformationen für das Netz-, System- und Applikationsmanagement in großem Umfang zur Verfügung. Entsprechend muss der Frage nachgegangen werden, wie diese Informationen in die Etablierung einer Dienstsicht einbezogen werden bzw. auf Diensteigenschaften abgebildet werden können. Dies umfasst Möglichkeiten zur Spezifikation von Abbildungsvorschriften und deren Einbindung in eine Dienstmanagementinformationsbasis.

- **Wie kann eine Dienstmanagementinformationsbasis aktualisiert und operationalisiert werden?**
  Wie am Ende des letzten Abschnitts dargelegt wurde, müssen mit der Aktualisierung einer Dienstmanagementinformationsbasis betraute Werkzeuge adaptiv agieren können. Deshalb soll untersucht werden, inwieweit eine automatisierte Konfiguration dieser Werkzeuge vorgenommen werden kann. Desweiteren setzt sich diese

Teilfragestellung mit Möglichkeiten zur Einbindung einer Dienstmanagementinformationsbasis in den täglichen Betrieb eines Providers auseinander.

1.3 Vorgehensmodell dieser Arbeit

Um einen möglichst weitreichenden Ansatz für die Spezifikation von dienstbezogener Managementinformation zu entwickeln, wird ein Top Down Vorgehen gewählt, das im Nachfolgenden beschrieben und in Abbildung 1.1 dagestellt wird.

In Kapitel 2 wird zunächst eine umfassende Analyse von Anforderungen an eine Managementinformationsbasis für das Dienstmanagement vorgenommen, die sich zum Großteil auf reale Szenarien stützt. In die Betrachtung fließen dabei der virtuelle Web-Hosting Dienst des Leibniz Rechenzentrums (LRZ) und ein GRID-Szenario aus dem Umfeld des D-GRID Projekts mit ein. Das Ergebnis dieser Analyse stellt ein strukturierter Anforderungskatalog dar, der im weiteren Verlauf der vorliegenden Arbeit als Bewertungsinstrument sowohl für bestehende Arbeiten als auch den eigenen Lösungsansatz dient.

Gegenstand von Kapitel 3 ist eine Untersuchung der bestehenden Ansätze zu der in Abschnitt 1.2 formulierten Fragestellung dieser Arbeit. Dazu werden Arbeiten von Standardisierungsgremien, einschlägige Forschungsarbeiten und konkrete Produkte verschiedener Hersteller von Managementanwendungen einer Bewertung gemäß dem entwickelten Anforderungskatalog unterzogen. Die Analyse mündet in der Feststellung, dass einerseits dienstorientierte Managementinformation nicht in ausreichendem Maße spezifiziert und andererseits eine Abbildung zwischen komponenten- und dienstorientierter Managementinformation bisher nicht adäquat adressiert wurde.

Ansatz stützt sich auf vierstufige Methodik

Die genannten Aspekte münden in der Notwendigkeit eines neuen Ansatzes (*Service-MIB*), der sich auf eine vierstufige Methodik stützt und in Kapitel 4 erstmals vorgestellt wird. Die vorrangige Idee besteht dabei darin, die Konzeption einer Service-MIB in Phasen zu unterteilen und sukzessive Lösungsmöglichkeiten für die einzelnen Phasen zu entwickeln. Entsprechend dient dieses Kapitel primär dem Zweck, den in

dieser Arbeit entwickelten Ansatz übersichlich darzustellen und somit die in den nächsten Kapiteln erfolgende Durchführung der Methodik einzurahmen.

Informationsanalyse anhand mehrerer Gesichtspunkte

Den ersten Schritt der Methodik bildet die in Kapitel 5 vorgestellte *Analysephase*. Sie widmet sich einer Feststellung des Bedarfes an dienstorientierter Managementinformation und geht somit der Frage nach, welche Informationen zur Erfüllung von Dienstmanagementaufgaben aus Betreibersicht effektiv benötigt werden. Neben der Berücksichtigung von MNM-Dienstmodell, Dienstlebenszyklus und interorganisationalen Aspekten sind es hierbei vorangig Beschreibungen von Managementprozessen im Rahmen der *IT Infrastructure Library (ITIL)*, die als Grundlage zur Ableitung von Informationsanforderungen dienen. Mit der Analysephase wird somit das Fundament für eine weiterführende Spezifikation von konkreter Dienstmanagementinformation geschaffen.

Definition konkreter Managementinformationen

Daran anschließend wird in Kapitel 6 die *Spezifikationsphase* eingeleitet und somit die Modellierung und Definition von Dienstmanagementinformationen adressiert. Ziel hierbei ist es, die vorab ermittelten Informationsanforderungen in geeignete Entitäten, Attribute und Beziehungen zu überführen und damit eine Modellsicht zu erstellen. Weiterhin wird in diesem Kapitel die Abbildung zwischen komponenten auf dienstorientierter Managementinformation weiter konkretisiert und dafür eine deklarative Spezifikationssprache für Dienstmanagementinformationen in Abhängigkeit von Komponentenparamtern, die *Service Information Specification Language (SISL)*, entwickelt. Dies wird notwendig, da bisher nur domänen- (z.B. Spezifikation von SLA-Parametern) oder technologiespezifische (nur mit einer bestimmten Managementarchitektur nutzbare) Ansätze vorhanden sind.

Aktualisierung und Operationalisierung

Weiter ausgeführt werden danach in Kapitel 7 die *Überwachungs-* und *Nutzungsphase* dieser Arbeit unterliegenden Methodik. Dabei wird zunächst die Erweiterung einer bestehenden Überwachungsarchitektur vorgenommen und somit ein Aktualisierungsmechanismus für die Service-MIB geschaffen. Die daraus entstehende *Service Monitoring Architecture (SMONA)* ermöglicht die Überwachung von Diensteigenschaften basierend auf SISL-Spezifikationen, stellt darüberhinaus das Bindeglied zu der vorhergehenden Spezifikationsphase dar und schafft somit eine Integration zwischen den Phasen der Methodik. Weiterhin

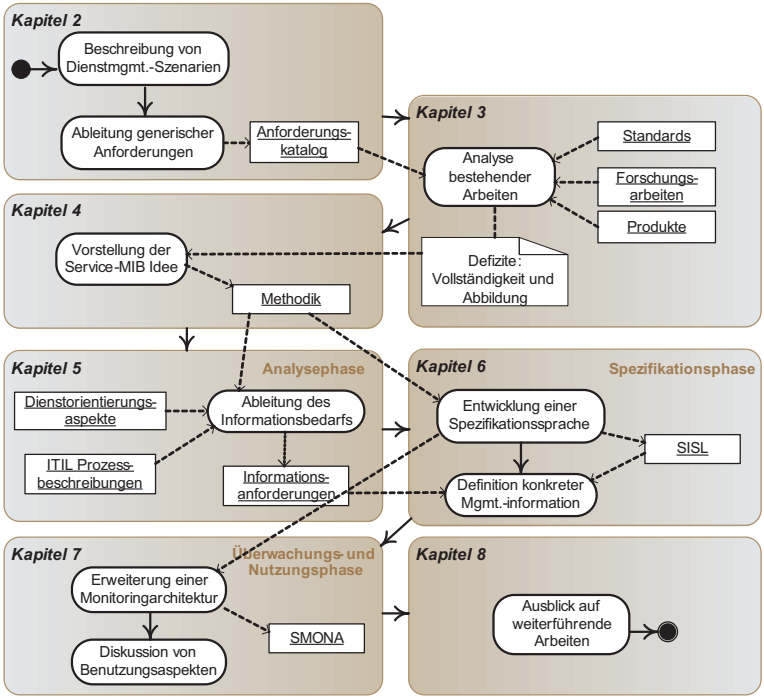

Abbildung 1.1: Vorgehensmodell dieser Arbeit

werden in der *Nutzungsphase* einer Service-MIB auftretende Aspekte diskutiert. Dies umfasst die Einbindung des Lösungsansatzes in bestehende Managementarchitekturen am Beispiel des *Common Information Models* sowie die Vorstellung von Vorgehensrichtlinien zur szenariospezifischen Verfeinerung einer Service-MIB.

Die Zusammenfassung der wichtigsten Ergebnisse und die Vorstellung weiterführender Forschungsfragestellungen in Kapitel 8 runden die vorliegende Arbeit ab.

Kapitel 2

Anforderungsanalyse

Um die im letzten Kapitel skizzierten Fragestellungen umfassend beantworten zu können, ist es zunächst notwendig, Anforderungen an potentielle Lösungsmöglichkeiten zu bestimmen. Zu diesem Zweck werden im Folgenden Managementherausforderungen anhand von zwei Referenzszenarien ermittelt, konsolidiert und verallgemeinert. Somit entsteht ein Anforderungskatalog, der eine strukturierte Bewertung verwandter Arbeiten und der in dieser Arbeit entwickelten Lösungen ermöglicht.

Auswirkungen auf Informationsmodellierung

Wie in der Einleitung dargelegt wurde, beschäftigt sich die vorliegende Arbeit mit dem Einfluss von Dienstorientierungsaspekten auf die Modellierung und Spezifikation von Managementinformationen. Um in diesem Kapitel identifizierte Anforderungen ordnungsgemäß einzuordnen, werden deshalb zunächst relevante Begriffe und Konzepte vorgestellt.

2.1 Informationsmodellierung im Management

Uneinheitliche Terminologie in der Fachliteratur

Die Modellierung von Managementinformationen stellt ein essentielles Konzept von Managementarchitekturen und -rahmenwerken dar und wird dementsprechend in zahlreicher Fachliteratur behandelt (siehe z.B.

[HAN99, Bla92]). Allerdings lässt sich – vor allem in neueren Arbeiten – eine Uneinheitlichkeit in den verwendeten Begriffen feststellen. Der genannte Aspekt lässt die Schaffung eines begrifflichen Rahmens für die vorliegende Arbeit notwendig erscheinen, der sich vorwiegend an Arbeiten von Standardisierungsgremien orientiert und im Folgenden vorgestellt wird.

2.1.1 Managementobjekte und MIBs

Das wohl grundlegendste Konzept in der Modellierung von Managementinformationen stellt die Beschreibung von *Managementobjekten* (MOs) dar. Es wurde im Rahmen des OSI Management Frameworks [ISO89b] definiert und bildet seitdem die Basis für die meisten existierenden Netz- und Systemmanagementarchitekturen.

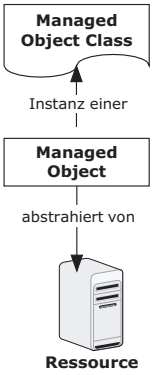

Abbildung 2.1: Managementobjekte und -objektklassen

Die Grundidee des MO-Konzepts fußt auf Abstraktion: Managementoperationen werden nicht direkt auf den zu managenden Ressourcen ausgeführt, sondern stattdessen auf deren Repräsentanten, den Managementobjekten [HAN99]. Dies ermöglicht es, den internen Aufbau von

Ressourcen zu verschatten und lediglich aus Sicht des Managements relevante Eigenschaften zu berücksichtigen. Entsprechend werden für eine Netzkomponente (z.B. Router) keine Implementierungsdetails als MO modelliert, sondern ausschließlich Parameter (z.B. der Durchsatz pro Port), die zur Erfüllung von Managementaufgaben benötigt werden.

Wie in Abbildung 2.1 dargestellt, repräsentieren MOs mit einer gemeinsamen Definition Instanzen einer bestimmten *Managementobjektklasse* (MOC) [ISO93]. Grundsätzlich ist festzustellen, dass die Bestandteile eines MOs bzw. einer MOC von dem zugrundeliegenden Modellierungsansatz festgelegt (siehe Abschnitt 2.1.2) werden: Während dies im einfachsten Fall eine Reihe von Attributen zur Beschreibung von Eigenschaften des MOs umfasst, sehen komplexere Modelle zusätzlich Beschreibungsmöglichkeiten für Aktionen, Meldungen und Verhaltenseigenschaften vor [HAN99].

MIB bezeichnet Kompositum an Managementinformation

Managementobjekte, die einem bestimmten System zugeordnet sind, werden als *Managementinformationsbasis* (MIB) dieses Systems bezeichnet. MIBs repräsentieren damit ein Kompositum von einem offenen System zur Verfügung gestellter Managementinformationen. Der Begriff System darf in diesem Zusammenhang nicht zu eng gefasst werden: In Abhängigkeit von der Betrachtungsweise des Managements können verschiedene Sachverhalte einem System zugeordnet werden (z.B. auch Organisationen).

An dieser Stelle muss beachtet werden, dass sowohl MOs als auch MIBs in erster Linie Grundkonzepte darstellen; für eine Formalisierung bzw. computergestützte Verarbeitung müssen diese Konzepte in eine geeignete Modellierung überführt werden. Welche Gesichtspunkte in diesem Zusammenhang berücksichtigt werden müssen, soll im nächsten Abschnitt aufgezeigt werden.

2.1.2 Modellierung von Managementinformation

MOs können auf verschiedenen Ebenen definiert werden

Ähnlich der Beschreibung von Computerprogrammen im *Software Engineering* werden zur Repräsentation von Managementobjekten eine Reihe von Konzepten benötigt. Zunächst muss ein Beschreibungsrahmen festgelegt werden, der es gestattet, managementrelevante Aspekte einer Ressource in Form eines Modells abzubilden. Dieses umfasst, neben

der Festlegung einer Syntax zur Definition von Modellelementen (*Modellierungssprache*), die Auswahl eines Detailgrades, in dem Managementinformationen modelliert werden (*Informations- vs. Datenmodell*). Letzterer wird vor allem davon beeinflusst, inwieweit implementierungsspezifische Aspekte in das entstehende Modell mitaufgenommen werden sollen.

Es darf nicht unerwähnt bleiben, dass die in diesem Zusammenhang auftretenden Begriffe oftmals in abweichenden Bedeutungen verwendet werden: Beispielsweise wird der Beschreibungsrahmen eines Managementmodells, also die Kombination aus Modellierungsansatz (z.B. objektorientiert) und Modellierungssprache, ebenfalls als Informationsmodell bezeichnet. Um eine Einheitlichkeit bezüglich der Begriffsverwendung zu erzielen, veröffentlichte die Internet Engineering Task Force (IETF) deshalb zwei *Requests for Comments (RFCs)* [WSS+01, PS03]. Da die Begriffsdefinitionen in diesen Dokumenten insbesondere Konformität mit der Terminologie neuerer Managementmodelle (siehe Vorstellung in Abschnitt 3) aufweisen, finden sie in der vorliegenden Arbeit Verwendung. Dementsprechend werden drei grundlegende Elemente in der Beschreibung von Managementobjekten unterschieden:

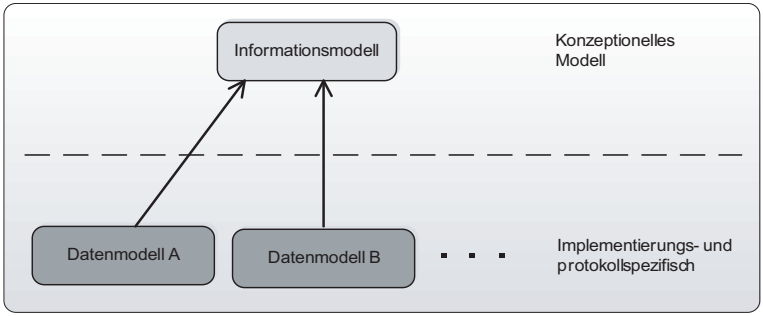

Abbildung 2.2: Unterscheidung zwischen Informations- und Datenmodellen in Anlehnung an [PS03]

Informationsmodell (IM) Auf der Ebene des Informationsmodells werden einem Managementobjekt zugrundeliegende Konzepte definiert bzw. modelliert. Das daraus entstehende konzeptionelle Modell dient somit dem Zweck, grundlegende Zusammenhänge und Hintergründe aufzuzeigen bzw. zu erklären, *warum* ein Managementobjekt auf eine gewisse Weise modelliert wurde. Konsequenterweise werden Sachverhalte in einem Informationsmodell unabhängig von bestimmten Managementplattformen, Protokollen oder Daten-Repositories dargestellt – eine unmittelbare Implementierung eines IMs ist deshalb nicht vorgesehen (siehe auch [Str04, WSS+01, PS03]). Nachfolgendes Zitat aus [WSS+01] stellt diesen Zusammenhang nochmals dar:

Informationsmodell stellt zugrundeliegende Konzepte dar

> *An Information Model is an abstraction and representation of the entities in a managed environment, their properties, attributes and operations, and the way that they relate to each other. It is independent of any specific repository, software usage, protocol, or platform.*

Während Informationsmodelle grundätzlich auch rein informell beschrieben werden können (z.B. textuelle Erläuterung), wird verstärkt einem grafischen Repräsentationsformat (z.B. E/R-Diagramme, UML-Klassendiagramme) der Vorzug gegeben.

Datenmodell (DM) Wie bereits erwähnt, eignen sich Informationsmodelle nicht zur unmittelbaren Implementierung, sondern müssen vielmehr um protokoll- und plattformspezifische Merkmale erweitert werden. Daraus entstehende *Datenmodelle* weisen einen geringeren Abstraktionsgrad auf, beinhalten implementierungsspezifische Details (z.B. Naming-Konstrukte) und erlauben eine einfache Umsetzung in ein Daten-Repository. Letzerer Aspekt wird insbesondere in folgender Definition aus [Str04] verdeutlicht:

Datenmodell verfeinert ein Informationsmodell

> *A data model is a concrete implementation of an information model in terms appropriate to a specific type of repository that uses a specific access protocol or protocols. It includes data structures, operations, and rules that define how the data is stored, accessed and manipulated.*

Der Zusammenhang zwischen Daten- und Informationsmodellen wird in Abbildung 2.2 dargestellt: Da konzeptionelle Modelle typischerweise auf verschiedene Arten implementiert werden können, kann ein Informationsmodell in mehrere Datenmodelle überführt werden.

DMS legt Syntax des Datenmodells fest

Datenmodellierungssprache (DMS) Die Datenmodellierungssprache (DMS) bestimmt das Repräsentationsformat des Datenmodells; sie legt in erster Linie fest, in welcher Syntax Entitäten innerhalb des Datenmodells spezifiziert werden. Um eine computergestützte Verarbeitung zu ermöglichen, muss eine Datenmodellierungssprache ferner die Formalisierung von Informationen unterstützen und wird deshalb selbst oftmals als formale Sprache konzipiert.

Nachdem mit der Einführung von MOs, MIBs und Modellierungsebenen grundlegende Konzepte in der Modellierung von Managementinformationen dargelegt wurden, widmet sich der nächste Abschnitt Aspekten, die bei der Umsetzung dieser Konzepte bzw. Definition konkreter Managementinformation auftreten.

2.1.3 Definition konkreter Managementinformation

Basierend auf den im vorhergehenden Abschnitt genannten Modellen erfolgt eine Definition konkreter Managementinformation, d.h. die Festlegung von Managementobjekten in Form von dazugehörigen Attributen, Operationen usw. Dies stellt einen entscheidenden Schritt dar, da hiermit einerseits die Funktionalität von Managementanwendungen festgelegt und andererseits die Grundlage für eine Interoperabilität von Werkzeugen gelegt wird [Neu93].

Entsprechend haben sich in den letzten Jahren mehrere Gremien um die Standardisierung konkreter Managementinformationen bemüht und im Zuge dessen eine Sammlung sogenannter Objektkataloge veröffentlicht (siehe Kapitel 3). Diese enthalten eine Reihe von standardisierten Managementobjekten bzw. Managementobjektklassen und ermöglichen so eine einheitliche Sicht auf Ressourcen über verschiedene Werkzeuge hinweg.

Szenario-spezifische Verfeinerung kann notwendig erscheinen

Möchte ein Provider allerdings Managementinformationen für ein bestimmtes Szenario definieren, stellt sich oftmals die Frage, inwieweit standardisierte Objektkataloge seinen Bedürfnissen entsprechen bzw. er Anpassungen vornehmen muss. In diesem Zusammenhang gilt es drei grundlegende Fälle zu unterscheiden:

1. *Der Provider übernimmt standardisierte Managementinformationen* Die von standardisierten Managementmodellen zur Verfügung gestellte Sammlung an Objektkatalogen wird unverändert übernommen, es erfolgt keine Modifikation durch den Provider (z.B. Erstellung eigener Attribute). In diesem Fall erweist sich die standardisierte Information für den Provider als ausreichend, er kann sowohl seinen eigenen Informationsbedarf als auch den seiner Kunden durch bestehende Objektkataloge abdecken.

2. *Standardisierte Managementinformation wird vom Provider erweitert.* Basierend auf standardisierter Managementinformation nimmt der Provider eine szenariospezifische Erweiterung vor, z.B. durch Hinzufügen von neuen Klassen und Attributen in objektorientierten Modellen. Mehrere Gesichtspunkte können diese Erweiterungen notwendig erscheinen lassen (vgl. Template-Hierarchie in [DR02]): Die Kunden des Providers könnten Informationen bezüglich des von ihnen in Anspruch genommmenen Dienstes nachfragen, die nicht durch die standardisierten Managementmodelle abgedeckt werden, aber Teil von vertraglichen Vereinbarungen sind. Ebenso denkbar ist ein Bedarf seitens des Providers an spezifischen Metriken zur Durchführung interner Managementaufgaben.

3. *Der Provider spezifiziert Managementinformationen eigenständig.* In diesem Fall entschließt sich der Provider, die gesamte Managementinformation eigenständig und spezifisch für das jeweilige Szenario zu definieren. Er verzichtet bewusst auf die Unterstützung durch standardisierte Objektkataloge.

Obwohl der letzte Fall eine Extremform darstellt, tritt er dennoch vereinzelt in kleineren Managementszenarien und in Kombination mit vom Provider selbstentwickelten Managementwerkzeugen auf. In der Praxis

allerdings weitaus häufiger anzutreffen ist der zweite Fall, wie sich auch in der nachfolgenden Analyse von Managementszenarien zeigen wird.

Erweiterungsmechanismen werden benötigt

Weiterhin gilt es zu beachten, dass diese Ausprägungen wiederum Auswirkungen auf Informations- und Datenmodelle sowie Überwachungswerkzeuge haben. Insbesondere im zweiten Fall müssen entsprechende Erweiterungsmechanismen auf Modellebene vorhanden und die Möglichkeit zur Anpassung des Überwachungsvorgangs auf providerspezifische Metriken gegeben sein.

2.2 Analyse von Managementszenarien

Szenarien repräsentieren unterschiedliche Facetten der Problematik

Zwei Referenzszenarien, ein Web-Hosting und ein GRID Dienst, wurden für die Anforderungsanalyse ausgewählt. Sie repräsentieren unterschiedliche Ausprägungen hinsichtlich der im Dienstmanagement auftretenden Problematik: Während es sich bei dem Webhosting-Dienst um einen klassischen "Massendienst" handelt, der von vielen Providern in ähnlicher Form angeboten wird, stellen GRID-Dienste noch relativ neue Anwendungsszenarien dar, die vor allem im Bezug auf die dabei auftretende Dynamik eine Vielzahl neuer Anforderungen an Service Provider stellen.

Zusammen mit den Szenariobeschreibungen werden typische Problemstellungen genannt und somit Managementherausforderungen veranschaulicht. Eine strukturierte Einordnung und Konsolidierung dieser Herausforderungen in einen Anforderungskatalog erfolgt anschließend in Abschnitt 2.3.

2.2.1 Web-Hosting Szenario

LRZ stellt Web-Hosting Dienst für Universitäten bereit

Das Leibniz Rechenzentrum (LRZ) ist für den Betrieb der Infrastruktur des Münchener Wissenschaftsnetzes (MWN) zuständig und ermöglicht im Zuge dessen die Vernetzung der wissenschaftlichen Organisationen in München und Umgebung. Neben der Verwaltung einer Netzinfrastruktur mit über 60 Standorten und mehr als 60.000 angeschlossenen Geräten bietet das LRZ höherwertige Applikationsdienste für affiliierte Institutionen an. Darunter fällt auch unter anderem die im Folgenden

betrachtete Bereitstellung von (virtuellen) Webservern für Institute der angeschlossenen Universitäten und universitätsnahen Einrichtungen.

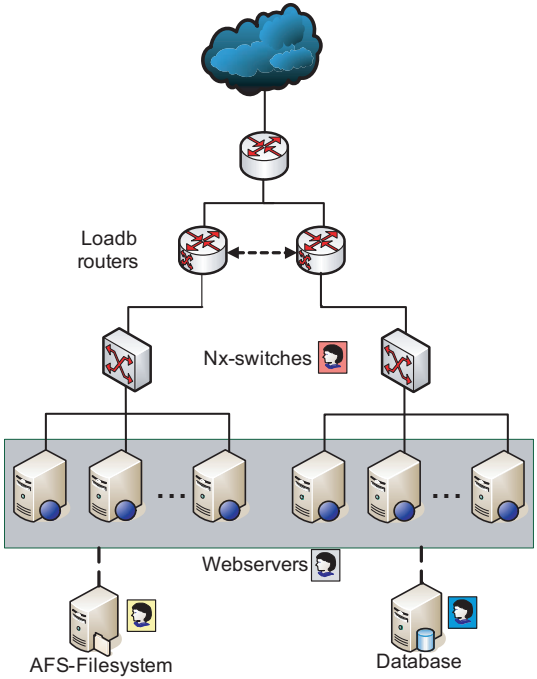

Abbildung 2.3: Web-Hosting Dienst am LRZ

Der Web-Hosting Dienst wird intern durch einen Pool von 20 Webservern realisiert und verfolgt die Zielsetzung, eine möglichst unterbrechungsfreie Außendarstellung der Institute mit ca. 300 virtuellen Webservern zu gewährleisten. Wie aus Abbildung 2.3 ersichtlich wird, sind neben den Webservern eine Storage-Komponente (*AFS*), ein Datenbank-Cluster sowie mehrere Koppelelemente (*Loadb routers*, *Nx-switches*) an der Diensterbringung beteiligt. Aus Gründen der Fehlertoleranz bzw. Lastverteilung wurden zusätzlich Komponenten redundant

ausgelegt und sind deshalb mehrfach vorhanden.

Organisatorische Aufteilung

Für die Größe dieses Szenarios typisch ist die organisatorische Aufteilung der Diensterbringung: Dedizierte Organisationseinheiten sind mit der Verwaltung einzelner Dienstkomponenten betraut. Wie in Abbildung 2.3 durch entsprechende Symbole angedeutet wird, liegt für den Web-Hosting Dienst dabei eine Aufteilung in die Abteilungen Kommunikationsnetze, Datenbanken, Storage und Web vor. Jede Abteilung führt eine Reihe von Managementaufgaben aus bzw. verwaltet Managementinformationen, um eine reibungslose Funktion des Web-Hosting Dienstes sicherzustellen. Die dafür eingesetzten Managementwerkzeuge sind allerdings abteilungsspezifisch; während die Abteilung Kommunikationsnetze *HP OpenView Network Node Manager* zur Überwachung der an der Diensterbringung beteiligten Router und Switches benutzt, greift die mit der Administration der Datenbank betraute Abteilung zur Erfüllung der gleichen Aufgabe auf *Nagios* zurück.

Herausforderungen für das Management

Übergang zum Dienstmanagement stellt neue Anforderungen

Während der am LRZ eingesetzte Managementansatz ein effektives und integriertes Netz- und Systemmanagement ermöglicht, erfordern veränderte Marktbedingungen und Kundenwünsche eine verstärkt dienstorientierte Betrachtung. Der Fokus der verwendeten Werkzeuge, respektive der verwalteten Managementinformation, liegt allerdings bisher auf den dienstrealisierenden Komponenten – Aussagen über den Dienst als Ganzes können nicht getroffen werden. Beispielsweise kann der Status des Routers oder eines Web-Servers mit Hilfe der verwendeten Managementtools einfach bestimmt werden, was nicht auf den Zustand des von diesen Komponenten realisierten Web-Hosting Dienstes zutrifft. Als Konsequenz werden anfallende Dienstmanagementaufgaben anhand von Betriebserfahrung vorgenommen, ein strukturierter automatisierter Austausch von Informationen erfolgt dabei nur eingeschränkt. Um den angestrebten Übergang zum Dienstmanagement zu vollziehen, müssen demzufolge eine Reihe neuer Herausforderungen bewältigt werden, die im Folgenden skizziert werden:

- *Herstellung einer Managementsicht auf den Dienst*
  Das LRZ möchte zunächst den Web-Hosting Dienst in ähnlicher Weise wie bestehende Netz- und Applikationskomponenten managen. Die entscheidende Prämisse hierfür stellt eine Managementsicht auf den Dienst dar, d.h. eine für das Management erfassbare Beschreibung des Dienstes. Dafür müssen managementrelevante Charakteristika des Dienstes modelliert und somit der Dienst, wie in Abschnitt 2.1 beschrieben, als Managementobjekt aufgefasst werden.

- *Austausch von Dienstmanagementinformationen*
  Um einen kooperativen Betrieb des Dienstes über die verschiedenen Abteilungen hin zu ermöglichen, muss ferner sichergestellt werden, dass die beteiligten Personen/Managementwerkzeuge eine gemeinsame Managementsicht auf den Dienst teilen. Dies erfordert primär den Zugriff auf eine gemeinsame Datenbasis für Dienstmanagementinformationen, deren Struktur und Modellierungsansatz ebenfalls festgelegt werden müssen. Darüber hinaus legt das LRZ Wert auf eine Integration in die vorhandene Managementlandschaft: Eine neue Lösung ohne Modifikation der bestehenden Werkzeuge und Managementmodelle verwenden zu können, gilt hierbei als der vordringlichste Wunsch.

- *Emittlung des Informationsbedarfs*
  Zu ermitteln, welche Managementinformationen für den Web-Hosting Dienst tatsächlich benötigt werden, stellt eine weitere Managementherausforderung dar. Dabei lässt sich prinzipiell der Wunsch nach einer möglichst weitreichenden Unterstützung der anfallenden Dienstmanagementaufgaben über den gesamten Dienstlebenszyklus hinweg feststellen. Es sollte insbesondere möglich werden, Managementinformationen beginnend mit der Planung bis zum tatsächlichen Betrieb des Dienstes konsistent zu erfassen. Welche Informationen dabei konkret benötigt werden, variiert allerdings mit den Ansprüchen der jeweiligen Abteilung: Beispielsweise möchte die Abteilung *Web* Statistiken zu der Verfügbarkeit und Auslastung des Dienstes erheben, um Leistungsberichte gegenüber den Kunden erstellen zu können. Die Abteilung Kommunikationsnetze hingegen ist mehr an einer lückenlosen Do-

kumentation funktionaler, temporaler und topologischer Abhängigkeiten zwischen dem Web-Hosting Dienst und den vorhandenen Netzkomponenten interessiert, um somit Dienstfehler schnell und proaktiv zu erkennen. Aus diesen unterschiedlichen Anforderungen wird deutlich, dass an dieser Stelle ein methodisches Vorgehen notwendig wird, mit Hilfe dessen der Informationsbedarf umfassend und nachvollziehbar ermittelt werden kann.

- *Betrieb des Dienstes in Einklang mit vertraglichen Vereinbarungen*
  Von Seiten der Kunden besteht verstärkt der Bedarf nach festgelegten Qualitätseigenschaften des Dienstes und deren vertragliche Absicherung in Service Level Agreements (SLAs). Das LRZ möchte diese Entwicklung antizipieren und Managementlösungen schaffen, die einen Betrieb des Web-Hosting Dienstes in Einklang mit vertraglichen Vereinbarungen gewährleisten. Aus Sicht der Informationsmodellierung ergibt sich somit die Notwendigkeit, Beziehungen zwischen dem Dienst und den entsprechenden Kunden bzw. SLAs zu erfassen. Nicht zuletzt soll es damit Managementwerkzeugen (z.B. [Han07, Sch07]) ermöglicht werden, evtl. Verstöße gegen Dienstvereinbarungen frühzeitig zu erkennen.

- *Einbettung in Managementprozesse*
  Das LRZ steht im Begriff, den organisatorischen Ablauf des Dienstmanagements an den Vorgaben der IT Infrastructure Library (siehe Abschnitt 3.1.4) auszurichten. Dies umfasst die Einführung dokumentierter Managementprozesse, die Festlegung von Prozessverantwortlichen und die Einführung von Prozesskennzahlen. Für die Informationsmodellierung erwächst daraus die Herausforderung, zur Ausführung dieser Prozesse benötigte Daten zur Verfügung zu stellen und in der Festlegung von Dienstmanagementinformationen zu berücksichtigen.

Wie aus den genannten Punkten hervorgeht, besteht eine vorrangige Zielsetzung einer Informationsbasis für Dienstmanagementinformationen darin, für eine kooperative Bewältigung von Dienstmanagementaufgaben notwendige Informationen bereitzustellen. Als weitere Herausforderungen können in diesem Zusammenhang auch die Erfassung

von Abhängigkeitsbeziehungen und Überwachung von Dienstmanagementinformationen angesehen werden. Sie treten besonders deutlich in dem nachfolgenden Grid-Szenario auf und werden deshalb an dieser Stelle zusammen mit weiteren Aspekten vorgestellt.

2.2.2 GRID Computing Szenario

Im Gegensatz zu dem Web-Hosting Dienst stellen GRID-Applikationen eine relativ neue Art von Diensterbringungsszenarien dar. Darüber hinaus tragen der hohe Verteilungsgrad von GRID-Ressourcen und das Auftreten von in administrativer und legislativer Sicht verschiedener Akteure zu einer Vielzahl neuer Managementfragestellungen bei. Welche Herausforderungen dabei insbesondere für eine Dienstmanagementinformationsbasis erwachsen, wird im Folgenden anhand eines, aus dem D-Grid Projekt entlehnten, Szenarios illustriert.

Hohe Anforderungen bzgl. Rechen- und Speicherkapazität

Das vom Bundesministerium für Bildung und Forschung (BMBF) geförderte D-Grid Projekt (http://www.d-grid.de) verfolgt zum Ziel, eine robuste, flexible und nachhaltige Grid-Infrastruktur zur wissenschaftlichen Verwendung aufzubauen. Als Benutzer dieser Infrastruktur treten Grid-Communities auf, die sich kooperativ einem bestimmten Forschungsvorhaben widmen. Beispielsweise beschäftigt sich das HEP (High Energy Physics) Community Grid mit der Auswertung von Messergebnissen, die mit Hilfe des Large Hadron Collider (LHC) am European Organization for Nuclear Research (CERN) gewonnen wurden. Besondere Herausforderungen stellen dabei die dafür benötigten Datenhaltungs- und Rechenkapazitäten dar: Es werden jährlich etwa 15 Petabyte an Daten erwartet, deren Verwaltung und Auswertung von mehreren Tausenden, weltweit verteilten Forschern vorgenommen werden soll. Zur Bewältigung dieser Datenmenge stellen alle beteiligten Organisationen Speicher- und Rechenleistung zur Verfügung, auf die wiederum mit Hilfe der Grid-Infrastruktur zugegriffen werden kann.

Abhängigkeiten zwischen Diensten und Komponenten treten auf

Das in Abbildung 2.4 dargestellte Szenario stellt eine Vereinfachung des in diesem Grid auftretenden Computing-Dienstes dar. Er setzt sich aus einer Sammlung verschiedener Computing-Dienste zusammen, die jeweils von den angeschlossenen Einrichtungen betrieben werden. Dadurch bedingt besteht eine funktionale Abhängigkeit des Grid-

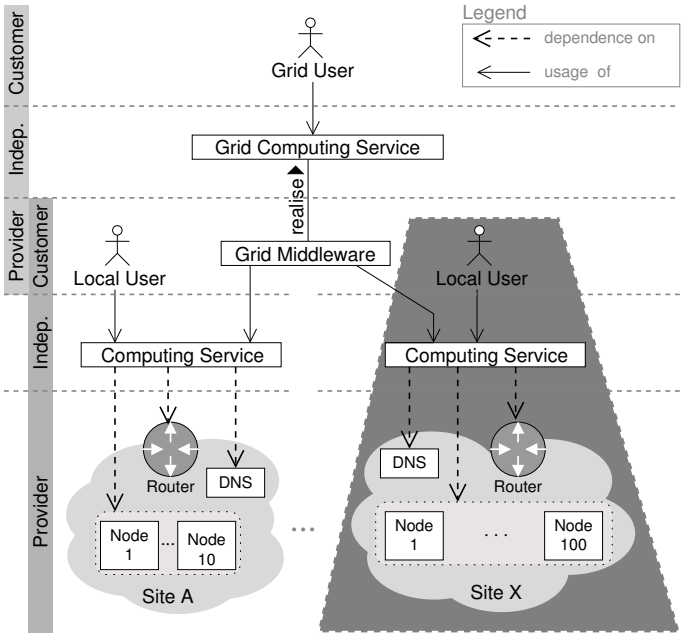

Abbildung 2.4: Vereinfachtes Grid-Computing Szenario aus [DSgF07]

Computing-Dienstes von den genannten Subdiensten, d.h. sein Status wird von lokalen Computing-Diensten bestimmt. Auf der Ebene der Computing-Dienste treten wiederum Abhängigkeiten zu den dienstrealisierenden Komponenten auf. In dieser vereinfachten Form des Szenarios sind dies Router, DNS-Dienst und lokale Knotenrechner, auf denen die eigentlichen Rechenvorgänge ausgeführt werden.

Die Aufgabe der Grid-Middleware besteht darin, diesen internen Aufbau vor dem Nutzer zu verschatten: Für ihn wird nur der Grid-Computing-Service und die dadurch zur Verfügung gestellte Funktionalität sichtbar. Im Gegensatz dazu muss sich der Provider mit dem internen Aufbau des Dienstes auseinandersetzen, will er einen möglichst unterbrechungsfreien Betrieb sicherstellen. Welche neuen Herausforderungen dadurch

für eine Dienstmanagementinformationsbasis entstehen, soll im nächsten Abschnitt näher erläutert werden.

Managementherausforderungen

Analog zu dem Web-Hosting Dienst besteht die übergeordnete Herausforderung darin, eine Managementsicht auf den Grid-Computing Dienst herzustellen. In diesem Fall kommt allerdings erschwerend hinzu, dass die Bereitstellung dieses Dienstes über mehrere autonome Domänen (graue Raute in Abbildung 2.4) verteilt ist – und damit dömanenspezifische Technologien und Managementwerkzeuge auftreten. Entsprechend betreibt jede Domäne bereits ein Netzmanagement, überwacht angeschlossene Komponenten und verwaltet mit diesen Komponenten assoziierte Managementinformation. Soll der Veränderungsaufwand für die einzelnen Domänen möglichst gering gehalten werden, müssen deshalb die vorhandenen Managementinformationen zu einer umfassenden Dienstsicht verknüpft werden:

- *Definition szenariospezifischer Dienstmanagementinformationen*
  Im Hinblick auf die Definition konkreter Dienstmanagementinformation liegt das Bestreben des D-GRID Projekts darin, möglichst auf die Vorgaben standardisierter Managementmodelle zurückzugreifen, um eine Interoperabilität hinsichtlich zukünftiger Managementwerkzeuge zu gewährleisten. Allerdings wird es vor allem wegen der Neuartigkeit dieses Szenarios notwendig, spezifische Dienstmerkmale festzulegen, um z.B. internen Planungsaufgaben nachzukommen. Dafür müssen Möglichkeiten zur flexiblen Spezifikation szenariospezifischer Dienstmanagementinformation geschaffen bzw. bestehende Modelle entsprechend erweitert werden.

- *Erfassung der Abhängigkeitsbeziehungen*
  Wie sich aus der Beschreibung entnehmen lässt, treten in diesem Szenario eine Reihe von Abhängigkeitsbeziehungen auf, die eine reibungslose Funktion des Dienstes nachhaltig beeinflussen. Dies umfasst Abhängigkeiten zwischen dem GRID-Dienst und den einzelnen Computing Diensten gleichermaßen wie Abhängigkeitsbe-

ziehungen zwischen Diensten und vorhandenen Netz- und Applikationskomponenten. Entsprechend besteht eine weitere Managementherausforderung darin, diese Beziehungen strukturiert hinsichtlich ihrer funktionalen, topgologischen und temporalen Ausprägung sowie Kritikalität zu beschreiben und sie somit Managementanwendungen zur Verfügung zu stellen.

- *Verdichtung von Managementinformationen*
  Obwohl der GRID-Computing Dienst eine Komposition mehrerer Subdienste darstellt, sind es letztendlich doch Eigenschaften des Gesamtdienstes, die gegenüber dem Kunden verantwortet werden müssen. Entsprechend besteht eine Managementaufgabe darin, die in diesem Szenario vorhandenen Managementinformationen zu verdichten (aggregieren), um Aussagen über den Gesamtdienst treffen zu können. Beispielsweise müssen Auslastungswerte des Routers, des DNS-Dienstes und der Rechnerknoten in einer Weise kombiniert werden, die eine Gesamtauslastung des Dienstes widerspiegelt. Aus Sicht der Informationsmodellierung muss deshalb eine Beschreibungsmöglichkeit für derartige Aggregationen geschaffen und formal dargestellt werden.

- *Überwachung des Dienstes*
  Um Aussagen über den aktuellen Zustand des Dienstes treffen zu können bzw. Ausfälle möglichst frühzeitig zu antizipieren, müssen aktuelle Werte für die vorab definierte Dienstmanagementinformation ermittelt werden. Dies bedingt die Einführung einer ständigen Überwachung (Monitoring) des Dienstes mit Hilfe geeigneter Werkzeuge. Allerdings sollte dies möglichst unter Einbeziehung der vorhandenen Toollandschaft geschehen – ein kompletter Austausch zugunsten neuerer Dienstmanagementwerkzeuge ist nicht geplant. Außerdem sollte das Monitoring flexibel auf Änderungen in der Dienstimplementierung reagieren können. Im besten Fall sollten Änderungen in der Dienstbeschreibung zur automatischen Neukonfiguration der Überwachungswerkzeuge führen und so die im GRID-Umfeld auftretende Dynamik adressiert werden.

Obige Herausforderungen verdeutlichen, dass bei der Spezifikation von Dienstmanagementinformation insbesondere Beziehungen zwischen

dem Dienst und den dienstrealisierenden Komponenten, vertraglichen Vereinbarungen und Kunden berücksichtigt werden müssen. Entsprechend findet sich der Aspekt auch in dem nachfolgenden Anforderungskatalog, der eine strukturierte Beschreibung der identifizierten Herausforderungen vermittelt.

2.3 Entwicklung des Anforderungskatalogs

Nachdem im letzten Abschnitt die Anforderungen an eine Managementinformationsbasis anhand von zwei Szenarien aufgezeigt wurden, folgt nun eine Konsolidierung dieser Anforderungen. Das Ergebnis dieses Schrittes stellt ein strukturierter Anforderungskatalog dar, der in Kapitel 3 zur Bewertung bestehender Ansätze Verwendung findet. Zunächst werden allgemeine Anforderungen identifiziert und, darauf basierend, spezifische Anforderungen bzgl. der Definition konkreter Dienstmanagementinformation, der Aggregation von Managementinformationen und zuletzt der Umsetzung einer Dienstmanagementinformationsbasis vorgestellt.

2.3.1 Allgemeine Anforderungen

Die vorliegende Arbeit setzt sich mit den Auswirkungen von Dienstorientierungsaspekten auf Modellierung von Managementinformationen auseinander. Entsprechend müssen folgende allgemeine Anforderungen Berücksichtigung finden:

ALL 1 *Dienstorientierung*
Entscheidend für eine Dienstmanagementinformationsbasis ist der Fokus auf höherwertige Dienste (siehe Kapitel 1.1). Mit dieser Forderung wird vor allem der Tatsache Rechung getragen, dass der Begriff Dienst oftmals synonym für einfache, technische Dienste verwendet wird, deren Betrachtung allerdings aus Sicht des Dienstmanagements nicht adäquat ist.

ALL 2 *Berücksichtigung des Dienstlebenszyklus*
Eine Dienstmanagementinformationsbasis muss den Dienstlebenszyklus dahingehend berücksichtigen, dass sie für in den Teilphasen auftretende Managementaufgaben notwendige Informationen bereitstellt. Wie bereits in Abschnitt 2.2.1 verdeutlicht wurde, erscheint im Gegensatz dazu eine ausschließlich an der Betriebsphase orientierte Betrachtung als nicht ausreichend.

2.3.2 Modelleigenschaften

Wie in Abschnitt 2.1 verdeutlicht wurde, bilden Informations- und Datenmodelle einen grundlegenden Bestandteil in der Modellierung von Managementinformationen. Entsprechend sollte das einer Dienstmanagementinformationsbasis unterliegende Modell folgenden Anforderungen gerecht werden:

MOD 1 *Einfache Anwendbarkeit*
Akkzeptanz und Nutzen eines Modells sind insbesondere davon abhängig, wie einfach sich die Anwendbarkeit gestaltet. Gleichermaßen wirkt sich ein zu hoher Komplexitätsgrad des Modells erschwerend auf den Entwicklungsaufwand und die Pflege der Modellinhalte aus.

MOD 2 *Erweiterbarkeit*
Die Möglichkeit, ein bestehendes Modell auf die eigenen Bedürfnisse anzupassen und zu erweitern (Customizing), nimmt eine Schlüsselrolle bei der Anwendung des Modells ein. Durch Erweiterungsmechanismen muss dementsprechend sichergestellt sein, dass Anpassungen einfach durchgeführt werden können und gleichzeitig die Grundstruktur des Modells erhalten bleibt.

MOD 3 *Ausdrucksmächtigkeit*
Die Modellierungseigenschaften des Modells müssen ausdrucksstark genug sein, um für das Dienstmanagement notwendige Sachverhalte zu erfassen. Darunter fallen beispielsweise Ausdrucksmöglichkeiten für Rollenmodelle, Vererbungs- und Enthaltenseinshierachien, aber auch Filtermöglichkeiten (Scoping) und Aliasbildung.

MOD 4 *Flexibilität*
Das Modell muss einen hohen Grad an Flexibilität aufweisen, um eine Verwendung für ein möglichst breites Spektrum an Szenarien und Diensten zu ermöglichen. Dieses Spektrum beinhaltet sowohl Individual- als auch Massendienste, ebenso wie einfache Transportdienste und höherwertige Anwendungsdienste. Gleichzeitig soll es möglich sein, neuartige Dienste, die augenblicklich erst noch im Entstehen begriffen sind, durch das Modell zu erfassen.

MOD 5 *Modularität*
Das Modell muss eine Wiederverwendung von immer wiederkehrenden oder allgemeingültigen Modellteilen ermöglichen und sollte deshalb einem modularen Aufbau folgen.

2.3.3 Definition konkreter Dienstmanagementinformation

In Abschnitt 2.1.3 wurden bereits verschiedene Möglichkeiten der Definition konkreter Managementinformation genannt. Während die komplette Eigenrealisierung eher die Ausnahme darstellt, tritt in der Praxis oftmals eine Mischform auf. Basierend auf standardisierten Objektkatalogen wird eine Verfeinerung und Konkretisierung im Hinblick auf ein spezifisches Szenario vorgenommen.

Die im Folgenden diskutierten Anforderungen an die Beschaffenheit konkreter Managementinformation dienen deshalb gleichermaßen zur Bewertung bestehender Kataloge als auch vom Provider selbst definierter Informationen. Weiterhin werden zwei wichtige Aspekte konkreter Managementinformation, die Beschreibung von Abhängigkeitsbeziehungen und die Aggregation komponentenorientierter Managementinformationen, berücksichtigt.

KMI 1 *Allgemeingültigkeit*
Um eine Anwendung in einem heterogenen Umfeld zu ermöglichen, darf sich konkrete Managementinformation nicht auf technische Implementierungen und individuelle Spezifika von Diensten und Managementwerkzeugen beschränken, sondern muss vielmehr auf eine Vielzahl unterschiedlicher Dienste anwendbar sein.

Nur dieses Maß an Allgemeingültigkeit erlaubt eine einfache szenariospezische Verfeinerung und Abbildung auf existierende Managementarchitekturen und Werkzeuge.

KMI 2 *Ausrichtung an Geschäftszielen und -prozessen*
Um eine Verzahnung von Dienstmanagement und Geschäftszielen zu erreichen, muss sich die Definition konkreter Managementinformation an dem Informationsbedarf von IT Service Providern ausrichten (*Top Down Ansatz*). Neben rein technischen Größen wird es somit notwendig, Geschäftskennzahlen (KPIs) und vertragliche Gesichtspunkte in die Betrachtung mitaufzunehmen.

KMI 3 *Vollständigkeit*
Konkreter Managementinformation darf sich nicht auf bestimmte Managementbereiche beschränken, sondern sollte dahingehend vollständig sein, dass die verschiedenen Aufgabengebiete in der Dienstbereitstellung (z.B. Erkennen von Fehlern, Umsetzung der Dienstqualität) durch entsprechende Objekts- und Attributsdefinitionen unterstützt werden.

KMI 4 *Einheitlichkeit*
Konkrete Managementinformation muss einheitlich sein bezüglich der verwendeten Terminologie und Modellbildung. Insbesondere sollen Sachverhalte nach einem einheitlichen Schema modelliert werden, Redundanzen vermieden und stattdessen Querbezüge zwischen Modellbereichen hergestellt werden.

KMI 5 *Nachvollziehbarkeit*
Voraussetzung für eine einfache Anpassbarkeit von konkreter Managementinformation bildet eine umfangreiche Dokumentation, die eine einfache Nachvollziehbarkeit der Modellinhalte erlaubt. Explizit dokumentiert werden muss dabei die zur Definition konkreter Modellinhalte verwendete Methodik.

KMI 6 *Kundenspezifische Aufbereitung*
Zur Erfüllung von Dienstmanagementaufgaben müssen oftmals kundenspezifische Einflussgrößen (SLAs, QoS Paramter) berücksichtigt werden. Die Möglichkeit, kundenorientierte Informationen mit einem Dienst zu verknüpfen, stellt somit eine wichtige Prämisse für umfassendes Dienstmanagement dar und muss durch konkrete Managemeninformation entsprechend unterstützt werden.

KMI 7 *Berücksichtigung interorganisationaler Aspekte*
Wie in der Szenariobeschreibung verdeutlicht wurde, überschreitet Dienstmanagement oftmals organisatorische Grenzen. Dementsprechend müssen interorganisationale Gesichtspunkte in der Definition konkreter Managementinformation berücksichtigt werden.

2.3.3.1 Beschreibung von Abhängigkeitsbeziehungen

In den Szenariobeschreibungen wurde mehrfach auf die Bedeutung von Abhängigkeitsbeziehungen zwischen Diensten und Komponenten hingewiesen. Entsprechend wird es als Aufgabe einer Dienstmanagementinformationsbasis angesehen, Abhängigkeiten umfassend zu beschreiben. Dabei müssen folgende Aspekte Berücksichtigung finden:

DEP 1 *Differenziertheit*
Die Abhängigkeitsbeschreibung hat die Aufgabe verschiedene Arten von Abhängigkeiten darstellbar und unterscheidbar zu machen. Dies umfasst sowohl Abhängigkeiten zwischen Diensten als auch Abhängigkeiten zwischen Komponenten und Diensten. Weiterhin müssen Kritikalität und Gewichtung von Abhängigkeiten berücksichtigt und eine Differenzierung zwischen funktionalen und organisatorischen Abhängigkeiten ermöglicht werden.

DEP 2 *Abhängigkeiten zwischen Kunden, Nutzern und SLAs*
Um einen Betrieb des Dienstes in Einklang mit vetraglichen Vereinbarungen zu ermöglichen, müssen ferner Bezüge zwischen Kunden, Nutzern und SLAs hergestellt werden. Entsprechend sollte dieser spezielle Fall von Abhängigkeitsbeziehungen in einer

Dienstmanagementinformationsbasis erfasst und somit durch Managementwerkzeuge nachverfolgbar gemacht werden.

Insbesondere in der Betrachtung funktionaler Abhängigkeiten zeigt sich, dass diese einer gewissen Dynamik unterworfen sind. Sie treten oftmals nur kurzzeitig auf und sind an eine bestimmte Funktionalität gebunden, wie z.B. bei der Nutzung eines DNS-Servers zur Namensauflösung. Die Berücksichtigung dieser Aspekte stellt somit eine weitere Anforderung in der Abhängigkeitsbeschreibung dar:

DEP 3 *Berücksichtigung dynamischer Aspekte*
Zur Beschreibung dynamischer Gesichtspunkte müssen Abhängigkeitsinformationen mit zeitlichen und funktionalen Aspekten versehen werden. Somit soll nachvollzogen werden können, ob eine Abhängigkeit zu einem gegebenen Zeitpunkt besteht, was vor allem die Fehlersuche erleichtert. Wichtig ist allerdings hierbei, einen geeigneten Kompromiss zwischen feingranularer Abhängigkeitsbeschreibung und der Übersichtlichkeit bzw. dem entstehenden Pflegeaufwand zu finden.

2.3.3.2 Aggregation komponentenorientierter Managementinformation

Wie insbesondere aus der Beschreibung des GRID-Szenarios hervorgeht, verkörpert das Wissen, in welcher Weise Diensteigenschaften bzw. -attribute durch Parameter der zur Diensterbringung verwendeten Komponenten beeinflusst werden, eine wichtige Prämisse zur Erfüllung von Dienstmanagementaufgaben dar. Demzufolge muss eine Dienstmanagementinformationsbasis Konzepte zur Verfügung stellen, die eine Verdichtung von Komponentenparametern unterstützen und somit Aggregationsbeziehungen explizit darstellbar machen.

AGG 1 *Berechnung von Dienstattributen*
Im Hinblick auf eine Abbildung zwischen komponenten- und dienstorientierter Managementinformation müssen Aggregationsvorschriften geschaffen werden. Letztere sollen festlegen, wie Dientattribute aus Komponentenparamtern berechnet werden können.

AGG 2 *Deklarativität*
Um Implementierungsunabhängigkeit zu gewährleisten, sollte die Spezifikation von Aggregationen deklarativ erfolgen. Dies bedeutet vor allem, dass für die auszuführende Verdichtung von Komponentenparametern keine Berechnungsabläufe (z.B. in Form konkreter Algorithmen) angegeben werden, sondern vielmehr das gewünschte Berechnungsergebnis beschrieben wird.

AGG 3 *Formalisierungsgrad*
Ein hinreichender Formalisierungsgrad bei der Beschreibung von Aggregationsbeziehungen ist notwendig, um eine computergestützte und automatisierte Verarbeitung zu ermöglichen. Von entscheidender Bedeutung ist hierbei ein geeigneter Kompromiss zwischen einfach les- und editierbarer Beschreibung (z.B. reiner Text) und einem für maschinelle Verarbeitung geeignetem Formalismus.

AGG 4 *Unterstützung der Dienstüberwachung*
Die Kenntnis von Aggregationsbeziehungen ist insbesondere auch für die Überwachung (*Monitoring*) von Diensten relevant. Entsprechend muss die Beschreibung dieser Beziehungen in einem Format vorliegen, das eine einfache Wiederverwendung der Informationen für Monitoring-Werkzeuge erlaubt.

2.3.4 Umsetzbarkeitsaspekte

Neben den konzeptionellen Eigenschaften eines Modells ist es für IT-Service Provider vor allem von Belang, wie einfach es in der Praxis umsetzbar und in bestehende Managementsysteme integrierbar ist. Im Folgenden werden deshalb Aspekte diskutiert, die in der Realisierung und dem täglichen Einsatz einer Dienstmanagementinformationsbasis auftreten.

REA 1 *Integrationsmöglichkeiten*
In der Regel setzen Service Provider für den Betrieb ihrer IT-Infrastruktur bereits komponentenorientierte Managementwerkzeuge, respektive Managementmodelle ein. Diese Managementan-

wendungen vollständig durch Dienstmanagementwerkzeuge zu ersetzen, ist mit hohem finanziellen Aufwand verbunden und kurzfristig schwer realisierbar. Stattdessen müssen dienstorientierte Managementwerkzeuge einfach in die bestehende Managementlandschaft integriert werden können und gegebenfalls eine Migration ermöglichen. Hinsichtlich einer Dienstmanagementinformationsbasis bedeutet dies vor allem, dass Mechanismen vorhanden sein müssen, die eine Integration in bestehende Managementmodelle erlauben.

REA 2 *Einfache Nutzung durch Managementanwendungen*
Um eine Dienstmanagementinformationsbasis durch Managementanwendungen nutzbar zu machen, ist das Vorhandensein entsprechender Protokolle erforderlich, die einen Informationsaustausch anhand von Zugriffs- und Übertragungsmechanismen steuern.

REA 3 *Pflege der Inhalte*
Ein erheblicher Aufwand bei der Umsetzung eines Modells fällt der Erstellung und Pflege der Modellinhalte zu. Zur Vereinfachung dieser Arbeitsschritte benötigt man eine adäquate Unterstützung in Form von Modellierungswerkzeugen.

REA 3 *Unterstützung durch Managementagenten*
Die inhärent hohe Dynamik im Dienstmanagementumfeld erfordert eine schnelle Reaktion auf auftretende Ereignisse (z.B. Verletzung eines SLAs). Entsprechend muss eine Dienstmanagementinformationsbasis eine möglichst aktuelle Sicht auf Diensteigenschaften reflektieren bzw. müssen Managementagenten vorhanden sein, die eine automatische Aktualisierung der Modellinhalte gewährleisten. In diesem Zusammenhang wäre ebenfalls eine automatische Erkennung von Diensten wünschenswert (*auto discovery*).

REA 5 *Reifegrad/Verbreitung*
In besonderem Maße wirkt sich der Reifegrad eines Modells auf die Einfachheit und Nachhaltigkeit einer praktischen Umsetzung

aus. Kriterien dafür sind der Stand des Standardisierungsprozesses oder die Verbreitung in Form von konkreten Produkten.

2.3.5 Zusammenfassung und Vorstellung des Anforderungskatalogs

Die in den vorhergehenden Abschnitten dargelegten Anforderungen werden im Folgenden in tabellarischer Form zusammengefasst. Der so entstehende Anforderungskatalog findet in den weiteren Kapiteln dieser Arbeit Verwendung. Einerseits dient er als Bewertungsinstrument für existierende Ansätze in Kapitel 3, andererseits fließt er in die Entwicklung eines neuen Lösungsansatzes in Kapitel 4 mit ein.

Anforderungskatalog	
Allgemeine Anforderungen	
ALL 1	Dienstorientierung
ALL 2	Berücksichtigung des Dienstlebenszyklus
Anforderungen an die Modelleigenschaften	
MOD 1	Einfache Anwendbarkeit
MOD 2	Erweiterbarkeit
MOD 3	Ausdrucksmächtigkeit
MOD 4	Flexibilität
MOD 5	Modularität
Definition konkreter Managementinformation	
Beschaffenheit	
KMI 1	Allgemeingültigkeit
KMI 2	Ausrichtung an Geschäftszielen und -prozessen
KMI 3	Vollständigkeit
KMI 4	Einheitlichkeit
KMI 5	Nachvollziehbarkeit
KMI 6	Kundenspezifische Aufbereitung
KMI 7	Berücksichtigung interorganisationaler Aspekte

Anforderungskatalog (*Fortsetzung*)	
Beschreibung von Abhängigkeitsbeziehungen	
DEP 1	Differenziertheit
DEP 2	Abhängigkeiten zwischen Kunden, Nutzern und SLAs
DEP 3	Berücksichtigung dynamischer Aspekte
Aggregation komponentenorientierter Managementinformation	
AGG 1	Berechnung von Dienstattributen
AGG 2	Deklarativität
AGG 3	Formalisierungsgrad
AGG 4	Unterstützung der Dienstüberwachung
Umsetzbarkeitsaspekte	
REA 1	Integrationsmöglichkeiten
REA 2	Einfache Nutzung durch Managementanwendungen
REA 3	Pflege der Inhalte
REA 3	Unterstützung durch Managementagenten
REA 5	Reifegrad/Verbreitung

Tabelle 2.1: Anforderungskatalog

Kapitel 3

Analyse bestehender Ansätze

Nachdem mit der im letzten Kapitel durchgeführten Analyse grundlegende Anforderungen an eine Dienstmanagementinformationsbasis ermittelt werden konnten, werden nun Arbeiten von Standardisierungs- und Industriegremien, Forschungsansätze und existierende Produkte begutachtet. Durch einen Vergleich mit dem Anforderungskatalog können so Aussagen darüber getroffen werden, ob und in welchem Maße bestehende Ansätze den identifizierten Anforderungen gerecht werden.

In die Auswahl betrachteter Arbeiten fließen dabei neben standardisierten Managementmodellen auch Ansätze zur Beschreibung von Diensten und zur Aggregation von Komponentenparametern sowie Prozessrahmenwerke mit ein. An die Vorstellung der Arbeiten schließt sich jeweils eine kurze Bewertung mit an, die gesammelten Ergebnisse der Evaluation finden sich am Ende dieses Kapitels.

3.1 Arbeiten von Standardisierungsgremien

In den nachfolgenden Abschnitten werden für die vorliegende Arbeit relevante Spezifikationen internationaler Standardisierungs- und Indus-

triegremien vorgestellt und bewertet. Dies umfasst Arbeiten der *Distributed Management Task Force (DMTF)*, des *TeleManagementForums (tmforum)* und der *Internet Engineering Task Force (IETF)*.

3.1.1 Distributed Management Task Force

Die 1992 gegründete DMTF verkörpert ein Industriegremium führender Hersteller von Netzwerkprodukten und Managementlösungen und beschäftigt sich mit der Entwicklung und Verbreitung interoperabler Managementstandards. Den Mittelpunkt dieser Initiativen bildet die Erstellung eines Modells für Managementinformationen, welches den Namen *Common Information Model (CIM)* [CIM06, CIM03] trägt und erstmals 1996 vorgestellt wurde.

Überblick

CIM umfasst vier Ebenen

CIM wurde als objektorientiertes Modell konzipiert und zielt darauf ab, im Netz-, System- und Dienstmanagement benötigte Informationen umfassend und implementierungsneutral zu repräsentieren. Es liegt sowohl in Form von Klassendiagrammen als auch in einer textuellen Repräsentation, dem *Managed Object Format (MOF)* vor. Um eine einfache Erweiterbarkeit zu gewährleisten wurde die Spezifikation von CIM in vier Ebenen gegliedert:

- *Metamodel*
  Das *Metamodel* bestimmt die grundlegende syntaktische Struktur von CIM und legt somit die Darstellungsform von Modellelementen (z.B. Klassen, Attribute, Assoziationen) fest. Dabei wurde vorwiegend eine Erweiterung des UML-Metamodells [OMG04] hinsichtlich CIM-spezifischer Assoziationen (*Weak Assocations*) vorgenommen. Weiterhin wurde strikte Vererbung eingeführt, d.h. Methoden und Attribute von Klassen können von der erbenden Klasse nicht überschrieben werden.

- *Core Model*
  Die Intention des *Core Models* besteht darin, grundlegende Klassen zu definieren, die gemeinsame Eigenschaften aller Elemente

des CIM Modells repräsentieren. Darin enthaltene Elemente (z.B. *ManagedElement*) bilden somit die oberste Ebene der CIM Vererbungshierachie und fungieren als Superklassen für *Common Model* und *Extension Schema*.

- *Common Model*
  Auf der Ebene des *Common Models* wurde eine Unterteilung in Managementbereiche vorgenommen. Diese umfasst gegenwärtig die Bereiche *Systems*, *Database*, *Device*, *Event*, *Interop*, *IPSecPolicy*, *Metrics*, *Network*, *Physical*, *Policy*, *Security*, *Support*, *System* und *User*. In diesen Teilmodellen definierte Klassen repräsentieren gemeinsame Konzepte des jeweiligen Managementbereichs, wurden aber wiederum technologie- und implementierungsneutral konzipiert.

- *Extension schema*
  Die unterste Ebene der CIM Spezifikation bilden technologiespezifische Erweiterungen des *Common Models*. In dem Extension Schema enthaltene Klassen spiegeln somit beispielsweise Spezifika des verwendeten Betriebssystems (UNIX, Windows ...) wider.

Für das Dienstmanagement relevante Modelle wurden als Teile des *Core Models* in CIM integriert (Abbildung 3.1). Die Klasse *Service* repräsentiert hierbei eine Spezialisierung des *EnabledLogicalElements* und wird über eine *weak reference* mit einem *System* in Beziehung gesetzt. Als Konsequenz dieser Modellierung kann ein Dienst nur zusammen mit genau einem System (z.B. C*IM_ApplicationSystem* oder *CIM_ComputerSystem*) auftreten.

Vielfätige Assoziationsbeziehungen

Weiterhin bestehen Assoziationsbeziehungen zwischen den Klassen *Service*, *ServiceAccessPoint* und *ManagedElement*. Auf diese Weise wird es ermöglicht, den Dienstzugriff über den SAP (*ServiceAccessBySAP*) und funktionale Abhängigkeiten zwischen Diensten und anderen CIM-Elementen (*ServiceAvailableToElement*, *ServiceAffectsElement*) zu modellieren. Ebenfalls wurde mit der Assoziation *ServiceServiceDependency* eine Beschreibungsmöglichkeit für Abhängigkeiten zwischen Diensten geschaffen. In Bezug auf Attribute der Klasse *Service* wird momentan nur eine sehr rudimentäre Auswahl angeboten: Neben dem Namen des Dienstes und den Kontaktdaten des Dienstverantwortlichen

(*PrimaryOwner*) umfasst dies lediglich Informationen zur Aktivierung des Dienstes (*StartMode*, *Started*). Desweiteren bietet das *CIM_Metrics* Modell prinzipiell die Möglichkeit, Dienstattribute auszudrücken, aber auch in diesem Zusammenhang wurden bisher keine konkreten Definitionen vorgenommen.

Einbindung in WBEM-Architektur

Implementierung von CIM Zur Implementierung von CIM wurde von der DMTF eine Eingliederung in das *Web Based Enterprise Management (WBEM)* Rahmenwerk vollzogen. Dies umfasst die Verwendung von XML zur Repräsentation von Klassen, Instanzen und möglichen Operationen [DMT07b] sowie den Transport der dadurch entstehenden XML-Dateien über das http-Protokoll [DMT07a].

Weiterhin folgt der Aufbau der WBEM-Architektur dem Client/Server-Prinzip. Als zentrale Komponente fungiert dabei der *CIM Object Manager (CIMOM)*. Er nimmt Anfragen von Managementapplikationen entgegen, verwaltet ein statisches CIM Schema mit Hilfe eines Repositories und greift zur Bestimmung aktueller Werte für Managementinformationen auf CIM Provider zu. Letztere stellen eine aktuelle Sicht auf die zu managenden Ressourcen durch geeignete Instrumentierungen sicher.

Bewertung

Aufgrund der Beteiligung führender Hersteller haben die von der DMTF verabschiedeten Standards zu einer gewissen Verbreitung und Verfügbarkeit WBEM- und CIM-konformer Implementierung geführt. Weiterhin stellt CIM insbesondere im Bereich Netz- und Systemmanagement detaillierte Managementinformation zur Verfügung.

Mangelnde Übersichtlichkeit

Allerdings wirkte sich die Zielsetzung, ein möglichst umfangreiches Modell zu schaffen, nachteilig auf dessen Übersichtlichkeit aus, was auch in [AK01] konstatiert wird: Bei über 1000 Klassen in der momentanen Version von CIM gestaltet es sich als äußerst schwierig, alle Bezüge zwischen Klassen nachzuvollziehen; alleine das oberste Element der Vererbungshierarchie, die Klasse *ManagedElement*, verfügt über mehr als 20 Assoziationen, die entsprechend an alle verbleibenden Klassen

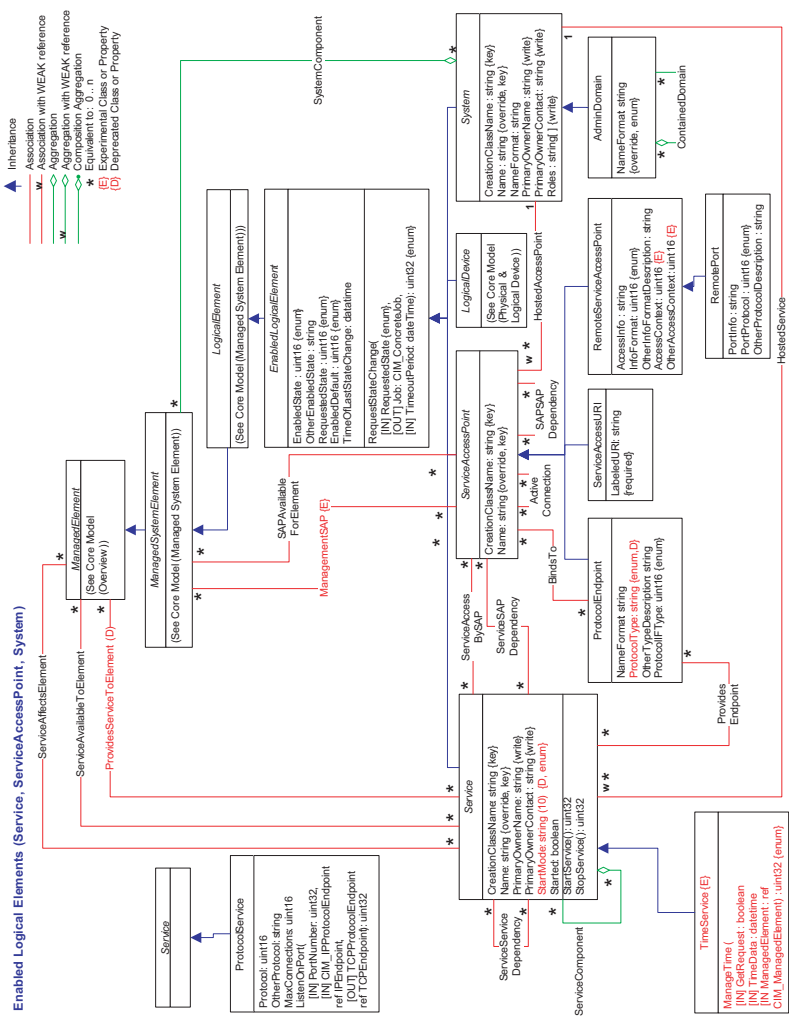

Abbildung 3.1: Das Dienstmodell des CIM Core Models

des Modells weitervererbt werden. Zudem verhindern die Erweiterungen des UML-Metamodells die Verwendung UML-konformer Bearbeitungswerkzeuge, was die Handhabbarkeit von CIM einschränkt[Str04].

Kein Fokus auf höherwertige Dienste

Weiterhin wird zwar von der DMTF eine Eignung für die Darstellung von Dienstmanagementkonzepten proklamiert, kann aber zum jetzigen Zeitpunkt als nicht gegeben betrachtet werden. Negativ wirkt sich hierbei vor allem das Fehlen grundlegender Elemente (z.B. SLA) und die inflexible Modellierung der Klasse *Service* aus. Deren untrennbare Verbindung mit einem *System* erschwert erheblich die Darstellung höherwertiger Dienste, wie z.B. den in Abschnitt 2.2.1 vorgestellten Web-Hosting Dienst. Zudem wurden bisher nur sehr rudimentäre Dienstattribute definiert, was den praktischen Einsatz für das Dienstmanagement weiterhin einschränkt.

3.1.2 TeleManagement Forum

Eine weitreichende Standardisierung und Integration von Managementlösung für die Telekommunikationsbranche zu schaffen, stellt das erklärte Ziel des *TeleManagement Forums (tmforum)* dar, eine über 500 Mitglieder umfassender Vereinigung von Dienstbetreibern und Herstellern von Managementsoftware und -komponenten. Den Mittelpunkt dieser Standardisierungsbemühungen bildet die *Next Generation Operations Systems and Software (NGOSS)* Initiative.

NGOSS stellt ein umfassendes Rahmenwerk zur Entwicklung und operationalen Einbindung von *Operation* und *Business Support Systems (OSS/BSS)* dar. Dies umfasst eine als *Enhanced Telecom Operations Map (eTOM)[GB902]* bezeichnete Sammlung standardisierter Managementprozesse, ein herstellerübergreifendes Informationsmodell (*Shared Data/Information Model*) [GB904a], eine modulare Softwarearchitektur (*Technology Neutral Architecture*) sowie Konformitätstests (*Compliance Tests*).

eTOM und SID werden betrachtet

Für die vorliegende Arbeit sind unter diesen Bestandteilen von NGOSS insbesondere das Prozessrahmenwerk und das Informationsmodell von Relevanz: Zunächst wird in Zusammenhang mit der Anforderung *Einbettung in Managementprozesse* (KMI 2) untersucht, ob die in eTOM

ausgeführten Managementprozesse Vorgaben hinsichtlich der Beschaffenheit konkreter Dienstmanagementinformation geben. Daran anschließend die wird das *Shared Data/Information Model* bezüglich dessen Eignung für eine Dienstmanagementinformationsbasis evaluiert und eine Bewertung beider Ansätze vorgenommen.

3.1.2.1 Enhanced Telecom Operations Map

Mit eTOM stellt das *tmforum* eine Sammlung von Referenzprozessen zur Verfügung, die Providern von Telekommunikationsdiensten als Grundlage zur weiteren unternehmensspezifischen Verfeinerung und Adaption dienen sollen. Der Fokus liegt hierbei auf der Verbesserung organisatorischer Abläufe, Implementierungsaspekte werden nur rudimentär tangiert.

Strukturierung entlang von Sichten und Bereichen

Die eTOM wurde als kohärentes Rahmenwerk konzipiert, in dem Prozesse entlang verschiedener Sichten und Bereiche strukturiert werden. Dabei wird zwischen fünf Detaillierungsstufen (*Levels*) für Prozesse unterschieden: Die in Level 0 definierten grundlegenden Unternehmensprozesse werden schrittweise weiter unterteilt und verfeinert und erreichen in Level 4 den vollen Detailgrad.

In Level 0 sichtbare Prozesse, Rollen und Bereiche werden in Abbildung 3.2 dargestellt. Im Hinblick auf das Rollenmodell liegt der Fokus hierbei auf dem *Customer*, ergänzend wurden die Rollen *Suppliers/Partner*, *Shareholders*, *Employees* und *Other Stakeholders* definiert. Weiterhin schuf man die Prozessbereiche *Operations*, *Strategy, Infrastructure & Product (SIP)* und *Enterprise Management*. Während Ersterer sich mit der Bereitstellung und dem Betrieb von Diensten beschäftigt, dienen die im Bereich *SIP* angesiedelten Prozesse vorwiegend der Planung und Vermarktung neuer Dienstklassen. Der Bereich Enterprise Management adressiert schließlich allgemeine betriebswirtschaftliche Fragestellungen. Weiterhin wurde für die Bereiche SIP und Operations eine vertikale Einteilung der Prozesse in *Functional Areas* vorgenommen, die nach den in ihnen hauptsächlich gemanagenden Objekten benannt wurden [Bre04].

Im Hinblick auf eine Dienstmanagementinformationsbasis sind dabei potentiell die *Operations*-Prozesse in der Kategorie *Service* interessant.

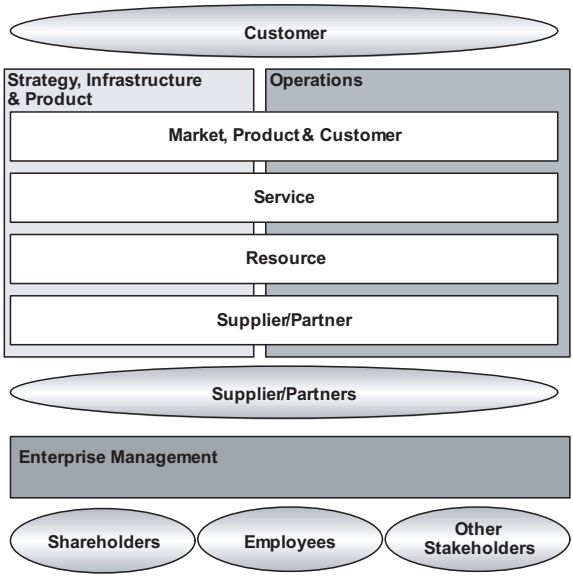

Abbildung 3.2: eTOM Level 0 Sicht aus [GB902]

Sie legen Arbeitsabläufe für beispielsweise die Behandlung von Dienstfehlern (*Problem Handling*) fest und dokumentieren somit Dienstmanagementaufgaben, deren Unterstützung zu den Aufgabengebieten einer Dienstmanagementinformationsbasis zählt (vgl. Anforderung KMI 2).

3.1.2.2 Shared Information/Data Model

Das *Shared Information/Data Model* wurde mit dem Ziel entwickelt, eine gemeinsame Datenbasis für OSS/BSS-Anwendungen zu schaffen und damit deren herstellerübergreifende Integration zu erleichtern. Es stellt damit einen wichtigen Teil der NGOSS *Knowledge Base* dar und erhebt den Anspruch, zur Ausführung von eTOM-Prozessen benötigte Managementinformationen zu verkörpern.

SID verkörpert vier Sichten

SID verfolgt einen objektorientierten Ansatz zur Modellierung von Managementobjekten und Interaktionen und setzt sich aus verschiedenen Sichten (*Views*) zusammen, die ihrerseits wiederum in Domänen und Subdomänen organisiert sind:

- *Business View*
  Diese Sicht vermittelt eine geschäftsorientierte Perspektive auf das Managementumfeld eines Providers. Insbesondere sollen im *Business View* definierte Entitäten[1], Attribute und Relationen dabei den Informationsbedarf der eTOM-Prozesse abdecken.
  Um diesen Zusammenhang zu verdeutlichen, wurde eine ähnliche Strukturierung vorgenommen: Der *Business View* setzt sich aus Domänen (*Business Domains*) zusammen, die den in eTOM Level 0 definierten Konzepten entsprechen. In der gegenwärtigen Version umfasst dies die Domänen *Customer*, *Product*, *Service*, *Resource* und sogenannte *Common Business Entities* (z.B. *Party*, *Location*, *Business Interaction*). Als weitere Unterteilung für Domänen wurden *Aggregate Business Entities (ABEs)* eingeführt, die eine Sammlung thematisch zusammenhängender Entitäten repräsentieren (die Domäne *Service* besteht z.B. aus *Service Test*, *Service Trouble*, usw.).

- *SystemView*
  Mit dem *System View* wird das aus Geschäftssicht definierte Modell weiter verfeinert und damit ein Detailgrad geschaffen, der den Ansprüchen von OSS/BSS-Systementwicklern gerecht werden soll. Dafür wurden neue Entitäten eingeführt, vorhandene Entitäten mit zusätzlichen Attributen erweitert und Assozationsbeziehungen detailliert (vorwiegend durch Verwendung von Assoziationsklassen). Die im *Business View* bereits vorgestellte Aufteilung wurde beibehalten, wobei allerdings die Namensgebung angepasst (*System Domains* bzw. *Aggregate System Entities*) und zusätzlich neue Domänen und ASEs geschaffen wurden.

- *ImplementationView und Run-Time View*
  Als weitere Verfeinerungsstufen sieht SID den *Implementation*

[1] Entitäten in SDI sind synonym zu Klassen in objektorientierter Terminologie

bzw. *Run-Time View* vor. Während Ersterer sich mit Implementierungsaspekten des *System Views* auseinandersetzen soll, zielt der *Run-Time View* auf die aktive Überwachung eines NGOSS-konformen Systems ab. Allerdings wurden beide Sichten zum gegenwärtigen Zeitpunkt noch nicht spezifiziert, so dass konkretere Aussagen über den Inhalte nicht möglich sind.

Die Spezifikation der einzelnen Sichten steht dabei in Form eines Überblicksdokuments und mehrerer Addenda zur Verfügung, die eine Sammlung von UML-Diagrammen und deren Beschreibung repräsentieren. Im Hinblick auf die Konzeption einer Dienstmanagementinformationsbasis ist dabei insbesondere das Addendum *Service* des *System Views* von Interesse und wird deshalb im Folgenden weiter ausgeführt. Die Modellierung weist dabei Ähnlichkeiten mit DEN-ng [Str02] auf, das gewissermaßen als Vorgänger von SID betrachtet werden kann.

Assoziation zwischen Service und Product

Grundsätzlich wurde auf eine Kopplung von Diensten und Produkten geachtet, die dadurch begründet wird, dass Dienste letztendlich immer in Form eines Produkts bzw. eines Produktangebots an Kunden vermarktet und verkauft werden. Dienste, die direkt in einem Produkt auftreten, werden dabei *Customer-Facing Services (CFS)* genannt. Sie werden unterstützt durch sogenannte *Resource-Facing Service*s *(RFS)*, wobei diese nicht unmittelbar gegenüber dem Kunden sichtbar sind. In dem SID-Dienstmodell (Abbildung 3.3) werden die genannten Sachverhalte durch eine Spezialisierung der Klasse *Service* bzw. der Aggregation *ProductHasCustomerFacingService* zwischen einem *RFS* und einem *Product* ausgedrückt. Die Modellierung von Komponenten erfolgt in SID durch die Klasse *Resource* bzw. deren Spezialisierung zu *Logical* und *Physical Resources*. Beispielsweise werden physische Bestandteile eines Routers der *Physical Resource* zugeordnet, wohingegen das Betriebssystem des Routers eine *Logical Resource* darstellt. Die Aggregationen *PhysicalResourcesHostRFS* und *LogicalResourcesImplementRFS* drücken dabei funktionale Abhängigkeiten zwischen einem *RFS* und einer physischen bzw. logischen Ressource aus.

Wenige Dienstattribute verfügbar

Neben den in Abbildung 3.3 aufgezeigten Bestandteilen umfasst SID eine Reihe weiterer Entitäten (z.B. *ServiceCharacteristics*), auf die (aus Platzgründen) nicht weiter eingegangen werden soll. Es bleibt jedoch

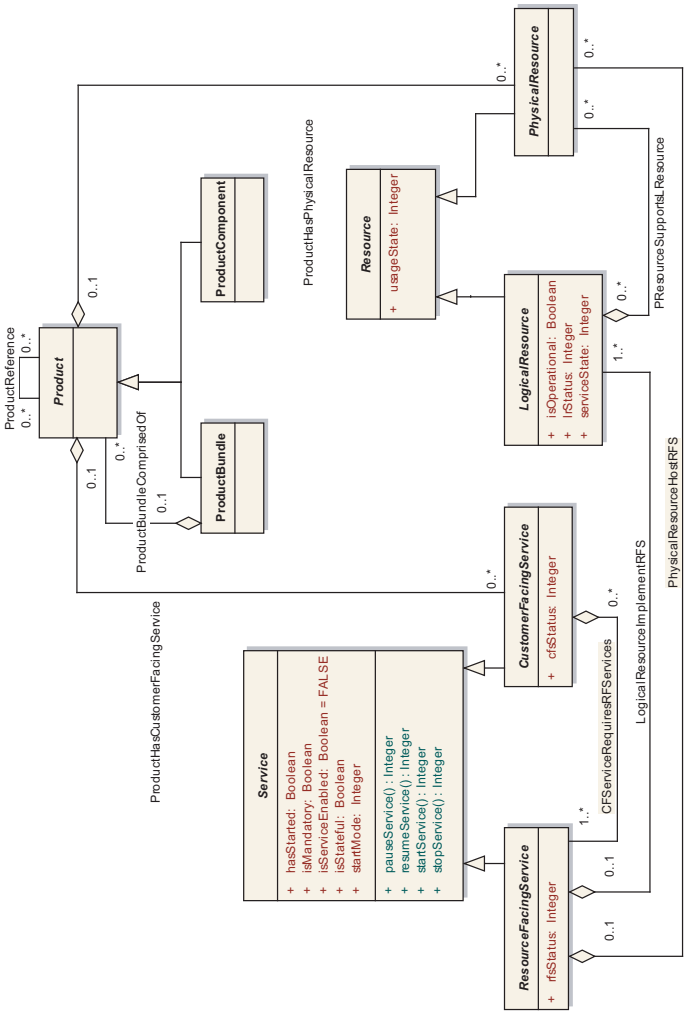

Abbildung 3.3: Dienstmodell in SID

anzumerken, dass zwar in SID eine umfassende Sammlung dienstbezogener Entitäten geschaffen wurde, aber bisher noch sehr wenige Dienstattribute verfügbar sind. Im Fall der Klasse *Service* beschränkt sich dies größtenteils, wie in Abbildung 3.3 dargestellt, auf Attribute zum Aktivierungszustand des Dienstes.

Bewertung

Die NGOSS-Iniative des TeleManagement Forums leistet einen wertvollen Beitrag zum integrierten Dienstmanagement und stellt den derzeit umfassendensten Ansatz dar. Die Aufteilung des Rahmenwerks in verschiedene Sichten kann als gelungen angesehen werden und erlaubt einen modularen Aufbau NGOSS-konformer Anwendungen.

Niedrigere Detailtiefe

Bei der Untersuchung von eTOM zeigte sich, dass darin beschriebene Prozesse die Dienstmanagementaufgaben eines Providers geordnet und bereichsübergreifend dokumentieren und somit prinzipiell die Möglichkeit zur Ableitung benötigter Dienstmanagementinformationen bieten. Schwächen zeigen sich lediglich im Rollenmodell [Bre04] und der niedrigeren Detailtiefe im Vergleich zu der in Abschnitt 3.1.4 vorgestellten IT Infrastructure Library.

Nachvollziehbare Dokumentation

Weiterhin wurde mit SID ein strukturierter und durchdachter Modellentwurf geschaffen, der im Vergleich zu anderen Managementmodellen über eine umfangreiche Dokumentation verfügt. Der Bezug zu eTOM-Prozessen wurde sowohl im Aufbau von SID als auch in den spezifizierten Entitäten hergestellt und Designentscheidungen wurden nachvollziehbar argumentiert. Obwohl sich SID sowohl als Informations- als auch als Datenmodell versteht, wurde allerdings die dafür notwendige Abbildung (vgl. Abschnitt 2.1) noch nicht vollzogen. Zwar sind entsprechende Sichten (*Implementation* und *Run-Time View*) vorgesehen, wurden aber auch noch nicht spezifiziert, was eine homogene Implementierung von SID erschwert. Die Modellinhalte betreffend zeigte sich, dass bisher vorwiegend eine Grundstruktur zur Repräsentation von Diensten geschaffen wurde und generische Dienstattribute nur sehr rudimentär definiert wurden. Zudem sind Klassen für konkrete Netz- und Systemkomponenten bisher in SID fast nicht vorhanden; ob eine Liasion [GB904a] mit der DMTF diesen Umstand ändert, bleibt abzuwarten.

NGOSS-Einführung erfordert hohen Aufwand

Insgesamt kann festgestellt werden, dass die Bestrebungen des TeleManagement Forums einen möglichst umfassenden Ansatz zu schaffen, gleichzeitig einen Vor- und Nachteil darstellen. Zwar kann mit Hilfe von NGOSS eine interoperable Lösung geschaffen werden, erfordert aber die Anpassung der gesamten Managementinfrastruktur eines Providers. Dies ist sicherlich, neben der Entscheidung Standards nicht öffentlich zugänglich zu machen, ein Hauptgrund dafür, warum NGOSS bzw. die Bestandteile eTOM und SID bisher nur eine verhältnismäßig geringe Verbreitung erfahren haben.

3.1.3 Internet Engineering Task Force

Eine wichtige Rolle in der gängigen Managementpraxis nehmen von Internet Engineering Task Force (IETF) verabschiedete Spezifikationen ein. Insbesondere die in den achtziger Jahren vorgestellte Internet-Managementarchitektur[2] hat eine große Verbreitung im Rahmen kommerzieller Produkte erfahren und trägt deshalb entscheidend zur herstellerübergreifenden Interoperabilität von Managementlösungen bei. Für die vorliegende Arbeit ist in diesem Zusammenhang das Informationsmodell dieser Architektur von Relevanz, einen umfassenden Überblick zu den weiteren Teilmodellen vermitteln [HAN99, Bla92].

Datenorientierte Modellierung

Das Internet-Informationsmodell (*Structure of Management Information, SMI*) [CMRW96] folgt einem, im Vergleich zu den bisher vorgestellten Ansätzen SID und CIM, einfacheren Modellentwurf: Konkrete Managementinformation liegt in Form von MIB-Definitionen vor, die sich größtenteils aus einer Sammlung von skalaren Variablen zusammensetzen. Zur Strukturierung von MIBs und zur Darstellung von Instanzen eines Typs finden Tabellen Anwendung. Allerdings wurde auf die Verwendung objektorientierter Konzepte verzichtet, so dass weder Kapselungs- noch Vererbungsmechanismen zur Verfügung stehen. Zwar wurde mit dem *Registrierungsbaum* eine Möglichkeit zur global eindeutigen Namensgebung von Managementobjekten geschaffen, eine *Vererbungshierarchie* kann jedoch nicht abgebildet werden.

[2] In Anlehnung an das verwendete Managementprotokoll SNMP [CFSD90] findet sich häufig auch der Begriff SNMP-Management

Seit Bestehen des Internet-Managements wurde eine Vielzahl von – teilweise herstellerspezifischen – MIB-Definitionen geschaffen, wobei allerdings eine klare Ausrichtung auf das Netz- und Systemmanagement konstatiert werden muss. Erst in den letzten Jahren wurde versucht, verstärkt Aspekte des Anwendungs- und Dienstmanagements zu berücksichtigen. Die aus diesen Bemühungen entstandenen MIBs werden im Folgenden kurz dargestellt:

Network Services Monitoring MIB

Fokus: Netzdienste

Der Überwachung spezieller Netzdienste (z.B. Datei-, Druck- oder Verzeichnisdiensten) widmet sich die *Network Services Monitoring MIB* [KF94a]. Sie nimmt eine verbindungszentrierte Managementsicht auf Netzdienste ein und setzt sich aus zwei Tabellen zusammen, die unterschiedliche Detaillierungsstufen repräsentieren: Während die in der *applTable* enthaltenen Variablen allgemeine Informationen zu Kommunikationsbeziehungen des Dienstes (z.B. Anzahl ein- bzw. ausgehender Verbindungen oder Anzahl fehlerhafter Verbindungsaufbauwünsche) reflektieren, werden in der *assocTable* die einzelnen Kommunikationsverbindungen detailliert dargestellt (z.B: durch Beschreibung der Verbindungsdauer oder des verwendeten Protokolls). Zusätzlich wurden als Teil der *applTable* eine Reihe allgemeiner Variablen (z.B. zur Darstellung des operationellen Zustandes des Dienstes) geschaffen.

Um die spezifischen Gegebenheiten bestimmter Klassen von Netzdiensten abzubilden, wurden domänenspezifische Erweiterungen der *Network Services Monitoring MIB* vorgenommen. Dies umfasst u.a. die *Mail Monitoring MIB* [KF94b], *X.500 Directory Monitoring MIB* [MK94] und *Relational Database Management System MIB* [BEP+94].

Application MIB

Service-level view

Die Application Management MIB [KKPS99] versucht eine dienstorientierte (als *service-level view* bezeichnete) Sichtweise auf verteilte Anwendungen zu vermitteln. Mit Hilfe der Tabelle *applServiceGroup* wird dabei ein Bezug zwischen einem Anwendungspaket und dem realisierten

Dienst hergestellt. Des Weiteren wurden zur Beschreibung managementrelevanter Aspekte einer Anwendung Tabellen geschaffen, die Auskunft über die offenen Dateien und Netzwerkverbindungen (z.B. Anzahl der Lese-/Schreibzugriffe) einer Anwendung (*applChannelGroup*) und Zeiten und Häufigkeiten durchgeführter Transaktionen (*applTransactionStreamTable*) geben, sowie einfache Eingriffsmöglichkeiten in den Betrieb der Anwendung (*applElmtRunControlGroup*) gestatten. Da Transaktionen Bezug nehmend auf Elemente der *applChannelGroup* definiert werden, sind allerdings keine benutzerorientierten Aussagen möglich [Hau01].

Definitions of Managed Objects for WWW Services

Verwendung eines abstrakten Protokolls

Als domänenspezifische Erweiterung der *Application Management MIB* auf WWW-Dienste kann die in RFC 2594 vorgestellte *Definitions of Managed Objects for WWW Services* [HKS99] angesehen werden. Eine interessante Neuerung stellt dabei die Verwendung eines *Abstract Document Transfer Protocols (DTD)* zur Beschreibung von Protokollstatistiken dar. Damit wird versucht, eine Abstraktion über Zugriffsprotokolle auf WWW-Dokumente (z.B. HTTP [FGM+97] oder FTP [BBC+71]) zu schaffen und somit eine implementierungsneutrale Beschreibung eines WWW-Dienstes zu ermöglichen.

Die Definition von MIB-Variablen erfolgte dabei mit einer Ausrichtung auf Leistungs- und Fehlermanagementaspekte und wurde in drei Gruppen strukturiert: Während sich die Gruppe *Service information* mit administrativen Aspekten (z.B. Ansprechpartner) eines Dienstes und dem verwendeten Übertragungsprotokoll beschäftigt, enthält die Gruppe *protocol statistics* detaillierte Informationen zu den getätigten Protokollaufrufen (z.B. Anzahl von *requests*). Die Gruppe *document statistics* gibt schließlich Auskunft über die Verteilung von Dokumentzugriffen (z.B. mit einer Liste der zehn am häufigsten angefragten Dokumente).

Expression und Event MIB

Im Hinblick auf die Anforderungskategorie *Aggregation komponentenorientierter Managementinformation* (Abschnitt 2.3.3.2) sind insbeson-

dere die im Folgenden kurz vorgestellten *Expression MIB* [Kav00b] und *Event MIB* [Kav00a] von Interesse. Ausgehend von der Beobachtung, dass oftmals MIB-Objekte benötigt werden, die in der ursprünglichen Definition keine Berücksichtigung erfuhren, aber durch Verknüpfung bestehender Objekte berechnet werden können, wurden Ausdrucksmöglichkeiten für Aggregationsvorschriften geschaffen. Die *Expression MIB* stellt dazu verschiedene Berechnungsoperationen zur Verfügung (z.B. Durchschnitt) und ermöglicht die Verwendung von *Wildcards*, um beispielsweise mehrere Instanzen des gleichen MIB-Objekts zu referenzieren. Die Berechnung aktueller Werte für die aggregierten MIB-Objekte erfolgt dabei nach einem festgelegten *sample interval*.

Ausdrucksmöglichkeiten für Aggregationen

In Kombination mit der *Event MIB* wird es weiterhin möglich, aggregierte MIB-Objekte zu überwachen und Benachrichtigungsoptionen (*notifications*) festzulegen. Zur Steuerung dieses Vorgangs stehen innerhalb der Event MIB Tabellen zur Beschreibung von Bedingungen (*triggers*) und daraus resultierenden Ereignissen (*events*) zur Verfügung.

Bewertung

Die Internet Managementarchitektur leistet einen wichtigen Beitrag zum integrierten Netz- und Systemmanagement, scheint aber zur Anwendung für eine Dienstmanagementinformationsbasis nicht geeignet. Insbesondere die verfolgte Designvorgabe, ein möglichst einfaches Infomationsmodell zu erstellen, wirkt sich nachteilig auf die Darstellung von Dienstmanagementkonzepten aus.

Schwächen im Modellierungsansatz

Zwar wird in neueren MIBs stellenweise versucht, objektorientierte Techniken wie (Mehrfach-)Vererbung über gemeinsame Indizes in den Tabellen abzubilden, was sich allerdings zu Nachteilen hinsichtlich deren Übersichtlichkeit führt [Kel98]. Des Weiteren wurde die geschäftsorientierte Perspektive auf Dienstmanagementinformationen nur sehr eingeschränkt berücksichtigt, so dass Bezüge zwischen Diensten, deren Kunden und Nutzern sowie SLAs gegenwärtig nicht abgebildet werden. In diesem Zusammenhang erschwert insbesondere das Fehlen einer Enthaltenseinshierachie (*Containment Hierarchy*) die Darstellung derartiger Relationen.

Nichtsdestotrotz zeigte sich bei der Betrachtung der einzelnen MIBs, dass einige interessante Ergebnisse erzielt wurden, die als Vorlage zur Konzeption einer Dienstmanagementinformationsbasis verwendet werden können. Dies umfasst den im Rahmen der *Definitions of Managed Objects for WWW Services* eingeführten Abstraktionsmechanismus sowie die Verwendung von Wildcards zur Referenzierung mehrerer Instanzen (*Expression MIB*).

3.1.4 IT Infrastructure Library

Trend zu organisatorischen IT-Service Management

In den letzten Jahren ließ sich ein verstärkter Trend hin zum prozessorientierten IT-Service Management beobachten. Das Ziel dieser Bestrebung fußt darin, die Qualitätseigenschaften der erbrachten IT-Dienstleistungen – im Gegensatz zur rein technischen Betrachtung – mit Hilfe von organisatorischen Maßnahmen zu verbessern und gleichzeitig eine Verzahnung zwischen Unternehmensgeschäft und IT zu erreichen [Bre06, BKP02].

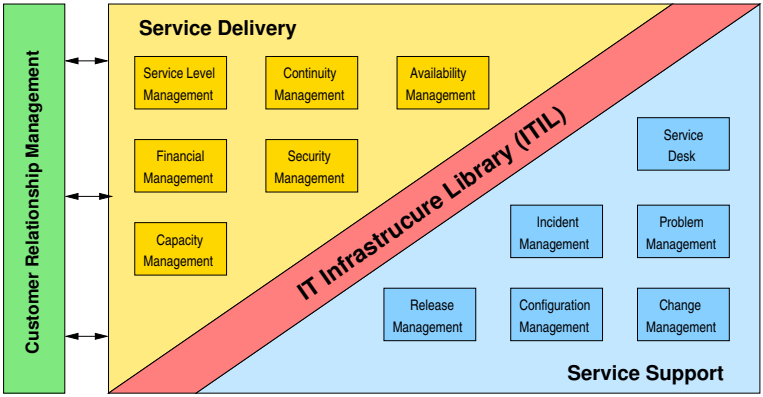

Abbildung 3.4: Service Delivery und Service Support

Gestützt wird diese Zielsetzung durch prozessorientierte Rahmenwerke, die Unternehmen Hilfestellung bei der Einführung von IT-Service

Managementprozessen bieten. In diesem Zusammenhang hat neben der bereits vorgestellten *eTOM* vor allem die von dem *British Office of Government Commerce (OGC)* herausgegebene *IT Infrastructure Library (ITIL)* [Off00, Off01, Off02] an zunehmender Bedeutung gewonnen. ITIL setzt sich mit der Planung und Bereitstellung *einer* kundenorientierten Dienstleistung auseinander[BKP02] und stellt zu diesem Zwecke eine Sammlung von Referenzprozessen zur Verfügung, die als Grundlage zur weiteren unternehmensspezifischen Verfeinerung und Adaption dienen sollen. Die einzelnen Prozessbeschreibungen wurden dabei durch Analyse von Betriebserfahrungen festgelegt, so dass in diesem Zusammenhang oftmals auch von einer *Best Practice*-Sammlung die Rede ist. Das ITIL-Rahmenwerk gliedert sich in die folgenden Teilbereiche:

- Die geschäftliche Perspektive (*The Business Perspective*)
- Planung und Lieferung von IT-Services (*Service Delivery*)
- Unterstützung und Betrieb der IT-Services (*Service Support*)
- Management der Infrastruktur (*ICT Infrastructure Management*)
- Anwendungsmanagement (*Applications Management*)

Unterstützung durch CMDB

Als Kernbereiche gelten *Service Support* und *Service Delivery*, welche sich mit der Planung bzw. dem Betrieb von IT-Diensten beschäftigen und in Abbildung 3.4 überblicksartig dargestellt werden. ITIL sieht zur Unterstützung dieser Prozesse die Schaffung einer *Configuration Management Database (CMDB)* vor, die u.a. ein logisches Modell von Diensten und Infrastrukturkomponenten abbilden soll und deshalb im Folgenden eingehender betrachtet wird.

Configuration Management Database

Prinzipiell verfolgt die CMDB, deren Erstellung und Verwaltung dem Configuration Management Prozess [Off00] obliegt, zwei verschiedene

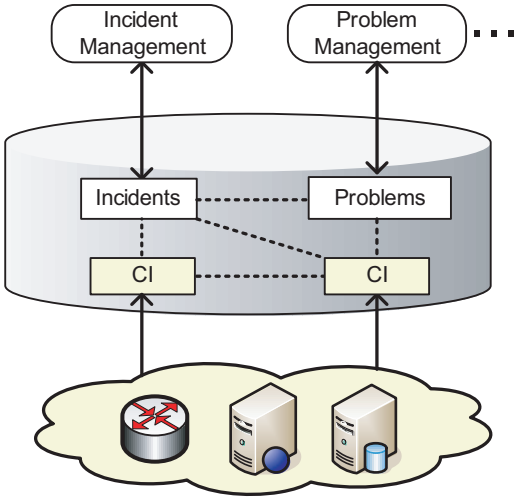

Abbildung 3.5: Aufgaben der CMDB

Zielsetzungen (siehe Abbildung 3.5): Einerseits soll sie ein logisches Modell der IT-Infrastruktur repräsentieren und dabei insbesondere Relationen zwischen den darin enthaltenen Elementen dokumentieren. Andererseits dient die CMDB dem Informationsaustausch zwischen verschiedenen ITIL-Prozessen, was vor allem Artefakte wie Incident oder Problem Records betrifft. In der CMDB verwaltete Elemente werden als *Configuration Items (CIs)* bezeichnet und sollten gemäß den ITIL-Vorgaben einem bestimmten Typ (CI Typ) zugeordnet werden. CIs repräsentieren damit Abstraktionen realer Entitäten in ähnlicher Weise, wie das für Managementobjekte (siehe Abschnitt 2.1) der Fall ist.

Allerdings zeigte ein Vergleich der CMDB mit aktuellen Managementmodellen erhebliche Unterschiede in der Zielsetzung [BSSG06]: Während Managementmodelle zur Unterstützung des täglichen Betriebs von IT Infrastrukturen konzipiert wurden, dient die CMDB vorwiegend den Entscheidungsträgern der verschiedenen ITIL-Prozesse, für die eine überblicksartige Darstellung oftmals wichtiger erscheint als technische

Details. Entsprechend entstehen wesentlich höhere Echtzeitanforderungen an Managementmodelle und dadurch ein Bedarf an Werkzeugunterstützung, wohingegen die Inhalte einer CMDB vorwiegend durch organisatorische Maßnahmen und unter Kontrolle des *Change Management* Prozesses [Off00] aktualisiert werden.

Bewertung

ITIL eignet sich als Quelle für Anforderungen

ITIL transportiert eine geschäftsorientierte Sichtweise und beschreibt in umfassender Weise während der Planung und des Betriebs von Diensten anfallende Managementaufgaben. Die enthaltenen Prozessbeschreibungen bieten deshalb eine wertvolle Quelle für Anforderungen an eine Dienstmanagementinformationsbasis, auch wenn Umsetzungs- und Realisierungsaspekte explizit nicht berücksichtigt werden. Gegenüber der in Abschnitt 3.1.2.1 vorgestellten eTOM liegt zwar ein geringerer Formalisierungsgrad vor, allerdings weist ITIL die größere Detailtiefe auf und hat insbesondere im europäischen Raum eine stärkere Verbreitung erzielt [Aal04].

CMDB verfolgt abweichende Zielsetzung

Weiterhin konnte die Erkenntnis gewonnen werden, dass substantielle Unterschiede zwischen dem CMDB-Konzept und Managementmodellen bestehen. Damit wurde deutlich, dass eine CMDB bzw. entsprechende kommerzielle CMDB-Produkte nicht zur Realisierung einer Dienstmanagementinformationsbasis geeignet sind.

3.2 Forschungsarbeiten

In diesem Abschnitt werden Forschungsarbeiten auf den Gebieten Modellierung und Beschreibung von Diensten, der Abhängigkeitsmodellierung sowie der Abbildung von Managementinformationen vorgestellt. Diese Aufteilung spiegelt die Anforderungskategorien **MOD/KMI**, **DEP** und **AGG** wider und stellt somit einen Bezug zu dem in Abschnit 2.3 dargestellen Anforderungskatalog her.

3.2.1 Modellierung und Beschreibung von Diensten

Das mit Abstand umfangreichste Gebiet stellen Arbeiten zur Modellierung und Beschreibung von Diensten dar. Neben Arbeiten, die sich mit grundlegenden Strukturierungsmaßnahmen für (Dienst-)Modelle beschäftigen, umfasst dies domänenspezifische Modelle und Dienstbeschreibungssprachen. Es sei bereits an dieser Stelle angemerkt, dass wegen des Umfangs und der Komplexität einer umfassenden Modellierung von Diensten allerdings zumeist partielle Lösungen geschaffen wurden. Eine Ausnahme hierzu bildet die am Ende dieses Abschnitts vorgestellte Habilitationsschrift von Dreo-Rodosek, die einer eingehenderen Betrachtung unterzogen wird.

Strukturierung von Managementmodellen Ausgehend von der Beobachtung, dass (Dienst-)Managementmodelle zu einer nicht mehr beherrschbaren Komplexität neigen, wird versucht, dieser Entwicklung durch Strukturierungsmaßnahmen bzw. einem mehrstufigen Entwicklungsprozess zu begegnen. Dazu wird in [SWMFR02, MF01, JMFW03] eine neuartige Vorgehensweise zur Definition von Managementinformationen (Abbildung 3.6) vorgeschlagen, die sich aus mehreren miteinander verwobenen Ebenen (*tiers*) zusammensetzt:

Einteilung in Ebenen (tiers)

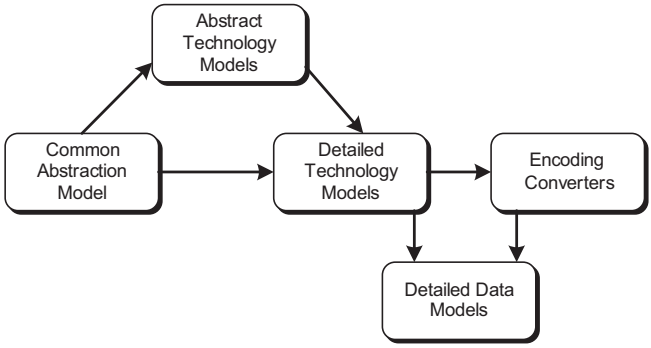

Abbildung 3.6: Der mehrstufige Entwicklungsprozess nach [SWMFR02]

- Als Vorlage für die Entwicklung technologiespezifischer Modelle soll das *Common Abstraction Model* fungieren. Es beinhaltet grundlegende Klassen (z.B. LogicalDevice, System, Service) und Assoziationen und bildet somit die Wurzelelemente der Vererbungshierachie ab.

- Das auf der nächsten Ebene angeordnete *Abstract Technology Model* soll grundlegende, für eine bestimmte Domäne gültige Konzepte verkörpern. Es erweitert dazu die vom *Common Abstraction Model* zur Verfügung gestellten Basisklassen, ohne allerdings auf Implementierungsdetails einzugehen. Nach [SWMFR02] sollte dieser Entwicklungsschritt sowohl von Technologieexperten als auch von Spezialisten im Bereich objektorientierter Modellierung durchgeführt werden.

- Durch Anreicherung des *Abstract Technology Models* mit technologie- und implementierungsspezifischen Aspekten sollen schließlich *Detailed Technology Models* entstehen und mit Hilfe von *Encoding Converters* auf konkrete Datenmodelle (*Detailed Data Models*) abgebildet werden. Dieser Vorgang sollte dabei ausschließlich von Technologieexperten vollzogen werden und entspricht im Grundsatz dem in Abschnitt 2.1 vorgestellten Transformationsprozess.

Anlehnung an OMG Metamodeling Architecture

Ein ähnliches Ziel verfolgt der [ZSCBB04] vorgestellte Ansatz, orientiert sich aber an den Bedürfnissen von Telekommunikationsanbietern. Hierbei wird eine Strukturierung von Managementinformationen entlang von *visibility levels (VLs)* vorgeschlagen, die sich an der *OMG Metamodeling Architecture* [OMG01] ausrichtet. Auf diese Weise sollen verschiedene Abstraktionsstufen (*Equipment*, *Network*, *Service* und *Business*) entstehen, die mit Hilfe eines gemeinsamen Metamodells (*ENST Metamodel*) zueinander in Beziehung gesetzt werden können.

Domänenspezifische Modelle Um die Bedürfnisse eines konkreten Anwendungsgebietes abzubilden, wurden eine Reihe domänenspezifischer Modelle vorgestellt. Diese entstanden entweder durch Erweiterung bestehender Managementmodelle oder durch Schaffung eines neuen Modellentwurfs.

Erweiterungen standardisierter Modelle

Beispielsweise wurde in [AK01] eine Erweiterung des *CIM Application Schemas* (Abschnitt 3.1.1) hinsichtlich von Lebenszyklusaspekten vorgenommen, wobei insbesondere auf die Klasse *CIM_Service* zurückgegriffen wurde. Dabei konnte die Erkenntniss gewonnen werden, dass sich aufgrund der Existenzabhängigkeit (*Weak Association*) zwischen *CIM_System* und *CIM_Service* die Modellierung von Diensten schwierig gestaltet und dass Erweiterungen oftmals zu einer neuen *Schema*-Version führen, was die Kompatibilität mit bestehenden Installationen gefährdet. Des Weiteren wurde in [MLB05] ein Informationsmodell zur Beschreibung von Dienstfehlern entwickelt und in das *SID* Rahmenwerk (Abschnitt 3.1.2) integriert. Hierbei bestätigte sich die in Abschnitt 3.1.2 getroffene Aussage, dass zwar in SID eine wohlgeformte Basisstruktur vorliegt, der Entwicklungsprozess aber noch nicht vollständig abgeschlossen wurde und deshalb detaillierte Teilmodelle nicht zur Verfügung stehen.

In dem Bereich *stand alone* Modelle wurde beispielweise in [LN00, LLN99] ein Ansatz für das Management IP-basierter Netzdienste vorgestellt. Dabei wurden zwar interessante Ergebnisse bezüglich des betrachteten Szenarios (*Customer Service Management*) erzielt, eine Anwendungsmöglichkeit für eine Dienstmanagementinformationsbasis erscheint jedoch wegen der spezifischen Ausrichtung des Modells nicht möglich. Ähnliches trifft auf den in [RS06] vorgestellten Ansatz zum Abrechnungsmanagement für IP-basierte Dienste zu.

Beschreibung funktionaler Eigenschaften

Dienstbeschreibungssprachen Insbesondere mit der Verbreitung von *Web Service* Technologien wurden eine Reihe von Beschreibungssprachen entwickelt, die sich allerdings vorwiegend mit der Auswahl und Zusammenstellung von Diensten (*service discovery* und *composition*) unter dynamischen Gesichtspunkten auseinandersetzen. Entsprechend liegt der Fokus dieser Sprachen auf einer Beschreibung der Dienstfunktionalitäten, wie beispielweise anhand der Web Service Description Language (WSDL) [CCMW01] und OWL-S [OWL] deutlich wird: Während WSDL eine Schnittstellenbeschreibung von Web Services ähnlich der IDL von CORBA [OMG99] ermöglicht, beschäftigt sich OWL-S mit der Darstellung von funktionalen Eigenschaften von Web Services und schafft so die Voraussetzung für eine Agenten-basierte Zusammenstellung verschiedener Dienste in Einklang mit Anwendervorgaben. Bei-

de Sprachen betrachten Dienste in einer Implementierung-agnostischen Sichtweise, Abhängigkeiten zu dienstrealisierenden Komponenten und damit für das Management wichtige Gesichtspunkte werden nicht berücksichtigt. Eine stärker am Management von Diensten ausgerichtete und als *Service Modelling Language (SML)* bezeichnete Sprache wurde in [Gop02] vorgestellt. Hierbei werden Dienste anhand von Wertebelegungen in einem neundimensionalen Raum (*type*, *size*, *duration*, *connectivity*, *QoS parameters*, *protocol parameters*, *value added features*, *assurance*, and *pricing*) charakterisiert, wobei die einzelnen Dimensionen als eine Art von Dienstattributen angesehen werden können. Zusätzlich stellt SML eine Reihe von mathematischen Operationen im neundimensionalen Raum zur Verfügung, mit deren Hilfe Dienste modifiziert und miteinander kombiniert werden können. Ein Erweiterungsmechanismus wurde allerdings nicht explizit vorgesehen, so dass fraglich erscheint, ob weitere Attribute bzw. Dimensionen integriert werden können.

SML unterscheidet neun Dimensionen

Übergreifende Arbeiten Wie bereits erwähnt, wurde innerhalb von Forschungsarbeiten zumeist nur ein Ausschnitt des Themenkomplexes Dienstmanagement betrachtet, so dass übergreifende Arbeiten zur Modellierung und Beschreibung von Diensten fast ausschließlich im Kontext von Standardisierungsgremien (Abschnitt 3.1) vorhanden sind. Eine Ausnahme hierzu bildet der in [DR00, DR02, DR03] vorgestellte Ansatz, der im Folgenden eingehender betrachtet wird.

Rahmenwerk für das Dienstmanagement

Den zentralen Bestandteil dieser Arbeit bildet ein Rahmenwerk für das Dienstmanagement, welches eine systematische Anforderungsanalyse, einen Modellierungsansatz für Dienste sowie Konzepte zur dynamischen Zusammenstellung von Diensten und Spezifikation von QoS-Eigenschaften umfasst. Ferner werden architekturelle Bestandteile einer Dienstmanagementplattform vorgestellt und Integrationsmöglichkeiten mit bestehenden Werkzeugen diskutiert. Von besonderer Bedeutung für die vorliegende Arbeit sind in diesem Zusammenhang die Methodik zur Analyse von Anforderungen an das Rahmenwerk sowie das entwickelte Informationsmodell. Zur Ableitung von Anforderungen wurden dabei zunächst die Funktionsbereiche des OSI-Managements (*FCAPS*) sowie verschiedene Aspekte des Dienstmanagements (*Roles*, *Quality*, *Information*, *Service Life Cycle* und *Management Disciplines*) betrachtet. Dadurch wurde die Grundlage für eine Verfeinerung der identifizierten

Anforderungen entlang der Teilmodelle einer Managementarchitektur (Funktions-, Organisations, Informations- und Kommunikationsmodell) geschaffen.

Dreistufige Template-Hierarchie

Ausgehend von der Anforderungsanalyse und unter Berücksichtigung des MNM-Dienstmodells [GHK+01] wurde schließlich ein Informationsmodell vorgestellt, welches sich auf eine dreistufige *Template*-Hierarchie stützt. Das auf der obersten Ebene befindliche *Service Template Model* stellt dabei grundlegende Klassen und Relationen unabhängig von einer provider- oder kundenspezifischen Ausprägung dar. Es dient als Vorlage zur Verfeinerung in ein *Provider-centric template model*, welches die Spezifika der Dienstbereitstellung aus Sicht eines Providers aufnimmt. Das *Customer-centric template model* (siehe Abbildung 3.7) stellt schließlich einen weiteren Verfeinerungsschritt dar und bildet zusätzlich Aspekte ab, die aus einer kundenspezifischen Dienstbereitstellung hervorgehen. Die verschiedenen Teilmodelle sind dabei als Vorlage zur weiteren Instanziierung und damit Anpassung an einen konkreten Provider bzw. Kunden beabsichtigt.

Fazit und Bewertung

Aufgrund der Fülle an Forschungsarbeiten zur Modellierung und Beschreibung von Diensten konnte im Rahmen dieses Abschnitts nur ein selektiver Überblick gegeben werden. Dennoch zeigen sich eine Reihe interessanter Ergebnisse: Zunächst ist festzustellen, dass Arbeiten, die sich mit der Strukturierung von Managementmodellen befassen, zwar keine unmittelbaren Realisierungsmöglichkeiten bieten aber dennoch wichtige Vorgehensrichtlinien hinsichtlich der Entwicklung einer Dienstmanagementinformationsbasis vermitteln. Insbesondere die Schaffung eines technologieneutralen Basismodells als Grundlage zur weiteren spezifischen Verfeinerung erscheint in diesem Zusammenhang als vernünftig.

Defizite konnten durch Erweiterungen nicht behoben werden

Bei der Betrachtung domänenspezifischer Modelle zeigte sich weiterhin, dass dabei sowohl Erweiterung von standardisierten Managementmodellen als auch eigenständige Modellentwürfe vorgestellt wurden. Während in den ersten Bereich fallende Arbeiten vorwiegend die in Abschnitt 3.1 aufgeführten Defizite von CIM und SID bestätigen und so-

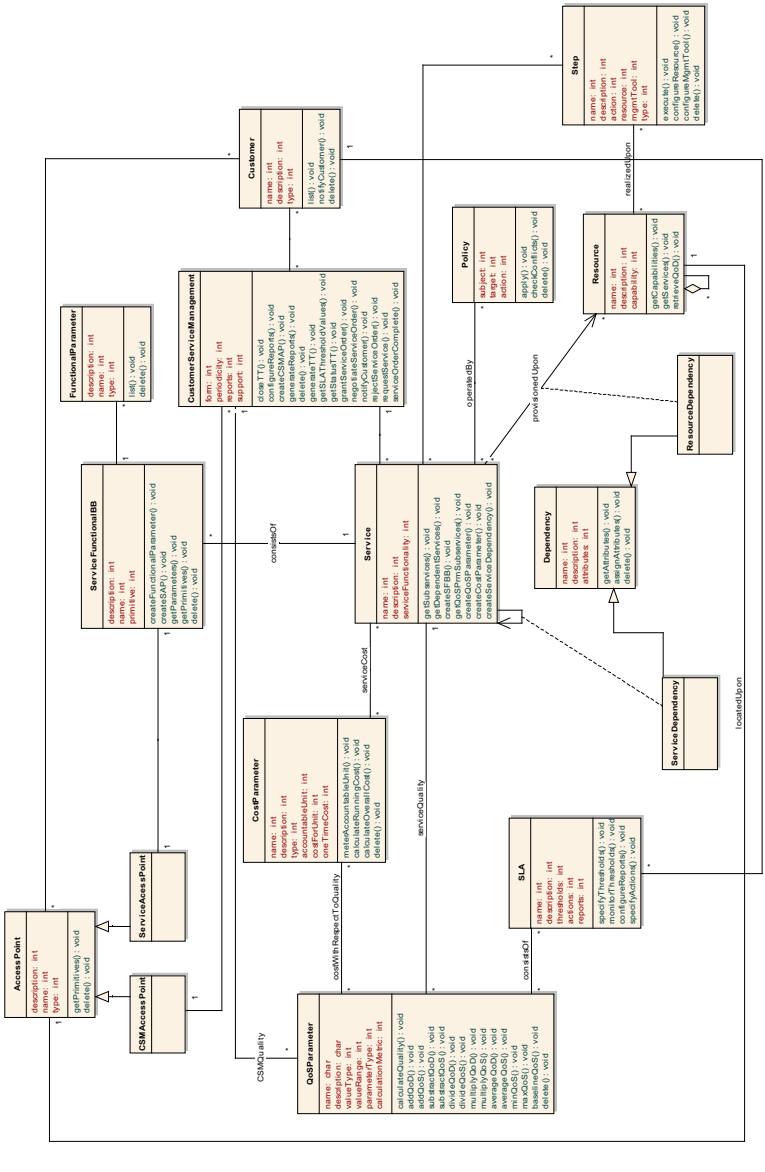

Abbildung 3.7: Customer-centric Template Model aus [DR02]

mit die Schwierigkeiten des Erweiterungsprozesses verdeutlichen, kann im Hinlick auf eigenständige Modelle generell festgestellt werden, dass sie sich aufgrund des engen Anwendungsbereichs nicht als Vorlage zur Umsetzung einer Dienstmanagementinformationsbasis eignen.

Wie weiterhin deutlich wurde, zeichnen sich die zur Beschreibung von Diensten entwickelte Sprachen oftmals durch den Fokus auf die Beschreibung von Dienstfunktionalitäten aus und berücksichtigen deshalb Managementaspekte (wie z.B. Relationen zu dienstrealisierenden Komponenten) nur äußerst rudimentär. Der stärker auf Managementbedürfnisse ausgerichtete Ansatz von Gopal [Gop02] erwies sich hingegen als ungeeignet aufgrund der eingeschränkten Ausdrucksmöglichkeiten der *Service Modeling Language*: Die Festlegung auf neun Dimensionen zur Beschreibung von Diensten ermöglicht nur eine sehr oberflächliche Darstellung der Spezifika der Dienstbereitstellung.

Vorlagen zur Erstellung einer Service-MIB

Abschließend konnte festgestellt werden, dass übergreifende Forschungsansätze nur in geringem Umfang existieren. Dennoch zeigte sich in dem Ansatz von Dreo-Rodosek, dass bereits wichtige Erkenntnisse hinsichtlich der Entwicklung einer Dienstmanagementinformationsbasis gewonnen wurden. Dies umfasst sowohl die systematische, anhand mehrerer Gesichtspunkte durchgeführte, Anforderungsanalyse als auch die in dieser Arbeit ebenfalls vorgestellten *Template Models*. Letztere repräsentieren verschiedene Verfeinerungsstufen eines Dienstmodells und eignen sich als Vorlage zur Erstellung eines Dienstmanagementinformationsbasis.

3.2.2 Abhängigkeitsmodellierung

Wie aus der Anforderungsanalyse in Kapitel 2 hervorgeht, muss eine Dienstmanagementinformationsbasis insbesondere in der Dienstbereitstellung auftretende Abhängigkeiten zwischen Diensten und Komponenten abbilden. Deshalb werden im Folgenden Forschungsarbeiten zur Abhängigkeitsmodellierung unter spezieller Berücksichtigung von Anforderungen der Kategorie **DEP** betrachtet.

Klassifizierung von Abhängigkeiten

Kar et al. Eine umfassende Sammlung von Arbeiten [Ale00, KKC00, KBK00, KK01] zur Modellierung und automatischen Gewinnung von Abhängigkeiten wurde von Kar et al. veröffentlicht. Zwar liegt der Fokus hierbei auf Abhängigkeiten zwischen verteilten Applikationen, die entwickelten Konzepte erscheinen aber aufgrund ihrer Generizität ebenfalls auf Dienste anwendbar. Dies betrifft insbesondere die in [KBK00] vorgestellte Klassifizierung von Abhängigkeitsbeziehungen anhand von mehreren Dimensionen:

- Zunächst wird mit der Dimension *Space/Locality/Domain* eine Unterscheidung dahingehend eingeführt, ob Abhängigkeiten Domänen- oder Systemgrenzen überschreiten (i*nter-domain*, *intra-domain*, i*nter-system*, *intra-system*).

- Die Dimensionen *Component Type* und *Component Activity* charakterisieren die Art der in Abhängigkeitsbeziehung stehenden Entitäten (Hardware, Software Package, Service) bzw. deren zur Verfügung gestellten Managementschnittstellen (aktiv oder passiv überwachbar). Ferner gibt die Dimension *Dependency Formalization* Auskunft über den Formalisierungsgrad von Abhängigkeiten.

- Weiterhin wurden Dimensionen zur Beschreibung der Kritikalität und Stärke von Abhängigkeiten (*Dependency Criticality* und *Strength*) eingeführt. Während die *Strength* Auskunft darüber gibt, ob Abhängigkeiten permanent auftreten oder nur während eines kurzen Zeitraumes benötigt werden (z.B. DNS-Zugriff zur Namensauflösung), zeigt die *Criticality* an, welchen Einfluss die betreffende Abhängigkeit auf die Verfügbarkeit des Dienstes hat.

Neben diesen sechs Dimensionen wird eine weitere Unterscheidung zwischen funktionalen und strukturellen Abhängigkeiten eingeführt und auf die Bedeutung temporaler Aspekte hingewiesen. Als funktional werden dabei Abhängigkeiten eingestuft, die implementierungsneutral auftreten und während der Planung eines Dienstes dokumentiert werden. Strukturelle Abhängigkeiten hingegen sollen während der Betriebsphase des Dienstes durch Anreicherung der funktionalen Abhängigkeiten mit szenariospezifischen Details entstehen.

Ausgehend von dieser Klassifikation werden von Kar et al. Ansätze zur automatischen Gewinnung von Abhängigkeiten (z.B. durch Auslesen von *System Repositories*) entwickelt. Da dies allerdings nicht als Aufgabe einer Dienstmanagementinformationsbasis angesehen wird, wurde auf eine weitergehende Analyse dieser Lösungen verzichtet.

Caswell and Ramathan In der bereits erwähnten Arbeit von Caswell und Ramathan [CR99] (Abschnitt 3.2.1) wird innerhalb des als *Service topology* bezeichneten Teils des entwickelten Dienstmodells ein Abhängigkeitsgraph vorgestellt, der verschiedene Arten von Abhängigkeiten unterscheidet:

- *Execution dependency:* Repräsentiert Abhängigkeiten zwischen einem Applikationsprozess und dem Rechnerknoten, auf dem dieser Prozess ausgeführt wird.
- *Link dependency:* Zeigt Abhängigkeiten auf, die aufgrund von Netzverbindungen zu dem Dienst enstehen.
- *Component dependency:* Ausgehend von der Beobachtung, dass Komponenten in der Dienstbereitstellung oftmals redundant ausgelegt werden, wurden diese in dem Abhängigkeitsmodell als Einheit betrachtet. Eine *Component Dependency* beschreibt somit die Abhängigkeit zwischen einer Sammlung redundanter Komponenten und dem Dienst.
- *Inter-service dependency:* Stellt Abhängigkeiten zwischen verschiedenen Diensten dar.
- *Organizational dependency:* Um den Übergang zwischen verschiedenen Zuständigkeitsbereichen in der Dienstbereitstellung zu verdeutlichen, wurden weiterhin organisatorische Abhängigkeiten eingeführt.

Geringe Formalisierung

Obwohl mit dieser Aufteilung ein wichtiger Beitrag zur Abhängigkeitsmodellierung geschaffen wurde, muss allerdings festgehalten werden, dass in [CR99] keine weitergehende Formalisierung erfolgt. Ferner wird

erwähnt, dass die unterschiedlichen Abhängigkeitstypen durch Kantenbeschriftungen im Dienstmodell unterschieden werden können, was aber auf die vorgestellten Beispiele nur sehr eingeschränkt zutrifft.

Hanemann In [Han07] werden Abhängigkeiten aus dem Blickwinkel der dienstorientierten Ereigniskorrelation betrachtet und im Gegensatz zu den bisher vorgestellten Arbeiten ein objektorientiertes Abhängigkeitsmodell in den Dienstmanagementkontext eingebettet. Dabei wird zunächst eine Unterscheidung zwischen Dienstabhängigkeiten (*InterServiceDependency*), Abhängigkeiten zwischen Diensten und Komponenten (*ServiceResourceDependency*) sowie Abhängigkeiten auf Komponentenebene (*InterResourceDependency*) getroffen. Zur weiteren Strukturierung werden diese verschiedenen Arten dann jeweils als zusammengesetzte (*CompositeDependency*) und atomare Abhängigkeiten (*SingleDependency*) modelliert[3].

Berücksichtigung temporaler Aspekte

Des Weiteren wurde ein *strength*-Attribut eingeführt, das die erwarteten Auswirkungen eines Ausfalls einer Komponente bzw. eines Dienstes beschreibt und somit als Basis für eine *Impact Analyse* dienen soll. Ebenfalls wurden Konzepte (*usageRecord*, *admin-* und *operationalStatus*) zur Beschreibung temporaler Aspekte geschaffen, die beispielsweise dokumentieren, ob die betreffende Abhängigkeit zu einem bestimmten Zeitpunkt bestand.

Abschließend muss noch erwähnt werden, dass sich eine Reihe weiterer Arbeiten (z.B. [Ens02, Ale03, AAG+04]) mit der automatischen Erfassung von Abhängigkeiten auseinandersetzen. Da allerdings im Hinblick auf die Erstellung einer Dienstmanagementinformationsbasis vorwiegend Strukturierungs- und Modellierungsaspekte von Abhängigkeiten im Vordergrund stehen, wird auf eine weiterführende Betrachtung an dieser Stelle verzichtet.

Bewertung

Die Ansätze von Kar et al. und Caswell und Ramathan weisen keinen Formalisierungsrad auf, der eine unmittelbare Anwendung gestat-

[3] Zu der verwendeten Modellierungstechnik siehe auch [HMSS06]

ten würde, vermitteln aber wichtige Vorgaben hinsichlich der Struktur einer Abhängigkeitsmodellierung. Durch die vorgestellten Klassifizierungen konnten verschiedene Arten von Abhängigkeiten identifiziert werden, die ebenfalls in der Entwiclung einer Dienstmanagementinformationsbasis berücksichtigt werden sollten. Allerdings gilt es hierbei einen Kompromiss zwischen einer möglichst detaillierten Abhängigkeitsmodellierung und dem dadurch entstehenden Pflegeaufwand zu finden.

OO-Modellenwurf erleichtert Einbettung

Wie sich weiterhin in der Betrachtung der von Hanemann vorgestellten Arbeit zeigte, liegt hierbei ein objektorientierter Modellentwurf vor, der neben Abhängigkeiten bereits grundlegende Beziehungen zu Entitäten im Dienstmanagementumfeld (z.B. Service, QoS, SLA) umfasst. Eine Einbettung dieses Ansatzes in eine Dienstmanagementinformationsbasis erscheint somit sinnvoll und einfach durchführbar.

3.2.3 Abbildung von Managementinformation

Bei der Betrachtung von Forschungsarbeiten zur Abbildung zwischen komponenten- und dienstorientierter Managementinformation zeigt sich, dass die überwiegende Mehrzahl von Ansätzen für spezielle Anwendungsfälle konzipiert wurde. Dies umfasst insbesondere Spezifikationssprachen für QoS- und SLA-Parameter.

Bei der Entwicklung von QoS-Spezifikationssprachen wurde dabei zumeist das Ziel verfolgt, eine Abbildung zwischen Dienstgüteparametern und Leistungsparametern von Netz- und Applikationskomponenten zu schaffen, was oftmals auch als vertikales QoS-Abbildungsproblem bezeichnet wird [HAN99]. In diesem Kontext wurden z.B. die *Quality of Service Specification Language (QoSSL)* [Gar04b], die *Quality Modeling Language (QML)* [FK98] sowie QUAL [DR02] entwickelt. Dabei wurden nützliche Sprachkonstrukte (z.B. zur Definition von Berechnungsvorschriften basierend auf Abhängigkeitsgraphen) entworfen. Eine Erweiterung dieser Sprachen hinsichtlich einer Abbildung von Komponentenparametern auf Dienstattribute erscheint aber schwierig: Es müssten zentrale Sprachelemente umbenannt bzw. adaptiert werden (beispielsweise durch Substituierung von QoS-Parameter mit Dienstattribut) was die Semantik der Sprache entscheidend verändern würde.

Ähnliches gilt für Ansätze im Bereich SLA-Spezifikation: Hierbei wurden neben Sprachen zur Formalisierung von SLAs (z.B. SLAng [LSE03]) auch Rahmenwerke zur Definition und Überwachung von SLA-Parametern vorgestellt. Um zu verdeutlichen, welche Bestandteile für eine Abbildung zwischen Komponentenparametern und Dienstattributen adaptiert werden können bzw. welche Schwierigkeiten dabei entstehen, wird im Folgenden mit dem *Web Service Level Agreement Framework (WSLA)* ein besonders umfassender Ansatz exemplarisch dargestellt.

Kennzahlen werden durch Metrics repräsentiert

Web Service Level Agreement Framework Das WSLA Rahmenwerk [KL03] widmet sich der Definition und Überwachung von SLAs und umfasst eine formale Spezifikationssprache sowie einen auf *Web Services* basierenden Architekturentwurf. Als zentrale Sprachelemente zur Beschreibung eines SLAs (siehe Abbildung 3.10) fungieren dabei *parties*, *service descriptions* und *obligations*. Vorgaben bezüglich der Abbildung von Komponentenparametern auf im SLA definierte Kennzahlen werden dabei innerhalb der *service description* durch das Element *Metric* abgebildet: Dabei können sowohl abstrakte Berechnungsfunktionen (*Function*) als auch Messvorschriften (*MeasurementDirectives*) spezifiziert werden. Letztere legen dabei fest, in welcher Weise Komponentenparameter gewonnen werden sollten (z.B. durch Ausführung eines Skriptaufrufs).

Zur Realisierung des WSLA-Frameworks wurde eine *Web Service*-basierte Architektur geschaffen, die sich aus Komponenten zur Aushandlung *(SLA Establishment Service)* und Überwachung von SLAs zusammensetzt (*SLA Compliance Monitor*). Interessanterweise erfolgt dabei insbesondere die Überwachung bzw. Berechnung von SLA-Parametern nach den in WSLA-Dokumenten enthaltenen Vorgaben. Um eine Einbindung in bestehende Managementarchitekturen zu ermöglichen wurde weiterhin eine Integration in CIM vorgenommen [DK03].

Bewertung

Wie bereits erwähnt stehen zur Abbildung zwischen Komponenten- und Dienstmanagementinformation bisher fast ausschließlich domänen-

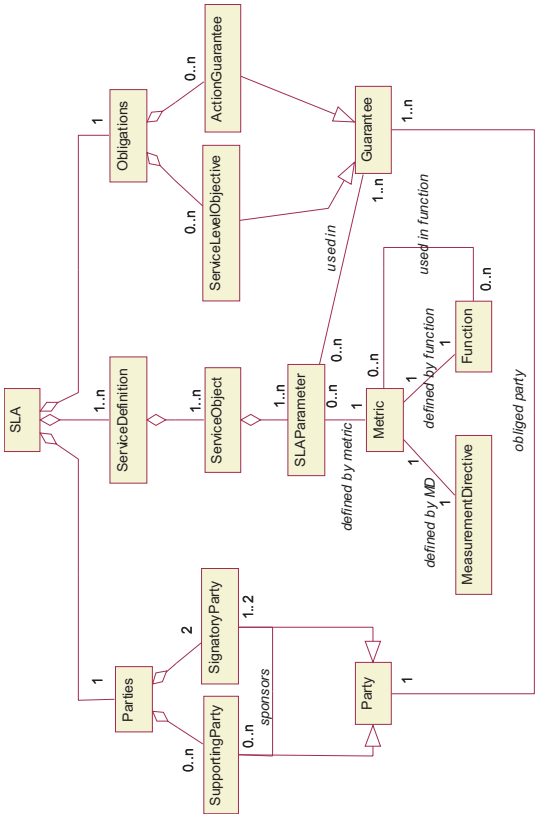

Abbildung 3.8: Sprachelemte von WSLA in Anlehnung an [KL03]

Semantische Änderungen erforderlich

spezifische Ansätze zur Verfügung. Bei der Untersuchung von Spezifikationssprachen für QoS- und SLA-Parameter wurde deutlich, dass zwar wertvolle Sprachkonstrukte zur Beschreibung von Berechnungsvorschriften entwickelt wurden, allerdings kein unmittelbarer Bezug zu allgemeinen Dienstattributen hergestellt wurde. Eine Anpassung dieser Sprachen würde deshalb gravierende Änderungen (z.B. Einfügen

und Umgruppieren von Elementen) nach sich ziehen, was deren ursprünglicher Zielsetzung widersprechen und somit praktisch einem neuem Sprachentwurf gleichkäme. Dies wird insbesondere am Beispiel von WSLA deutlich: Das zur Beschreibung von Diensten verwendete Element service definition wurde hierbei "unterhalb" des SLAs angeordnet, was nicht der Ausrichtung dieser Arbeit entspricht.

Kopplung von Beschreibung und Ausführung

Allerdings wurde als interessantes Konzept innerhalb des WSLA-Rahmenwerks die Kopplung von Beschreibungssprache und Ausführungsarchitektur vorgestellt. Dadurch konnte eine Integration von Spezifikation und Überwachung erzielt werden, was für eine Dienstmanagementinformationsbasis ebenfalls wünschenswert erscheint.

3.3 Beispiele kommerzieller Produkte

Die zunehmende Verbreitung des Dienstmanagements spiegelt sich ebenfalls in der Produktpalette von Herstellern kommerzieller Managementprodukte wider. In den letzten Jahren wurde eine Vielzahl kommerzieller Managementwerkzeuge lanciert und als integrierte und vollständige Lösungen für das *Service-Level Management, Business Alignment* etc. vermarktet.

Um zu untersuchen, welchen Beitrag kommerzielle Produkte zur Konzeption einer Dienstmanagementinformationsbasis leisten, werden in diesem Abschnitt zwei repräsentative Vertreter eingehender betrachtet. Zunächst wird mit dem *HP OpenView Service Quality Manager* ein Produkt vorgestellt, das eine dienst- und qualitätsorientierte Sichtweise auf IT-Infrastrukturen ermöglichen soll. Daran anschließend erfolgt die Untersuchung eines weit verbreiteten Werkzeugs zum *Service Level Reporting* für unterschiedliche Dienste und Nutzer (*InfoVista*).

3.3.1 HP OpenView Service Quality Manager

Als Teil des *HP OpenView for Operation Support Systems* wurde mit dem *Service Quality Manager* [HPO] ein Werkzeug zur Definition, Überwachung und Reporting von Qualitätseigenschaften eines Dienstes und SLAs vorgestellt. Dieses umfasst ein Modell zur Beschreibung von

Diensten und SLAs sowie eine Ausführungsarchitektur zur Sammlung und Verdichtung von Managementinformationen.

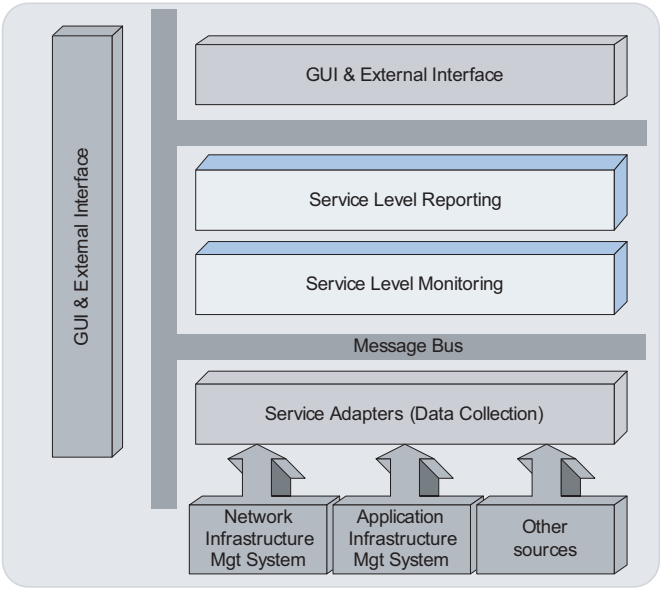

Abbildung 3.9: Architektureller Aufbau des HP Service Quality Managers in Anlehnung an [HPO]

Vereinfachtes Dienstmodell

Das zugrundeliegende Dienstmodell stellt allerdings nur einen schmalen und auf die Bedürfnisse dieses Werkzeugs zugeschnittenen Ausschnitt eines Dienstes dar. Er werden vorwiegend Beziehungen zwischen dem Dienst und dem dazugehörigen Dienstparameter (*Service Parameter*) dokumentiert, auf eine differenzierte Abhängigkeitsmodellierung oder Relationen zu Dienstkunden oder -nutzern wurde verzichtet. Um eine spätere Überwachung von Dienstparametern zu unterstützen, umfasst deren Definition zusätzlich Mess- und Aggregationsvorschriften. Letztere können in Form von PL/SQL-Anweisungen definiert werden. Ausgehend von einer Modellinstanz sammelt die Ausführungsarchitektur

(Abbildung 3.9) Daten von Netz- und Applikationskomponenten und verdichtet diese zu *Service Parameters*. Dabei ermöglichen verschiedene Agenten (*Service Adaptors)* den Zugriff auf heterogene Datenquellen und führen im Zuge dessen Datenanpassungsoperationen aus. Ferner erlaubt die Architektur eine Erstellung von *Service-Level Reports* basierend auf den ermittelten Werten für Dienstparameter.

3.3.2 Infovista

Fokus: Service Level Reporting

Von der Firma InfoVista [Inf] entwickelte Software-Produkte beschäftigen sich mit der Überwachung, Analyse und Darstellung der Performanz, Verfügbarkeit und Güte von IT-Diensten. Dabei liegt der Schwerpunkt auf einem *Service Level Reporting* anhand dienstspezifischer Kennzahlen, die die mit einem Kunden ausgehandelten SLAs repräsentieren. Der Provider soll in die Lage versetzt werden, mit Hilfe von Info-

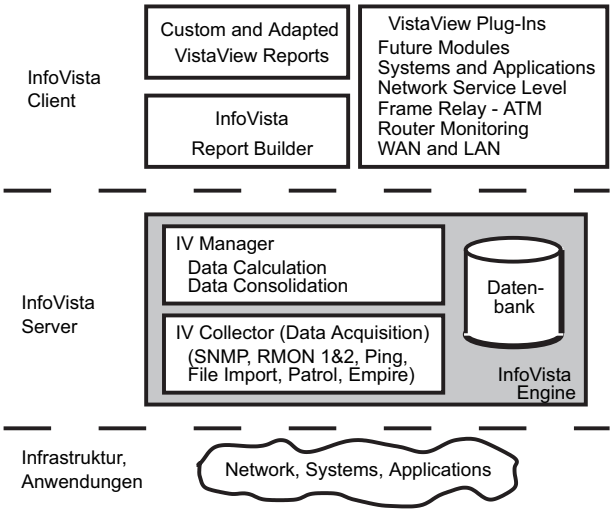

Abbildung 3.10: Die Architektur von InfoVista in Anlehnung an [Ner01]

Vista Kennzahlen über die IT-Infrastruktur und Dienste zu erheben, zu analysieren und aufzubereiten, um somit beispielsweise Entscheidungsgrundlagen für Optimierungsmaßnahmen und Kapazitätsplanungen zu gewinnen oder intern die Qualität der erbrachten Dienstleistung zu dokumentieren [Ner01]. Ferner unterstützt dieses Werkzeug die Erstellung von Reports, mit deren Hilfe Kunden über die erzielte Qualität ihrer abbonierten Dienste informiert werden können.

InfoVista folgt einem Client/Server Architekturentwurf, dessen zentraler Bestandteil die bestehende *InfoVistaEngine* bildet. Sie setzt sich im Wesentlichen aus drei Modulen zusammen:

- *InfoVista Collector:* Das Modul InfoVista Collector ist verantwortlich für die Sammlung von Managementinformationen (*Data Acquisition*) von Netz- und Applikationskomponenten. Dafür benutzt er vorwiegend die klassischen Managementprotokolle SNMP und ICMP, bietet aber auch eine Reihe weiterer Importschnittstellen (z.B. VisualBasic und Perl-APIs).
- *InfoVista Manager:* Der InfoVista Manager nimmt eine Verdichtung der gesammelten Rohdaten zu sogenannten Indikatoren vor. Zur Steuerung dieses Vorgangs werden arithmetische, statistische und temporalen Operationen zur Verfügung gestellt.
- *InfoVista Datenbank:* Die vom InfoVista Manager erstellten Indikatoren werden, zusammen mit den gesammelten Rohdaten, in der InfoVista Datenbank abgelegt und bilden somit die Basis für eine Erstellung von Reports.

Ausgehend von den bereits voraggregierten Indikatoren können nun mit Hilfe des *InfoVista Report Builders* und der *VistaView Plugins* dienst- und zielgruppenspezifische Reports erstellt werden. Um diesen Schritt für den Anwender zu erleichtern, wurden dabei Standardreports *(Report Templates)* für gängige Dienste geschaffen, die als Vorlage zur weiteren szenariospezifischen Verfeinerung benutzt werden können.

Bewertung

Geringe Ausdrucksmächtigkeit

Prinzipiell beschäftigen sich die vorgestellten Werkzeuge mit einer Definition von QoS- bzw. SLA-Parametern und deren Überwachung bzw. der Erstellung von Reports. Allerdings bieten die zugrundeliegenden Spezifkationskonzepte nur eine geringe Ausdrucksmächtigkeit [HLN01]. Insbesondere wird bei der Betrachtung des im *Service Quality Managers* verwendeten Dienstmodells deutlich, dass sich die Beschreibung vorwiegend an den Vorgaben des Netz- und Systemmanagements ausrichtet und somit beispielsweise Dienstabhängigkeiten nicht explizit dargestellt werden können.

Integration gestatet sich schwierig

Weiterhin wurden beide Produkte mit umfangreichen Ausführungsarchitekturen zur Aggregation von Komponentenparametern bzw. zur Erstellung von Reports ausgestattet. Diese ermöglichen zwar die Gewinnung von Daten aus Netz- und Applikationskomponenten bzw. Ausführung von Berechnungsoperationen, stellen aber keine offenen Schnittstellen zur Steuerung dieses Vorgangs zur Verfügung. Es erscheint deshalb nicht möglich, diese Werkzeuge im Hinblick auf die Aktualisierung einer Dienstmanagementinformationsbasis zu erweitern. Zudem wurden nur unzureichende Möglichkeiten zur Integration mit bestehenden Managementwerkzeugen geschaffen [HLN01], was insbesondere bei InfoVista deutlich wird. Statt auf bestehende Datenbestände anderer Managementwerkzeuge zurückzugreifen, führt die *InfoVista Engine* eigenständig Abfragen auf MIB-Variablen betreffender Komponenten durch [Ner01].

3.4 Gesamtbewertung bestehender Arbeiten

In diesem Kapitel wurden eine Vielzahl von Standardardisierungsarbeiten, Forschungsansätzen und kommerziellen Produkten vorgestellt und hinsichtlich ihres möglichen Beitrags zur Erstellung einer Dienstmanagementinformationsbasis evaluiert. Die individuelle Bewertung der Arbeiten erfolgte dabei aufgrund der in Abschnitt 2.3 identifizierten Anforderungen. Dabei zeigte sich, dass zwar interessante Ergebnisse erzielt wurden, allerdings keine der vorgestellten Arbeiten zur unmittelbaren

Realisierung einer Dienstmanagementinformationsbasis verwendet werden kann.

Ergebnisse der Evaluation

Um die in diesem Kapitel gewonnenen Erkenntnisse strukturiert aufzubereiten, werden nun zunächst die wichtigsten Defizite bestehender Arbeiten dargelegt. Daran anschließend werden Aspekte aufgezeigt, die eine gute Bewertung hinsichtlich eines Teilbereichs von Anforderungen erzielen und deshalb als Vorlage zur Entwicklung einer Dienstmanagementinformationsbasis verwendet werden können. Begleitend zu diesen Ausführungen wird in Tabelle 3.1 eine grafische Zusammenfassung der Evaluationsergebnisse präsentiert, wobei aus Übersichtlichkeitsgründen nur die wichtigsten Arbeiten berücksichtigt wurden. Mit ↯ ausgefüllte Felder verdeutlichen dabei Schwachpunkte bestehender Ansätze.

- Konkrete Dientmanagementinformation wurden nicht in ausreichendem Maße spezifiziert. Zwar wurden für das Dienstmanagement notwendige Entitäten wie *Service* oder *ServiceAccessPoint* eingeführt, aber gleichzeitig nur mit einer äußerst rudimentären Menge von generischen Attributen und Methoden versehen, die oftmals auf Namen und Beschreibung der Entität beschränkt bleiben. Zwar sind diese "Container-Objekte" als Basis für die Definition von konkreter Managementinformation durchaus geeignet, sind aber zum gegenwärtigen Zeitpunkt aus praktischen Gesichtspunkten unzureichend. Eine Verfeinerung der spezifizierten Entitäten hinsichtlich eines konkreten Szenarios führt aufgrund der Unvollständigkeit der vorliegenden Information zwangsläufig zu unterschiedlichen Ergebnissen, was einer Vereinheitlichung von Managementinformationen entgegenwirkt (vgl Anforderung MOD4). Dementsprechend fehlt das notwendige Fundament generischer Dienstattribute, die für die angestrebte, einheitliche Sicht auf Dienstmanagementinformationen erforderlich ist. Ferner zeigt sich, dass aufgrund der Zielsetzung ein möglichst umfassendes Modell zu schaffen, eine unübersichtliche und fast nicht beherrschbare Anzahl von Klassen entstanden ist.

- Es wurde bisher noch keine klare Ableitungsmethodik für Dienstmanagementinformationen vorgestellt. Dadurch fällt es schwer nachzuvollziehen, welchem Anwendungszweck die spezifizierten

Entitäten und Attribute dienen. Ebenfalls wurden nur unzureichende Integrationsmöglichkeiten verschiedener Managementmodelle geschaffen, was die Anwendung in einem heterogenen Umfeld verkompliziert.

- Es exisistieren keine allgemeinen Konzepte zur Abbildung zwischen Komponentenparametern und Dienstattributen. Zwar wurden innerhalb von Spezifikationssprachen Möglichkeiten zur Beschreibung von Berechnungsvorschriften geschaffen, diese Arbeiten beschränken sich aber entweder auf spezifische Anwendungsdomänen (z.B. WSLA) oder können nur in Kombination mit einer bestimmten Managementarchitekturen verwendet werden (z.B: Expression-MIB). Zudem zeigte sich insbesondere in der Betrachtung kommerzieller Produkte, dass zwar umfassende Ausführungsarchitekturen zur Aggregation von Komponentenparamtern geschaffen wurden, allerdings aufgrund der verwendeten proprietären Schnittstellen keine Anpassungs- und Erweiterungsmöglichkeiten gegeben sind.

Nichstdestrotz zeigten sich eine Reihe interessanter Konzepte, die in die Entwicklung einer Dienstmanagementinformationsbasis einbezogen werden sollten und in Tabelle 3.1 durch eine entsprechende Bewertung (✓) verdeutlicht werden:

- Die IT Infrastructure Library enthält eine umfassende Sammlung von Dienstmanagementaufgaben und eignet sich deshalb zur Ableitung von Anforderungen an die Inhalte einer Dienstmanagementinformationsbasis.

- In [DR02] findet sich sowohl der Entwurf eines Dienstmodells als auch eine systematische Anforderungsanalyse. Da hierbei eine generische Sichtweise gewählt wurde, erscheint eine Anpassung dieser Konzepte sinnvoll.

- Die in [Han07] vorgestellte Abhängigkeitsmodellierung stützt sich auf einen objektorientierten Modellentwurf und stellt bereits Verbindungen zu grundlegenden Entitäten im Dienstmanagementumfeld her. Somit kann eine Einbettung dieses Ansatzes in eine

Dienstmanagementinformationsbasis einfach vorgenommen werden.

- In [KL03] konnte eine Integration zwischen Spezifikation und Überwachung von Managementinformationen erzielt werden, was für eine Dienstmanagementinformationsbasis ebenfalls wünschenswert erscheint.

Unter Einbeziehung der genannten Aspekte wird nun im folgenden Kapitel der Lösungsansatz dieser Arbeit entwickelt.

	CIM	SID	eTOM	IETF	ITIL	[DR02]	[Han07]	WSLA	HPSQM
ALL 1	(✓)	✓	✓	↯	✓	✓	↯		(✓)
ALL 2			✓		✓	✓			
MOD 1	↯	↯				✓	✓		(✓)
MOD 2	✓	✓				✓	✓		↯
MOD 3						(✓)			
MOD 4	(✓)	(✓)						✓	
MOD 5		(✓)				✓	✓		
KMI 1				↯				↯	↯
KMI 2	↯		✓		✓				
KMI 3									
KMI 4	✓			↯		✓	✓		
KMI 5	(✓)			↯					
KMI 6									(✓)
KMI 7		(✓)							
DEP 1	(✓)		↯			(✓)	✓		↯
DEP 2		(✓)		↯		(✓)	✓		↯
DEP 3							✓		
AGG 1								↯	
AGG 2									
AGG 3								✓	
AGG 4								✓	✓
REA 1				(✓)					
REA 2	(✓)	↯		✓				(✓)	(✓)
REA 3	(✓)	(✓)				(✓)	(✓)	(✓)	
REA 3	(✓)			(✓)				(✓)	(✓)
REA 5	(✓)	↯		✓					

Tabelle 3.1: Bewertung bestehender Arbeiten

Kapitel 4

Lösungsansatz und weiteres Vorgehen

Wie bereits im Rahmen der Anforderungsanalyse dargelegt, gilt es, mehrere Herausforderungen bei der Konzeption einer Managementinformationsbasis für das Dienstmanagement zu berücksichtigen. Aus dem vorigen Kapitel geht weiterhin hervor, dass verwandte Arbeiten nur partiell zur Lösung dieser Herausforderungen beitragen und damit die Notwendigkeit für eine neue Herangehensweise besteht. Die zentrale Lösungsidee dieser Arbeit vorzustellen und damit die genannten Defizite zu adressieren, stellt den Gegenstand dieses Kapitels dar. Der entwickelte Ansatz wird dabei von einem methodischen Konzept begleitet, das gleichzeitig das weitere Vorgehen dieser Arbeit bestimmt. Zunächst erfolgt allerdings die Vorstellung der Service-MIB Idee und deren Bestandteile.

4.1 Die Service-MIB Idee

An mehreren Stellen dieser Arbeit werden bestehende Konzepte zur Informationsmodellierung im Netz- und Systemmanagement genannt.

Diese Konzepte bestimmen weitgehend die heutige Managementlandschaft, haben sich über die Jahre hinweg bewährt und gingen zudem in die Betriebserfahrung von Administratoren über. Es erscheint deswegen vernünftig, auf dem Weg zu einem integrierten Service Management auf diese Konzepte zu bauen und sie mit neuen, erforderlichen Bestandteilen zu erweitern.

Service-MIB repräsentiert Managementsicht auf Dienst

Die Service-MIB verfolgt diesen Ansatz: Sie versucht, in ähnlicher Weise wie bestehende MIBs Netzkomponenten repräsentieren, eine umfassende Beschreibung von Diensten zu vermitteln. Auch in diesem Fall stellt Abstraktion das Schlüsselkonzept dar: Es werden managementrelevante Aspekte eines Dienstes identifiziert, durch geeignete Attribute und Operationen charakterisiert und in ein geeignetes Modell überführt. Ferner finden Beziehungen zu grundlegenden Entitäten im Dienstmanagementumfeld wie z.B. Dienstvereinbarungen Berücksichtigung. Es entsteht somit ein Kompositum an dienstorientierter Managementinformation, was sich in der Namensgebung des Ansatzes widerspiegelt.

Dienst-eigenschaften stehen in Abhängigkeit zu Komponenten-parametern

Gegenüber bestehenden MIBs muss allerdings ein weiterer Aspekt in die Betrachtung aufgenommen werden, der Erweiterungen notwendig erscheinen lässt: Die Eigenschaften eines Dienstes sind untrennbar mit den Eigenschaften der Komponenten verbunden, die diesen Dienst realisieren. Effektiv werden deshalb Überwachungs- und Kontrollaktionen auf dem Dienst zu einem großen Teil auf dienstrealisierenden Komponenten durchgeführt. Soll also eine managementorientierte Beschreibung eines Dienstes konzipiert werden, müssen derartige Zusammenhänge berücksichtigt und in die Spezifikation von Diensteigenschaften integriert werden. Hinsichtlich der Informationsmodellierung erwächst daraus vor allem die Notwendigkeit, eine Abbildung von komponentenorientierter auf dienstorientierte Managementinformation zu schaffen.

Abbildung 4.1 veranschaulicht nocheinmal die genannten Aspekte und ihre Einbettung in den Service-MIB Ansatz, der sich aus zwei grundlegenden Bestandteilen zusammensetzt, wie im rechten oberen Quadranten dargestellt:

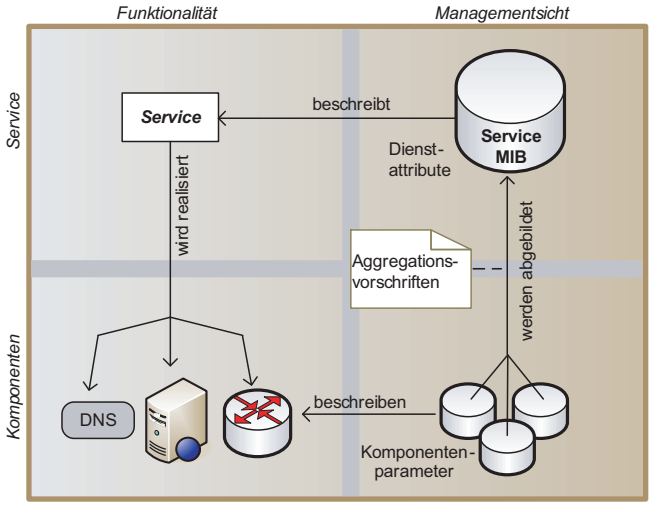

Abbildung 4.1: Der Service-MIB Ansatz

- *Repräsentation einer Managementsichtauf den Dienst*
  Wie auf der linken Seite in Abbildung 4.1 ersichtlich, wird die Funktionalität eines Dienstes durch eine Reihe von Komponenten realisiert. Um eine Managementsicht auf diese Komponenten herzustellen, werden nach gängiger Praxis (vgl. Abschnitt 2.1) managementrelevante Abstraktionen gebildet (insbesondere in Form geeigneter Kompontenparameter) und in Informationsbasen zusammengefasst. Effektiv können in dieser Weise Managementaktionen durch lesenden und schreibenden Zugriff auf die entsprechenden Informationsbasen initiiert werden. Wie in der oberen horizontalen Ebene dargestellt, wird mit der Service-MIB das Ziel verfolgt, geeignete Abstraktionen für einen Dienst zu schaffen und somit eine dienstorientierte Managementsicht zu repräsentieren.

- *Abbildung von Komponenten- auf Dienstinformationen*
  Um die erwähnte Abbildung zwischen Komponenten- und Dienstmanagementinformation zu ermöglichen, werden zusätzlich Ag-

gregationsvorschriften definiert und in die Service-MIB integriert. Dies erlaubt insbesondere eine Spezifikation von Dienstattributen in Abhängigkeit von Komponentenparametern und somit eine einfachere Überwachung und Kontrolle von Diensten, wie sich in den folgenden Abschnitten zeigen wird.

Um die Verflechtung dieser beiden Bestandteile zu verdeutlichen, wird ein kurzes Beispiel vorgestellt.

Beispiel

Für den im Rahmen der Anforderungsanalyse präsentierten GRID-Computing Dienst (Abschnitt 2.2.2) verkörpert die Verfügbarkeit[1] eine wichtige Diensteigenschaft: Sie lässt Rückschlüsse auf den Gesamtzustand des Dienstes zu, stellt Gegenstand der Dienstvereinbarung dar und wird deshalb als Dienstattribut `OperationalStatus` in die Service-MIB mitaufgenommen.

Die Verfügbarkeit dieses Dienstes bzw. der Wert des damit assoziierten Dienstattributs wird in diesem Beispiel von den Verfügbarkeiten der dienstrealisierenden Komponenten beeinflusst, wie in dem vereinfachten Modell in Abbildung 4.2 ersichtlich. In diesem Fall sind dies die Verfügbarkeiten der Rechnerknoten, auf denen die tatsächlichen Berechnungen durchgeführt werden, gepaart mit der Verfügbarkeit des Routers und des DNS Servers, die für die Netzkonnektivität des Dienstes verantwortlich sind.

Berechnung des Dienstattributwertes

Den Bezug zwischen Komponentenparametern und Dienstattribut stellt die *ServiceAvail* Aggregation her: Sie bildet die qualifizierten Komponentenparameter (*availDNS*, *availRout* und *availNode*) auf das gewünschte Dienstattribut (*availCS*) unter Verwendung der dargestellten Formel ab und erlaubt die Berechnung dessen Wertes. Durch die Verflechtung von Dienstattribut und Abbildungsvorschrift wird somit eine feingranulare Beschreibung des Dienstes erzielt und gleichzeitig die Überwachung des Dienstes vereinfacht. Ebenfalls wird in dieser Weise eine szenariospezifische Anpassung inhärent unterstützt.

[1] Für eine ausführliche Diskussion zu dieser Thematik siehe z.B. [Kai99]

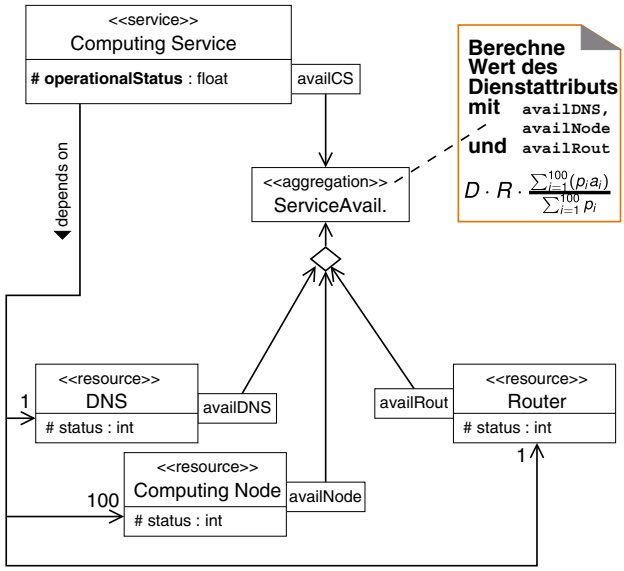

Abbildung 4.2: Beispielmodell für Dienstattribut und Aggregation

Nach der Skizzierung der Bestandteile des Service-MIB Ansatzes, wird nun im nächsten Abschnitt das methodische Vorgehen zur Realisierung dieser Konzepte vorgestellt.

4.2 Methodik zur Konzeption einer Service-MIB

Stellt man die in Kapitel 2 identifizierten Anforderungen dem Service-MIB Ansatz gegenüber, wird deutlich, dass im Hinblick auf seine Realisierung mehrere Aspekte Berücksichtigung finden müssen. Nicht nur sollte der Frage nachgegangen werden, welcher Informationsbedarf für Dienstmanagementinformationen tatsächlich besteht, sondern es bedarf auch Konzepte zur Spezifikation und späteren Überwachung bzw. Nut-

zung von dienstorientierter Managementinformation. Um dieser umfassenden Sichtweise gerecht werden zu können, wird eine methodischer Rahmen erforderlich, der zugleich das weitere Vorgehen der vorliegenden Arbeit bestimmt.

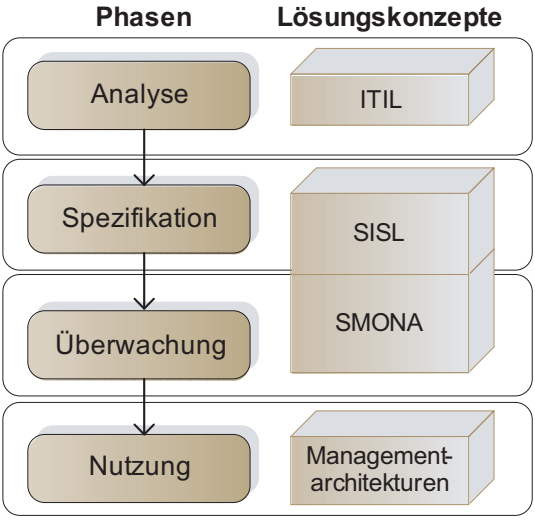

Abbildung 4.3: Methodik zur Synthese von Dienstmanagementinformationen

Methodik besteht aus vier Phasen

Die vorrangige Idee der in dieser Arbeit entwickelten Methodik besteht darin, Phasen für die Konzeption einer Service-MIB zu identifizieren und – darauf basierend – sukzessive Lösungen für die jeweiligen Phasen zu entwickeln. Der Hauptvorteil dieser Herangehensweise liegt in der damit vermittelten, integrierten Sichtweise: Durch die Berücksichtigung von Übergängen zwischen den Phasen kann somit eine umfassende Lösung geschaffen werden, die den gestellten Anforderungen bestmöglich gerecht wird.

Die entwickelte Methodik umfasst vier Phasen, die in Abbildung 4.3 dargestellt und im Folgenden weiter ausgeführt werden. Zusätzlich erfolgt die Skizzierung der für diese Phasen relevanten bzw. im Rahmen

dieser Arbeit enwickelten Konzepte; ihre umfassende Darstellung findet sich in den folgenden Kapiteln dieser Arbeit.

4.2.1 Analysephase

Die primäre Aufgabe der *Analysephase* besteht darin, den Bedarf an dienstorientierter Managementinformation zu ermitteln und im Zuge dessen der Frage nachzugehen, welche Informationen zur Erfüllung von Dienstmanagementaufgaben tatsächlich benötigt werden. Die Analysephase legt somit die Grundlagen für eine Definition konkreter Managementinformation: Sie bestimmt Informationsanforderungen an eine Service-MIB und zeigt somit den Bedarf an Entitäten, Attributen und Abhängigkeitsbeziehungen auf.

Top-Down Vorgehensweise wird gewählt

Für die vorliegende Arbeit wird für die Analysephase eine Top-Down Vorgehensweise gewählt, die sich konsequent an dem Informationsbedarf von Providern, Managementanwendungen und Nutzern orientiert. Sie wird gestützt von der in Abschnitt 3.1.4 vorgestellten *IT Infrastructure Library (ITIL)*: Anhand von ITIL-Prozessbeschreibungen werden Managementaufgaben ermittelt und daraus der Bedarf an dienstorientierter Managementinformation abgeleitet. Zusätzlich fließen Dienstlebenszyklus und organisatorische Aspekte in die Betrachtung mit ein. Auf diese Weise können eine vollständige und allgemeingültige Lösung geschaffen und somit die Anforderungen KMI 3 und KMI 4 adressiert werden. Eine ausführliche Darstellung der Analysephase findet sich in Kapitel 5.

4.2.2 Spezifikationsphase

Schaffung eines Modells

Daran anschließend widmet sich die *Spezifikationsphase* einer formalen Beschreibung von Dienstmanagementinformationen. Ziel hierbei ist es, den vorab ermittelten Informationsbedarf in Entitäts-, Attributs- und Beziehungsdefinitionen zu überführen und dadurch eine computergestützte Verarbeitung zu ermöglichen. Die Spezifikationsphase bestimmt somit die konzeptionellen Inhalte einer Service-MIB und vermittelt eine abstrakte Sicht auf Dienstmanagementinformationen. Letztere mündet in der Schaffung eines geeigneten Modells, das im Hinblick auf den in

Abschnitt 2.1.2 dargestellten Modellebenen auf der Informationsmodellebene angesiedelt ist.

Im Hinblick auf den zweiten Kernbestandteil der Service-MIB wird in der Spezifikationsphase des Weiteren die Abbildung von komponentenorientierter auf dienstorientierte Managementinformation adressiert. Gemäß den in Abschnitt 2 ermittelten Anforderungen ist an dieser Stelle eine deklarative Spezifikationsmöglichkeit (AGG 2) gefragt, die gleichzeitig Unterstützung für eine spätere Ausführung der Abbildungsvorschriften durch Überwachungswerkzeuge vorsieht (AGG 4). Im Hinblick auf die Phasenstruktur der vorgestellten Methodik bedeutet dies, dass der Übergang von der Spezifikations- auf die darauffolgende Überwachungsphase berücksichtigt werden muss.

Spezifikation von Dienstattributen in Abhängigkeit von Komponentenparametern

Zur Erreichung dieser Zielsetzung sieht die vorliegende Methodik eine Aufteilung der Spezifikationsphase in mehrere Teilschritte vor, wie auch in Abbildung 4.3 dargestellt. Wie bereits bei der Vorstellung der Service-MIB Idee erläutert, besteht die grundlegende Idee dieser Herangehensweise darin, Beschreibungen von Diensteigenschaften (*Dienstattribute*) mit Aggregations- und Messvorschriften anzureichern und somit den Grundstein für eine spätere Auswertung dieser Vorschriften in der Überwachungsphase zu legen. In diesem Zusammenhang werden 5 Teilschritte unterschieden (Abbildung 4.4):

I. Deklaration statischer Eigenschaften
Statische Eigenschaften eines Dienstattributs (z.B. der Name oder die textuelle Beschreibung) werden bestimmt und somit die Grundstruktur des Attributs festgelegt. Dies umfasst weiterhin den *Typ* und die *Einheit* des Attributs.
Beispiel: Im Hinblick auf das in Abbildung 4.2 dargestellte Szenario können in diesem Schritt der Name des Attributs (`OperationalStatus`), dessen Typ (`float`), sowie eine Beschreibung a priori ermittelt werden, wie in Tabelle 6.25 dargestellt wird.

II. Ermittlung relevanter Komponenten
Um Abbildungen von Komponenten- auf Dienstmanagementinformation zu schaffen, müssen zunächst relevante Komponenten für das in Schritt I angelegte Dienstattribut identifiziert werden.

Hierfür bieten sich in der Service-MIB dokumentierte Abhängigkeitsbeziehungen zwischen Diensten und Komponenten an: Durch Nachverfolgung von der Dienstinstanz ausgehender Relationen kann somit eine Kandidatenmenge relevanter Komponenten ermittelt werden. Es muss an dieser Stelle erwähnt werden, dass sich prinzipiell auch die in Kapitel 3 vorgestellten Managementmodelle (insbesondere SID und CIM) zur Durchführung dieses Schritts eignen – sofern die genannten Abhängigkeitsbeziehungen erfasst wurden. Stehen derartige Modelle nicht zur Verfügung, bleibt nur ein Rückgriff auf die Betriebserfahrung der mit dem Management des Dienstes betrauten Administratoren.
Beispiel: Anhand der in Abbildung 4.2 dargestellten `depends on` Relation (linke Seite des Bildes) kann festgestellt werden, dass der Computing Dienst funktional von DNS server, Router und Rechnerknoten abhängt.

III. Identifikation relevanter Komponentenparameter
Der nächste Schritt besteht darin, für eine spätere Abbildung relevante Komponentenparameter zu identifizieren. Ausgehend von der vorab ermittelten Kandidatenmenge an Komponenten müssen zu deren Beschreibung verwandte Parameter betrachtet und ihr Einfluss auf das vorliegende Dienstattribut ermittelt werden.
Beispiel: Der Computing Dienst aus Abbildung 4.2 ist nicht verfügbar, falls Router, DNS Server und die Rechnerknoten unerreichbar sind. Dementsprechend werden die jeweiligen `status`-Attribute als relevant für das Dienstattribut `OperationalStatus` angesehen.

IV. Festlegung von Messparametern
Im Hinblick auf eine Überwachung der identifizierten Komponentenparameter werden nun Messparameter definiert, die steuern, in welcher Weise Messungen dieser Parameter durchgeführt werden sollen. Dies umfasst die Festlegung von entsprechenden Abtastraten, Datenformaten, APIs oder Protokollen.
Beispiel: Für das Computing Dienst Szenario erweist es sich als ausreichend, den Zustand der identifizierten Komponenten dahingehend zu überprüfen, ob sie erreichbar sind (*up* oder *down*). Wie diese Überprüfung durchgeführt wird, hängt in erster Linie von

den vorhandenen Managementwerkzeugen ab. Denkbare Messparameter wären in diesem Fall eine Abtastrate von einer Sekunde oder ein gewünschtes Datenformat *boolean*.

V. Festlegung von Abbildungsvorschriften
Der letzte Schritt umfasst die Festlegung von Regeln zur Abbildung von Komponenten- auf Dienstmanagementinformation. Dafür werden Abbildungsvorschriften spezifiziert, die beschreiben, in welcher Form die in Schritt III identifizierten Komponentenparameter kombiniert werden sollen, um den Wert des Dienstattributs zu reflektieren.
Beispiel: Wie bereits erwähnt, beeinflussen die Verfügbarkeit des Routers, DNS Servers und der Rechnerknoten die Verfügbarkeit des Computing Dienstes bzw. des als Beispiel gewählten Dienstattributs. Die Formel in Abbildung 4.4 stellt eine mögliche Aggregationsvorschrift dar, die diesen Sachverhalt beschreibt.

Wie aus der Gegenüberstellung bestehender Arbeiten hervorgeht, eignet sich keiner dieser Ansätze zur unmittelbaren Realisierung dieser Schritte. Es wurden zwar Managementmodelle (z.B. CIM, SID) entwickelt, die Abhängigkeiten zwischen Diensten und Komponenten beinhalten, es fehlen allerdings Konstrukte, um eine feingranulare Abbildung von Komponenten- auf Dienstmanagementinformationen vorzunehmen (*Teilschritt 5* der Methodik). Ferner sind entsprechende Aggregationsrahmenwerke (z.B. WSLA, Expression MIB) domänen- und technologiespezifisch (siehe Abschnitt 3.2.3) und deshalb ebenfalls nicht zur Realisierung der Spezifikationsphase geeignet.

Um diese Lücke zu schließen, wird in Kapitel 6 die *Service Information Specification Language (SISL)* entwickelt. Sie erlaubt es, Abbildungen von Komponentenparametern auf Dienstattribute in Form von deklarativen Aggregationsvorschriften auszudrücken und somit in die Service-MIB zu integrieren.

4.2.3 Überwachungsphase

Ein entscheidendes Kriterium für die Service-MIB stellt ihre Aktualität dar (siehe Anforderung REA 3), d.h. in ihr enthaltene Informatio-

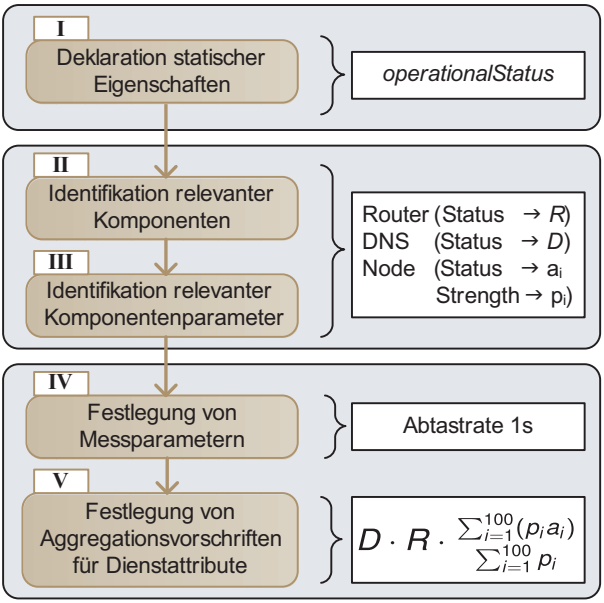

Abbildung 4.4: Teilschritte zur Abbildung von Komponenten- auf Dienstmanagementinformation

nen müssen den aktuellen Zustand des Dienstes reflektieren. Dies wird innerhalb der *Überwachungsphase* durch Einführung einer geeigneten Dienstüberwachung adressiert, die aktuelle Werte für die vorab spezifizierten Dienstattribute ermittelt und somit eine aktuelle Sicht auf den Dienst ermöglicht. Allerdings sind in diesem Fall die Anforderungen an Überwachungswerkzeuge vielfältiger: Sie müssen in der Lage sein, komponentenorientierte Monitoring-Daten aus vielfältigen Quellen zu sammeln und schließlich zu Dienstmanagementinformationen zu verdichten. Erschwerend kommt hierbei wieder hinzu, dass Dienste mit ähnlicher Funktionalität auf verschiedenste Arten realisiert werden können. Gefragt ist also an dieser Stelle eine Möglichkeit zur flexiblen Anpassung der Überwachung an die jeweiligen szenariospezifischen Gegebenheiten.

SISL steuert den Überwachungsvorgang

Eine Lösung zur Unterstützung dieses Vorgangs wurde bereits vorgestellt: Die innerhalb der Spezifikationsphase definierten und mit Hilfe von SISL formalisierten Aggregationsvorschriften zielen auf eine Steuerung von Überwachungswerkzeugen ab. Sie enthalten Informationen darüber, wie Komponentenparameter gemessen und schließlich auf Dienstattribute abgebildet werden sollen.

Da bisher kein Überwachungswerkzeug SISL-Anweisungen unmittelbar auswerten kann, wird in Kapitel 7 eine Erweiterung des in [DS05] vorgestellten Ansatzes vorgenommen. Dieser Ansatz wurde als Vorlage ausgewählt, da er die Überwachung höherwertiger Dienste unterstützt, Aggregationen von Komponentenparametern ermöglicht und gleichzeitig einfach modifiziert werden kann. Die aus dieser Erweiterung hervorgehende *Service Monitoring Architecture (SMONA)* erlaubt es somit, Daten aus verschiedenen Netz- und Systemmanagementwerkzeugen zu sammeln und gemäß einer SISL-Spezifikation zu Dienstattributen zusammenzustellen.

4.2.4 Nutzungsphase

Einbettung in CIM

Den Abschluss der Methodik bildet die Nutzungsphase. Sie trifft Festlegungen darüber, wie Managementanwendungen auf die Service-MIB bzw. den mit einem Dienst assoziierten Managementinformationen zugreifen können. In Anbetracht der vielfältigen Lösungsansätze für diese Problematik, wie sie bereits durch bestehende Managementmodelle bzw. sie begleitende Protokolle gegeben sind, wird auf die Entwicklung einer eigenen Lösung an dieser Stelle verzichtet. Vielmehr werden Möglichkeiten zur Einbettung der Service-MIB in bestehende Managementarchitekturen diskutiert (Kapitel 7) und exemplarisch anhand des CIM/WBEM Rahmenwerks demonstriert. Abschließend werden in der Nutzungsphase Vorgehensrichtlinien zur szenariospezifischen Anpassung der Service-MIB vorgestellt.

4.3 Zusammenfassung und weiteres Vorgehen

In diesem Kapitel wurde mit dem Service-MIB Ansatz die zentrale Lösungsidee dieser Arbeit präsentiert. Desweiteren erfolgte die Vorstel-

lung einer, aus vier Phasen bestehenden, Methodik zur Realisierung dieses Ansatzes. Sie umfasst die Ermittlung des Informationsbedarfs anhand von ITIL (*Analysephase*), die *Spezifikation* von Dienstmanagementinformation mit Hilfe von SISL und eine in der *Überwachungsphase* entwickelte Architektur (SMONA) zur Auswertung von SISL. Ferner werden in der *Nutzungsphase* notwendige Schritte zur Einbettung der Service-MIB in bestehende Managementarchitekturen am Beispiel von CIM/WBEM aufgezeigt sowie Vorgehensrichtlinien zur szenariospezifischen Anpassung dargelegt. In besonderem Maße findet dabei eine integrative Sichtweise Anwendung: Durch Berücksichtigung der Übergänge zwischen verschiedenen Phasen können die in Kapitel 2 identifizierten Anforderungen umfassend umgesetzt werden.

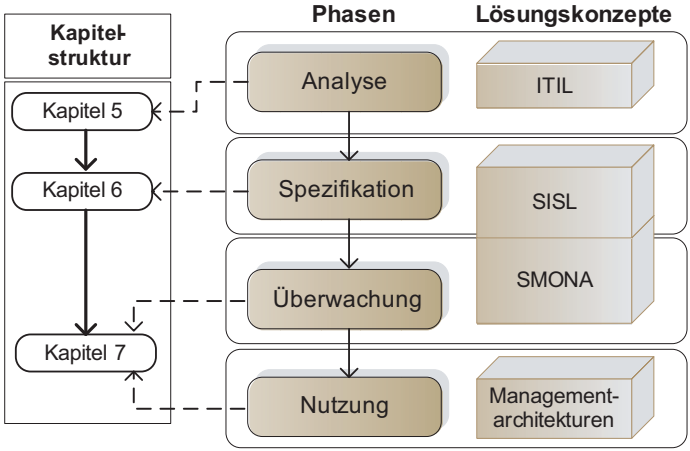

Abbildung 4.5: Abbildung der Methodik auf die weitere Kapitelstruktur dieser Arbeit

Die vorgestellte Methodik bildet gleichzeitig den Rahmen für das weitere Vorgehen in dieser Arbeit: Sukzessive werden Lösungen für die jeweiligen Phasen erarbeitet und somit der Service-MIB Ansatz weiter konkretisiert. In welchen Kapiteln die Beschreibungen der einzelnen Phasen zu finden sind, wird nocheinmal in Abbildung 4.5 dargestellt.

Kapitel 5

Informationsanalyse dienstbezogener Managementinformation

In diesem Kapitel wird der Informationsbedarf an dienstorientierter Managementinformation anhand mehrerer Gesichtspunkte ermittelt und dadurch die Grundlage für eine Spezifikation konkreter Managementinformation geschaffen. Die durchgeführte Informationsanalyse ist notwendig, da im Bereich des Dienstmanagements noch kein Ansatz vorgestellt wurde, der sich an einer Top Down Vorgehensweise orientiert und bis zur Definition konkreter Dienstmanagementinformation reicht. Im Hinblick auf die im letzten Kapitel vorgestellte Methodik wird somit die *Analysephase* beschritten und im Zuge derer das Ziel verfolgt, Informationsanforderungen an die konzeptionellen Inhalte einer Service-MIB zu bestimmen.

Zunächst ist es dazu erforderlich, eine geeignete Vorgehensweise zur Ableitung von Informationsanforderungen festzulegen. In diesem Zusammenhang relevante Konzepte werden im nächsten Abschnitt diskutiert und gleichzeitig die verschiedenen Teilbereiche der Analysephase vorgestellt.

5.1 Vorgehen in der Informationsanalyse

Der Vorgehensweise zur Bestimmung des Informationsbedarfs kommt eine entscheidende Bedeutung innerhalb der Analysephase zu. Sie bildet die Basis für eine Definition konkreter Managementinformation und legt damit letztendlich deren Beschaffenheit und Ausrichtung fest. Wie im Rahmen der Anforderungsanalyse festgestellt, wird hierbei Wert auf eine möglichst einheitliche (KMI 4), nachvollziehbare (KMI 5) und vollständige (KMI 3) Lösung gelegt, die zudem ein hohes Maß an Technologieunabhängigkeit (KMI 1) aufweisen soll.

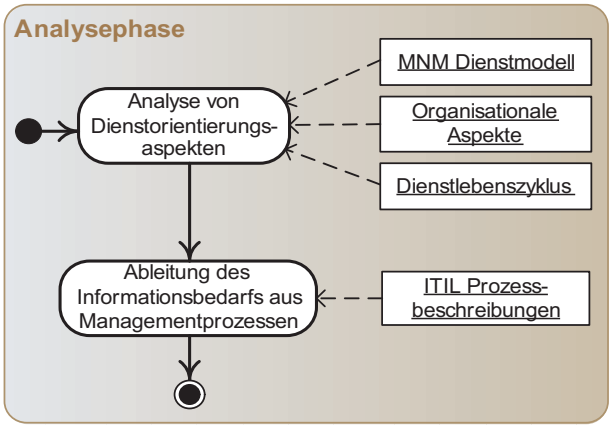

Abbildung 5.1: Vorgehen in der Analysephase

Prinzipiell können in diesem Zusammenhang zwei Herangehensweisen unterschieden werden[HAN99, Neu93]: Eine *Bottom Up* Vorgehensweise orientiert sich an zur Verfügung stehender Managementinformation, die in Form von vorgegebenen Protokollen, Produkten und Komponenten vorliegt und ermittelt darauf basierend managementrelevante Abstraktionen. Demgegenüber geht die *Top Down* Vorgehensweise konsequent der Frage nach, welcher Informationsbedarf auf Seiten von Providern, Managementanwendungen oder Nutzern besteht und versucht diesen zu konzeptualisieren.

Top Down Vorgehen wird gewählt

Um eine möglichst allgemeingültige und vollständige Lösung zu schaffen, wird in dieser Arbeit eine *Top Down* Vorgehensweise präferiert, die sich strikt an den Informationsanforderungen von Dienstbetreibern orientiert. Eine Kernidee besteht dabei darin, Managementprozesse in die Betrachtung aufzunehmen und als Quelle für Informationsanforderungen zu betrachten.

Der Vorteil dieser neuen Herangehensweise zur Ableitung von Managementinformation liegt vor allem in einer stärkeren Verflechtung von technischem Management und den Geschäftszielen der Unternehmung – eine Zielsetzung, die in zunehmendem Maße innerhalb von *IT-Business-Alignment* Maßnahmen verfolgt wird. Neben der Betrachtung von Managementprozessen stützt sich die Analysephase weiterhin auf die im Zuge der Dienstorientierung auftretenden Informationsanforderungen. Letztere repräsentieren durch den Übergang von Komponenten- auf Dienstmanagement induzierte, Rahmenbedingungen (z.B. Einbeziehung von Organisationsgrenzen), die ebenfalls in einer umfassenden Analyse Berücksichtigung finden müssen. Die genannten Aspekte münden in zwei wesentliche Teilschritte der Analysephase, die in Abbildung 5.1 illustriert werden:

- *Informationsanalyse von Dienstorientierungsaspekten*
  Die mit dem Übergang von einer komponenten- zu einer dienstorientierten Betrachtungsweise von IT-Infrastrukturen auftretenden Aspekte bilden den Gegenstand dieses Teilschritts. Zur strukturierten Ableitung von Informationsanforderungen werden dafür MNM-Dienstmodell, organisationale und Dienstlebenszyklus Aspekte analysiert und dabei essentielle Entitäten für ein Service-MIB Modell ermittelt.

- *Ableitung des Informationsbedarfs aus Managementprozessen*
  Wie bereits an mehreren Stellen dieser Arbeit verdeutlicht wurde, soll eine Service-MIB in erster Linie der Unterstützung von Dienstmanagementaufgaben dienen. Anhand von typischen Managementaufgaben die dafür benötigte Managementinformationen zu ermitteln stellt deshalb die Zielsetzung des zweiten Teilschritts der Analysephase dar. Basierend auf Beschreibungen von Managementprozessen innerhalb der bereits im Abschnitt 3 vorgestellten ITIL werden Informationsanforderungen bestimmt und

somit die im vorhergehenden Teilschritt gewonnenen Erkenntnisse verfeinert.

Wie aus den genannten Aspekten deutlich wird, zielt die Analysephase auf eine möglichst umfassende, aber gleichzeitig generische Bestimmung von Informationsanforderungen ab. Auf eine Berücksichtigung von hersteller- und technologiespezifischen Quellen wurde deshalb kategorisch verzichtet und somit auf eine strikte Beibehaltung eines *Top Down* Vorgehens geachtet.

5.2 Informationsanalyse von Dienstorientierungsaspekten

Mit dem Übergang zum Dienstmanagement und der damit einhergehenden Dienstorientierung treten eine Reihe neuer Aspekte auf, die beim Entwurf von Managementlösungen Berücksichtigung finden müssen. Sie fußen vor allem auf einer verstärkt dienstorientierten Sichtweise auf IT-Infrastrukturen und einer gesteigerten Ausrichtung an den Bedürfnissen des Kunden. Statt der isolierten Betrachtung von Einzelkomponenten, rücken damit die einem Kunden erbrachten Dienste in den Mittelpunkt der Betrachtung. Welche Informationsanforderungen für eine Service-MIB durch diese Dienstorientierungsaspekte erwachsen, wird im Folgenden erläutert.

Statische und dynamische Aspekte werden untersucht

Um im Dienstmanagementkontext auftretende Entitäten, Rollen und Relationen strukturiert zu ermitteln finden sowohl statische als auch dynamische Aspekte eines Dienstes in der Analyse Berücksichtigung: Während anhand des MNM-Dienstmodells ein Katalog grundlegender Entitäten und Rollen abgeleitet werden kann, zeigt die Untersuchung des Dienstlebenszyklus vorwiegend den Bedarf an Relationen zwischen Entitäten auf. Abschließend fließen jene Informationsanforderungen in die Analyse mit ein, die aufgrund der zunehmenden organisatorischen Verteilung von Diensten – und dem damit einhergehenden interorganisationalen Dienstmanagement – erwachsen.

5.2.1 Grundlegende Bestandteile eines Dienstes

Dienstmodell spezifiziert grundlegende Entitäten und Rollen

Einer Charakterisierung grundlegender Bestandteile eines Dienstes widmet sich das in [GHK+01, GHH+02] beschriebene MNM-Dienstmodell. Der Erstellung des Modells zugrundeliegende Zielsetzung war, neben der Definition von Schlüsselbegriffen für das Dienstmanagement, die Schaffung eines Universalmodells, das sich auf eine Vielzahl vorhandener Dienste anwenden lässt. Es dient deshalb im Folgenden zur Ermittlung des Informationsbedarfs hinsichtlich im Dienstmanagementkontext benötigter Entitäten und Rollen.

Drei Sichten werden unterschieden

Prinzipiell unterscheidet das MNM-Dienstmodell zwischen verschiedenen Sichten: Ausgehend von einem *Basismodell* wurde eine *Dienst-* und I*mplementierungssicht* konzipiert. Relevant für die Informationsanalyse sind insbesondere die ersten beiden Sichten, die im Folgenden zur Ableitung von Informationsanforderungen herangezogen werden.

5.2.1.1 MNM Basismodell

Das Basismodell stellt die fundamentalen Zusammenhänge zwischen den im Dienstmanagementumfeld auftretenden Entitäten und Rollen dar. Es ist bewusst einfach gehalten und dient als Grundlage zur weiteren Verfeinerung dieser Zusammenhänge in der Dienstsicht. Folgende Aspekte werden in dem in Abbildung 5.2 dargestellten Basismodell illustriert:

Dienst Grundsätzlich wird ein Dienst als eine Funktionalität verstanden, die einem Dienstnehmer an einer Schnittstelle mit einer bestimmten Dienstgüte von einem Provider zur Verfügung gestellt wird.

Verschiedene Zuständigkeitsbereiche lassen sich identifizieren

Rollen und Domänen Wie aus voriger Begriffsdefinition hervorgeht, sind verschiedene Rollen (*user*, *customer*, *provider*) an einem Dienst beteiligt. Typischerweise führen diese Rollen für ihren Zuständigkeitsbereich (*Domäne*) spezifische Aufgaben aus – die Art und Weise der Durchführung bleibt allerdings der jeweilig anderen Domäne größtenteils verborgen. Das MNM-Dienstmodell unterscheidet explizit drei Domänen:

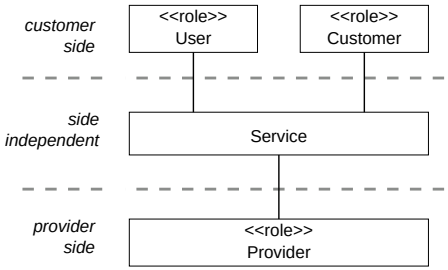

Abbildung 5.2: MNM-Basismodell nach [GHK+01]

- *Dienstleisterdomäne*
  Die Bereitstellung des Dienstes liegt in dem Verantwortungsbereich des Providers und umfasst typischerweise die Implementierung und das Management des Dienstes. Entsprechend werden im MNM-Dienstmodell alle Entitäten, die an der Erbringung des Dienstes gemäß einer Dienstvereinbarung (SLA) beteiligt sind, in der Dienstleisterdomäne zusammengefasst. Grundsätzlich fallen die Grenzen dieses Verantwortungsbereichs mit den Schnittstellen des Dienstes zusammen.

- *Seitenunabhängige Domäne*
  Eine wichtige Richtlinie des MNM-Dienstmodells besteht darin, einen Dienst unabhängig von seiner Implementierung aufzufassen. Dem wird in der Modellierung Rechnung getragen und der Dienst innerhalb der seitenunabhängigen Domäne eingeordnet.

- *Dienstnehmerdomäne*
  Innerhalb der Dienstnehmerdomäne treten die Rollen Dienstnutzer und Dienstkunde auf: Während der Dienstnutzer die Nutzungsfunktionalität des Dienstes an einem definierten Zugangspunkt (*Service Access Point*) in Anspruch nimmt, setzt sich der Dienstkunde vorrangig mit der Verhandlung und Überwachung einer Dienstvereinbarung auseinander. Letzteres geschieht üblicherweise unter Benutzung der vom Provider zur Verfügung ge-

stellten Customer Service Management (CSM) Funktionalität, die es dem Dienstkunden gestattet, Leistungsdaten über den abonnierten Dienst abzurufen und in einem weiteren Schritt mit der zugrundeliegenden Dienstvereinbarung zu vergleichen.

Bevor eine weiterführende Betrachtung der Dienstsicht des MNM-Modells erfolgt, werden zunächst die aus dem Basismodell abgeleiteten Informationsanforderungen vorgestellt.

5.2.1.2 Informationsanforderungen aus dem Basismodell

Grundsätzlich können, basierend auf dem MNM-Dienstmodell, zwei Arten von Informationsanforderungen abgeleitet werden: Einerseits treten Anforderungen hinsichtlich in einer Service-MIB zu berücksichtigender Entitäten und Rollen auf, andererseits vermittelt das MNM-Dienstmodell generische Dienstmanagementkonzepte, die wiederum Auswirkungen auf Designentscheidungen der Service-MIB haben. Entsprechend dieser Unterscheidung werden zunächst Informationsanforderungen in Bezug auf grundlegende Entitäten besprochen, die dann im weiteren Verlauf der Analyse – dem Entwurfsmuster des MNM-Dienstmodells folgend – anhand der Dienstsicht verfeinert werden.

- *Grundlegende Entitäten: Dienst und Kunde*
  Zuordnung zwischen Kunde und Dienst
  Zunächst stellt die Entität Dienst den zentralen und damit unverzichtbaren Bestandteil jeder Definition von dienstorientierter Managementinformation dar und muss damit den Mittelpunkt der Service-MIB bilden. Des Weiteren wird es im Hinblick auf eine Zuordnung zwischen einem Kunden und dem von ihm in Anspruch genommenen Dienst und SLAs notwendig, eine entsprechende Rolle einzuführen. An dieser Stelle muss allerdings beachtet werden, dass die Pflege von Kundendaten nicht als Teil des technischen Dienstmanagements verstanden wird und deshalb nicht die Daten selbst, sondern Verweise auf entsprechende Datenquellen (z.B. LDAP-Repository) berücksichtigt werden müssen. In diesem Zusammenhang erscheint es auch sinnvoll, eine rollenbezogene Entität Dienstnutzer mit in die Service-MIB aufzunehmen. Die mit der im MNM-Dienstmodell eingeführten Rolle Provider

assoziierten Informationsanforderungen werden in Abschnitt 5.2.3 besprochen.

Designrichtlinien für die Service-MIB

Insbesondere mit der Aufteilung des MNM-Dienstmodells in verschiedene Domänen wurde eine für das Dienstmanagement essentielle Sichtweise auf Dienste eingeführt. Daraus entstehende Konsequenzen für die Service-MIB lassen sich in den folgenden zwei Punkten subsummieren:

- *Trennung von Dienstspezifikation und -implementierung*
  Der dienstorientierte Ansatz sieht eine klare Trennung der Funktionalität eines Dienstes von seiner Implementierung vor. Dies bedeutet vor allem, dass sich Dienstnehmer und Provider auf eine mit bestimmten Qualitätsmerkmalen versehene Dienstfunktionalität einigen. Für den Dienstnehmer erübrigt sich damit die Notwendigkeit, Implementierungsdetails zu verhandeln und nachzuvollziehen. Dementsprechend können vertragliche Vereinbarungen (SLAs) auf eine Festlegung von Nutzungs- und Managementfunktionalität beschränkt bleiben und damit stärker an den Bedürfnissen des Kunden orientiert werden. Ferner bietet diese Vorgehensweise für den Provider den Vorteil, Änderungen und Optimierungen innerhalb der Dienstimplementierung durchführen zu können, ohne dabei den SLA verändern zu müssen. Entsprechend sollte eine Service-MIB dieses Konzept abbilden und eine Unterscheidung zwischen Dienstspezifikation und -implementierung ermöglichen.
- *Kundenspezifische Differenzierung*
  Als Konsequenz der verstärkten Kundenorientierung wird es für den Provider in zunehmendem Maße erforderlich, kundenindividuelle Dienstmerkmale in Bezug auf Funktionalität, Dienstgüte und Zugangsschnittstellen zu realisieren. Dies bedeutet vor allem, auf teilweise identischer Infrastruktur (z.B. Backbone-Netz) eine kundenspezifische Differenzierung vorzunehmen. Die damit einhergehende Komplexität muss entsprechend in einer Service-MIB abgebildet und somit die Möglichkeit zur Zuordnung zwischen Dienstinstanz und Kunden geschaffen werden.

Wie bereits erwähnt, dient das Basismodell als Vorlage zu einer weiterführenden Verfeinerung der dargestellten Entitäten, die in der Dienst-

sicht des MNM-Dienstmodells mündet und im Folgenden vorgestellt wird.

5.2.1.3 MNM-Dienstsicht

Dienstsicht erweitert den Dienstbegriff

Gegenüber dem Basismodell wurde in der MNM-Dienstsicht die Domänenaufteilung beibehalten, die darin enthaltenen Entitäten allerdings weiter verfeinert. Insbesondere trifft dies auf die Entität Dienst zu, die um Assoziationen und Kompositionsbeziehungen zu weiteren, neu definierten Entitäten angereichert wurde. Abbildung 5.3 stellt die Dienstsicht des MNM-Modells dar, dessen Elemente im Folgenden erklärt werden:

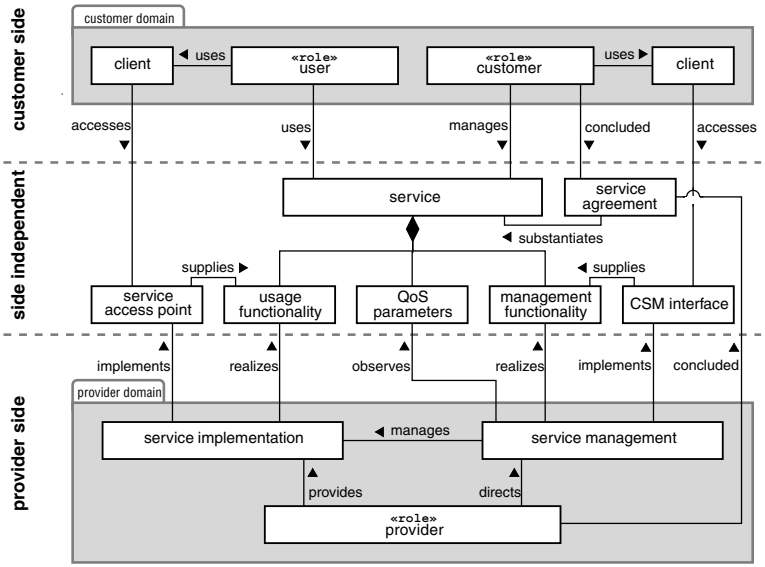

Abbildung 5.3: Dienstsicht des MNM-Dienstmodells nach [GHK+01]

Funktionalität Die Funktionalität eines Dienstes wird durch eine festgelegte Reihe von Interaktionen zwischen Dienstleister und Dienstnehmer charakterisiert. Interaktionen können z.B. auf Anwendungsaufrufe, Nutzung von Übertragungsprotokollen oder Workflows abgebildet werden. Ferner wird im MNM-Dienstmodell zwischen Funktionalitäten unterschieden, die vorwiegend der Nutzung eines Dienstes dienen und sogenannter Managementfunktionalität.

- *Nutzungsfunktionalität*
  Interaktionen zwischen Dienstleister und Dienstnehmer, die dem eigentlichen Zweck des Dienstes dienen, werden unter Nutzungsfunktionalität subsummiert. Diese Funktionalität wird vorrangig vom Dienstnutzer konsumiert.

- *Managementfunktionalität*
  Weitere Interaktionen, die primär für den Betrieb des Dienstes notwendig sind, werden zur Managementfunktionalität zusammengefasst. Darunter fallen beispielsweise der Betrieb einer Hotline oder die Bereitstellung von Abrechnungs- und Leistungsdaten bezüglich des Dienstes. Diese Funktionalität zeichnet sich weiterhin dadurch aus, dass nur ein festgelegter Teil von Operationen und Informationen kontrolliert an der Customer Service Management Schnittstelle zur Verfügung gestellt wird – ein uneingeschränkter Zugriff auf das Dienstmanagement des Providers wird in der Regel nicht ermöglicht.

Dienstgüteparameter bilden Grundlage für SLA

Dienstgüteparameter Entscheidend für den Übergang zum Dienstmanagement ist die Einführung eines Qualitätsbegriffs für den Dienst. Ziel dieser Herangehensweise ist es, die Qualität der Diensterbringung durch eine festgelegte Menge von Dienstgüteparametern beschreibbar und messbar zu machen [Gar04b]. Üblicherweise werden dazu innerhalb des SLAs zwischen Kunde und Provider Grenzwerte für diese Parameter vereinbart und Pönalen, falls diese nicht eingehalten werden können.

Client Auf Dienstnehmerseite ermöglichen Clients die Nutzung von Dienstfunktionalitäten über Dienst- und CSM-Zugangspunkte. Um

einen erfolgreichen Zugriff auf diese Schnittstellen sicherzustellen, werden typischerweise Systemvoraussetzungen (z.B. unterstützte Browserversionen) in der Dienstvereinbarung spezifiert, die Verantwortung für den Betrieb des Clients liegt allerdings auf Dienstnehmerseite.

Schnittstellen Zur Nutzung der Funktionalität des Dienstes durch den Dienstnehmer stehen wohldefinierte Schnittstellen zur Verfügung. Diese spiegeln die Aufteilung der Dienstfunktionalität wider und gliedern sich konsequenterweise in Dienstzugangspunkt und Customer Service Management (CSM) Schnittstelle:

Schnittstellen bieten Zugriff auf Dienstfunktionalität

- *Dienstzugangspunkt*
  Am Dienstzugangspunkt wird dem Dienstnutzer die in dem Service Level Agreement vereinbarte Nutzungsfunktionalität zur Verfügung gestellt.
- *Customer Service Management Schnittstelle*
  Über die Customer Service Management Schnittstelle greift der Dienstkunde auf die vereinbarte Managementfunktionalität zu. Die CSM Schnittstelle kann auf verschiedene Arten realisiert werden, denkbar sind eine wohldefinierte Programmierschnittstelle (API), aber auch gängige Kommunikationsmittel wie Email.

Dienstvereinbarung Die Dienstvereinbarung (SLA) stellt ein vertragliches Rahmenwerk zwischen Dienstkunde und Provider dar. Sie enthält eine Spezifikation der zu erwartenden Dienstfunktionalität, der Dienstgüteparameter sowie der Dienstschnittstellen [Lew99]. Typischerweise werden in einem SLA auch Vertragsstrafen vereinbart, zu deren Leistung sich der Provider verpflichtet, falls er sich nicht in der Lage sieht, die vereinbarte Dienstqualität zu liefern.

Dienstvereinbarung wird in einem rechtsgültigen Vertrag niedergelegt

Dienstimplementierung Mit der Dienstimplementierung wird primär die Realisierung der Nutzungsfunktionalität des betreffenden Dienstes beschrieben. Ferner fällt darunter auch die Implementierung der Nutzungsschnittstelle, die dem Dienstnutzer Zugriff auf die vereinbarte Funktionalität bietet. Neben technischen Lösungen umfasst die

Dienstimplementierung auch Managementwissen des beteiligten Personals oder Hard- und Software, die zur Realisierung des Dienstes erforderlich ist.

Betrieb des Dienstes

Dienstmanagementimplementierung Die Dienstmanagementimplementierung widmet sich der Planung, Installation und dem Betrieb des Dienstes innerhalb der vertraglich vereinbarten Qualitätsparameter. Dafür müssen u.a. eine Dienstüberwachung vorgenommen und – basierend auf den ermittelten Leistungsdaten des Dienstes – Managementaktionen durchgeführt werden. Ferner stellt die Implementierung der CSM-Schnittstelle einen Bestandteil der Dienstmanagementimplementierung dar.

5.2.1.4 Informationsanforderungen aus der MNM-Dienstsicht

Aufgrund der durch die Dienstsicht vermittelten, verfeinerten Darstellung von Dienstmanagementkonzepten konnten, zusätzlich zu den in Abschnitt 5.2.1.2 genannten Aspekten, weitere Informationsanforderungen gewonnen werden:

- *Berücksichtigung von Dienstqualität und Dienstvereinbarungen*
  Während in der Vergangenheit IT-Leistungen oftmals nach dem Best-Effort-Prinzip bereitgestellt wurden, wird zunehmend die Qualität des erbrachten Dienstes als wichtiges Differenzierungsmerkmal zwischen verschiedenen Providern angesehen. Aus Kundensicht scheint diese Entwicklung verständlich: Sollen für die Unternehmung geschäftskritische IT-Dienstleistungen an Dritte ausgelagert werden, erscheint eine festgelegte, garantierte Dienstgüte – und deren vertragliche Absicherung in einem SLA – unverzichtbar. Für die Service-MIB bedeutet dies vor allem, dass die entsprechenden Entitäten SLA und QoS abgebildet und mit dem Dienst verknüpft werden müssen. Ferner ergibt sich die Notwendigkeit die Schnittstelle dem Kunden gegenüber, den Dienstzugangspunkt, als weitere Entität mitaufzunehmen

- *Weitergabe von Managementinformationen an Kunden*
  Zur kontrollierten Weitergabe von Managementinformationen an

Kunden steht im MNM-Dienstmodell die CSM-Schnittstelle zur Verfügung. Um diesen Vorgang zu unterstützen, muss die Service-MIB sowohl entsprechende Basisdaten bereitstellen (z.B. zur Erstellung von Leistungsberichten) als auch Managementinformationen zu der Schnittstelle dem Kunden gegenüber (SAP) verwalten.

- *Beschreibung von Funktionalitäten als Teil der Dienstspezifikation*
  Die mit dem Dienst verknüpfte Nutzungs- und Managementfunktionalität dokumentiert Soll-Zustände bezüglich der Dienstbereitstellung und sollte entsprechend innerhalb einer Service-MIB Berücksichtigung finden. Im Hinblick auf die im letzten Abschnitt geforderte Trennung von Dienstspezifikation und -implementierung bietet sich eine Einordnung von Funktionalitätsbeschreibungen als Teil der Spezifikation an.

Nachdem vorwiegend statische Aspekte der Beschreibung von Diensten anhand des MNM-Dienstmodells betrachtet wurden, beschäftigt sich der nächste Schritt der Analyse mit er Ermittlung von Informationsanforderungen entlang des Dienstlebenszyklus.

5.2.2 Managementaufgaben entlang des Dienstlebenszyklus

Phasen des Dienstlebenszyklus

Geht man der Frage nach, welche Managementinformationen für ein umfassendes Dienstmanagement benötigt werden, stellt sich eine exklusive Betrachtung während des Betriebs anfallender Aufgaben als nicht adäquat heraus. Vielmehr ist es notwendig, Aspekte zu berücksichtigen, die beispielsweise während der Planung, der Verhandlung zwischen Kunde und Provider oder bei der Beendigung eines Dienstes auftreten. Dieser Forderung wird eine lebenszyklusbasierte Betrachtung gerecht [HAN99, DR02]. Die vorrangige Idee dieser Herangehensweise besteht darin, den Lebenszyklus in Phasen, die nach typischen Aktionen innerhalb dieser Phase benannt sind, einzuteilen (siehe Abbildung 5.4).

In der vorliegenden Arbeit werden die Planungs-, Vereinbarungs-, Bereitstellungs-, Betriebs- und Beendigungsphase berücksichtigt und

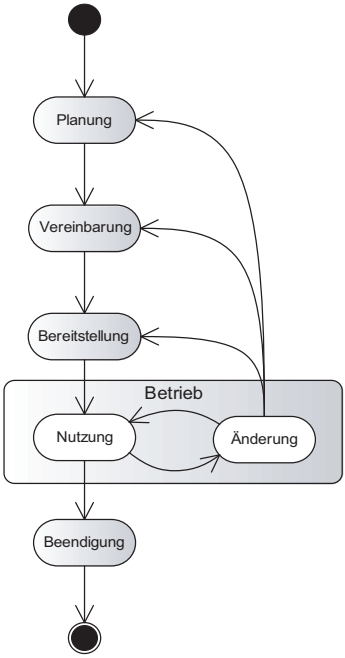

Abbildung 5.4: Dienstlebenszyklus

im Folgenden mit Bezug auf das in Abschnitt 2.2.1 vorgestellte Web-Hosting Szenario weiter ausgeführt. Gleichzeitig finden Informationsaspekte Erwähnung, die eine Service-MIB erfüllen muss bzw. wo sie unterstützend wirken soll.

5.2.2.1 Lebenszyklusphase Planung

Wird ein neuer Dienst von einem (potentiellen) Kunden nachgefragt oder entschließt sich der Provider einen neuen Dienst anzubieten, beginnt die Planungsphase. Beispielsweise könnte eine Forschungseinrichtung einen webbasierten Dienst mit neuer Funktionalität wünschen oder

das LRZ sich entschließen, erstmalig einen weiteren Dienst anzubieten. In beiden Fällen müssen Planungsschritte dahingegend vorgenommen werden, dass resultierende Konsequenzen antizipiert und ermittelt werden. Dies umfasst eine Festlegung, welche Subdienste benötigt werden, ob gegebenenfalls ein neuer Dienst implementiert oder ein bestehender modifiziert werden soll, aber auch welche Qualitätsansprüche für die an der Diensterbringung beteiligten Komponenten berücksichtigt und überwacht werden müssen. Letzteres mündet in der Definition aussagekräftiger Dienstkennzahlen, die im weiteren Verlauf des Lebenszyklus die Basis für eine vertragliche Vereinbarung mit dem Kunden darstellen. Zudem muss die Festlegung der Ende-zu-Ende Diensterbringungskette erfolgen und eine Einbindung in bestehende Managementsysteme geplant werden. Das Ergebnis dieser Schritte stellt eine detaillierte Dienstspezifikation dar, die im weiteren Verlauf eine kundenspezifische Verfeinerung erlaubt.

Informationsanforderungen in der Planungsphase

Dienstspezifikation entsteht

Die Service-MIB soll auf zwei Arten unterstützend auf die Planungsphase einwirken: Einerseits stellt sie Informationen zu bestehenden Diensten bereit, zeigt Verkehrsgrößen und Trendvorhersagen vorhandener Dienste auf und hilft somit eine Kapazitätsplanung vorzunehmen. Andererseits hat eine Service-MIB umfangreiche Templates für die Erfassung der Planungsinformationen bereitzustellen; die Struktur muss intuitiv sein, um von den in der Planungsphase beteiligten Personen leicht erfasst und ausgefüllt werden zu können. Weiterhin sollte eine Verfeinerung im Hinblick auf die Spezifitäten des betreffenden Dienstes ohne Weiteres möglich sein.

Die entstehende Dienstspezifikation wird üblicherweise von den an der Diensterbringung beteiligten Abteilungen komplementiert. Insbesondere müssen Zusammenhangsinformationen zwischen Diensten und Komponenten aufgenommen und damit für spätere Managementaufgaben verfügbar gemacht werden. Nicht zuletzt soll die in dieser Phase entstehende Dienstspezifikation eine Basis für eine spätere Implementierung des Dienstes bieten.

5.2.2.2 Lebenszyklusphase Vereinbarung

Wurden die in der Planungsphase anfallenden Schritte erfolgreich beendet, erfolgen konkrete Verhandlungen zwischen Kunde und Provider bezüglich der zu erwarteten Qualität des Dienstes. Ziel dieser Phase ist es, vertragliche Vereinbarungen zu treffen und in einem Service Level Agreement festzuhalten. Letzteres beinhaltet Übereinkünfte, welche Qualitätsparameter der Dienst erfüllen soll, Grenzwerte und Schwellenbereich zu diesen Parametern sowie Eskalationsmechanismen und Pönalen, falls die vereinbarte Dienstqualität nicht erreicht wird. Zusätzlich finden sich oft Festlegungen, wann und in welcher Form der Provider dem Kunden Informationen über den Dienst zur Verfügung stellen soll oder welche Verfahren bei der Messung der Dienstqualität eingesetzt werden. Typische Beispiele für Qualitätsparameter – die auch für den Web-Hosting Dienst in Betracht kommen – stellen die Verfügbarkeit des Dienstes pro Monat, die maximal zulässige Antwortzeit oder die durchschnittliche Wiederherstellungszeit (*Mean Time to Repair*) dar.

Informationsanforderungen in der Vereinbarungsphase

Kundenspezifische Verfeinerung wird vorgenommen

Nachdem eine generelle Dienstspezifikation in der Planungsphase kreiert wurde, muss darauf basierend eine Instanzierung und damit kundenspezifische Verfeinerung vorgenommen werden können. Dies gilt insbesondere für die in dem SLA vereinbarten Qualitätsparameter und Pönalen, die nun mit der betreffenden Dienstinstanz assoziiert werden. Damit soll es die Service-MIB im späteren Betrieb des Dienstes ermöglichen, eine Überprüfung dahingehend vornehmen zu können, ob die aktuell gemessene Dienstqualität mit der vereinbarten übereinstimmt. Weiterhin soll mit der in diesem Schritt entstehenden kundenspezifischen Dienstspezifikation eine Grundlage für die Implementierung des Dienstes geschaffen werden.

5.2.2.3 Lebenszyklusphase Bereitstellung

Technische Umsetzung

Nachdem in den letzten beiden Phasen die Voraussetzungen für eine technische Umsetzung des Dienstes geschaffen wurden, wird dieser in

der Bereitstellungsphase realisiert: Geeignete Dienstkomponenten und Subdienste werden identifiziert und alloziiert, die Ende-zu-Ende Diensterbringungskette entsteht. Damit verbunden muss die Abbildung der für den Dienst vereinbarten Qualitätsmerkmale auf die Dienstkomponenten erfolgen. Anders ausgedrückt, es bedarf der Festlegung, innerhalb welcher Parameter die einzelnen Ressourcen operieren müssen, um die Dienstqualität zu erfüllen (vgl. [DR02]). Für den Web-Hosting Dienst umfasst eine solche Festlegung die Antwortzeiten von Datenbank und Storage. Da diese unmittelbar die Antwortzeit des WebHosting-Dienstes beeinflussen, müssen sie in jedem Falle ein besseres Antwortverhalten aufweisen als für den Dienst im SLA vereinbart wurde. Gemäß der organisatorischen Aufteilung der Diensterbringung sind die jeweiligen Abteilungen verantwortlich, diese Vorgaben einzuhalten, und können gegebenfalls auch vertraglich daran gebunden werden. Ebenfalls muss in der Bereitstellungsphase eine Einbindung in bestehende Managementsysteme vollzogen werden um einen reibungslosen Betrieb des Dienstes zu gewährleisten.

Informationsanforderungen in der Bereitstellungsphase

Abhängigkeitsmodell wird abgebildet

Die bestehende Dienstspezifikation wird mit den technischen Details, die in der Bereitstellungsphase bekannt werden, angereichert. Insbesondere müssen die in der Bereitstellungsphase entstehende Dienst-Topologie in die Service MIB abgebildet und die Beziehungen zwischen den auftretenden Elementen formalisiert werden können, um sie in der nachfolgenden Betriebsphase Managementsystemen zur Verfügung zu stellen. Wie aus der Dienst-Topologie für den Web-Hosting Dienst (Abbildung 5.5) ersichtlich wird, müssen dabei verschiedene Arten von Abhängigkeiten unterschieden werden können: Einerseits treten *Dienstabhängigkeiten* zwischen dem Web-Hosting Dienst und den assoziierten Subdiensten auf (*besteht-aus* Relation), andererseits werden Dienste durch eine (Sammlung) von Netz- und Applikationskomponenten realisiert (gestrichelte Linien). In jedem Fall muss es ermöglicht werden, die Semantik dieser Abhängigkeiten formal darzustellen. Eine solche formale Beschreibung umfasst Auswirkungen von Fehlern in Subdiensten auf den Dienst bzw. von Komponentenfehlern auf Dienste. Die Komplexität der Abhängigkeitsmodellierung variiert dabei mit dem

Anwendungsfall: Während einfache funktionale Abhängigkeiten lediglich ausdrücken, wie sich der vollständige Ausfall einer Komponente auf den Status des Dienstes auswirkt, wird eine feingranulare Modellierung notwendig, um beispielsweise die erwartete Verminderung der Dienstqualität bei bestimmten Komponentenfehlern ausdrückbar zu machen.

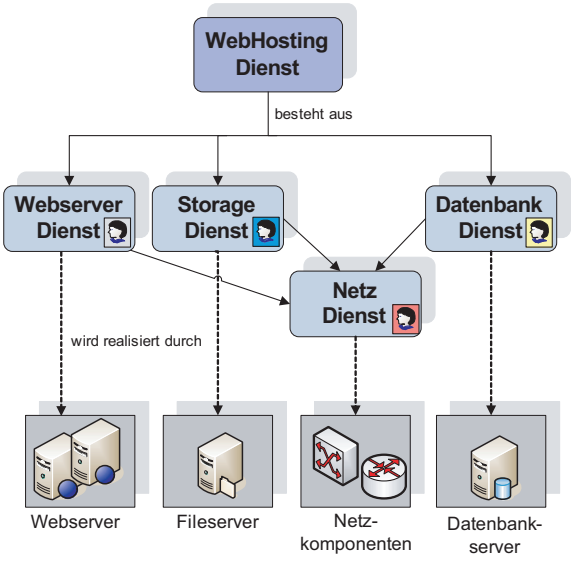

Abbildung 5.5: Topologie des Web-Hosting Dienstes

5.2.2.4 Lebenszyklusphase Betrieb

Wurden die vorhergehenden Phasen erfolgreich durchgeführt, steht der Dienst dem Kunden in der Betriebsphase zur Verfügung. Während sich der Kunde in der Regel vorwiegend an der Nutzung der vereinbarten Funktionalitäten interessiert zeigt, liegt die Herausforderung für den Provider vor allem darin, eine korrekte Funktionsweise des Dienstes sicherzustellen. Die dafür notwendigen Maßnahmen lassen sich in zwei

Teilaktivitäten (siehe Abbildung 5.4) zerlegen: Während in der Nutzungsphase vor allem die Überwachung (Monitoring) der Dienste und Komponenten im Hinblick auf eine Einhaltung der vereinbarten Dienstqualität im Vordergrund steht, werden gleichzeitig immer wieder Anpassungen am Dienst notwendig (Änderungsphase).

Die in der Nutzungsphase durchgeführten Überwachungsmaßnahmen zielen auf eine (frühzeitige) Erkennung von Fehlern und Kapazitätsengpässen ab und sollen damit Verletzungen von Service Level Agreements vermeiden. Dazu ist es erforderlich, die dienstrealisierenden Komponenten zu überwachen und die gemessenen Werte in Beziehung mit dem Dienst zu setzen. Voraussetzung dafür ist die bereits erwähnte Dienstsicht, die eine akkurate Repräsentation des Dienstes sowie der Abhängigkeiten zwischen Dienst und Komponenten bzw. der Kritikalität der Komponenten bezüglich des Dienstes vermitteln soll. Der Umfang der in dieser Phase anfallenden Managementaufgaben ist zu komplex, um an dieser Stelle erschöpfend dargestellt zu werden. Für weitere Ausführungen sei hier auf Kapitel 5.3 oder [HAN99] verwiesen. Beispiele für weitere Aufgaben stellen die dienstorientorientierte Eventkorrelation [Han07] und der Impact-Analyse [HSS05b] dar. Während bei der Eventkorrelation, ausgehend von Fehlermeldungen über den Dienst, die betreffenden fehlerhaften Komponenten lokalisiert werden, beschreitet die Impact-Analyse den umgekehrten Weg. Ausgehend von Fehlermeldungen über Ressourcen werden hierbei die davon betroffenen Dienste und der dadurch (potentiell) entstehende Schaden identifiziert. Offensichtlich stellt die vorher erwähnte Dienstsicht eine notwendige Voraussetzung für beide Ansätze dar.

Treten Probleme mit dem Dienst oder Änderungswünsche des Kunden auf, wird es notwendig, Änderungen an der Dienstimplementierung durchzuführen (Änderungsphase). In den meisten Fällen kann danach mit der Nutzungsphase weiter verfahren werden. Sind die vorzunehmenden Änderungen allerdings zu einschneidend, muss auf eine vorhergehende Phase zurückgegangen und eine erneute Durchführung der dort notwendigen Schritte veranlasst werden. Wünscht beispielsweise eine Universität schnellere Reaktionszeiten für ihren Webdienst, muss die Verhandlungsphase erneut initiiert und eventuell die Dienstimplementierung in der Bereitstellungsphase modifiziert werden, bevor der Dienst wieder zur Verfügung steht.

Einen weiteren Aspekt in der Betriebsphase stellt das Reporting (Berichterstellung) gegenüber dem Kunden dar. Letzterer möchte über den von ihm bestellten Dienst informiert werden, ist aber in der Regel weniger an technischen Details interessiert. Vielmehr möchte er wissen, ob der Dienst im Einklang mit den im SLA festgelegten Kennzahlen betrieben wurde. Wie bereits anhand des MNM-Dienstmodells (Abschnitt 5.2.1) verdeutlicht, wird die Weitergabe dieser Informationen als Teil des Customer Service Managements angesehen und muss durch Schaffung entsprechender Schnittstellen unterstützt werden.

Informationsanforderungen in der Betriebsphase

Service-MIB soll aktuellen Zustand des Dienstes repräsentieren

Wie bereits aus der obigen Ausführung hervorgeht, ist für die in der Betriebsphase anfallenden Dienstmanagement-Aufgaben eine Dienstsicht von essentieller Bedeutung. Entsprechend liegt die vorrangige Aufgabe der Service-MIB darin, diese Dienstsicht zu repräsentieren und beispielsweise aktuelle Informationen bezüglich der Dienst-Topologie, den aktuell gemessenen Qualitätsparametern des Dienstes, der Anzahl der Benutzer, bekannter Ausfälle in Dienstkomponenten, Wartungsfenster, aber auch Ansprechpartner für Dienstkomponenten bereitzustellen.

Für eine ausführliche Darstellung der in diesem Zusammenhang benötigter Managementinformationen wird auf den zweiten Teil der Analysephase, der Untersuchung von Managementaufgaben anhand von Prozessbeschreibungen in Abschnitt 5.3 verwiesen.

5.2.2.5 Lebenszyklusphase Beendigung

Wird ein Dienst nicht mehr benötigt oder durch einen neuern abgelöst, beschreitet man die Beendigungsphase. Dabei werden für die Dienstererbringung verwendete Ressourcen freigegeben, das zwischen Kunde und Provider geschlossene SLA erlischt. Ebenso erfolgt eine Rekonfiguration der Managementsysteme und die Überwachung des Dienstes wird beendet.

Informationsanforderungen in der Beendigungsphase

Dienst wird als beendet markiert

Die betreffende Dienstinstanz wird als beendet markiert, die dazugehörigen Daten werden gelöscht bzw. archiviert. Letzteres kann aus rechtlichen Gründen erforderlich sein oder dem Provider als Referenzmaterial zur Kapazitätsabschätzung für neue Dienste dienen.

Abschließend zu der Analyse von Dienstorientierungsaspekten werden nun im Folgenden Übergange von Organisationsgrenzen und die daraus resultierenden Konsequenzen für die Service-MIB betrachtet.

5.2.3 Berücksichtigung von Organisationsgrenzen

Eine weitere Quelle von Informationsanforderungen an die Service-MIB stellen interorganisationale Aspekte dar. Die Beücksichtigung dieser Aspekte wird notwendig, da Dienste verstärkt über Providergrenzen hinweg erbracht werden. Dies tritt zum einen in Szenarien auf, die inhärent auf eine kooperative Diensterbringung ausgerichtet sind, wie z.B. das in Abschnitt 2.2.2 geschilderte GRID-Computing Beispiel. Zum anderen müssen Übergänge zwischen Organisationsgrenzen im Rahmen von *Outsourcing*, *Outtasking* und *Peering*-Bestrebungen berücksichtigt werden. Während sich im Falle des Outsourcings ein Unternehmen entschließt, die Verantwortung für die Diensterbringung vollständig an ein anderes Unternehmen abzugeben, werden beim Outtasking nur Teile der Diensterbringung an Dritte ausgelagert, meistens unter Beibehaltung der Gesamtverantwortlichkeit für den Dienst [Sch01]. Peering hingegen bezeichnet den Fall, dass verschiedene Provider eine Kopplung ihrer Dienste vornehmen (z.B. Weiterleitung von Datenverkehr zwischen Internetprovidern).

Für die Service-MIB resultieren Informationsanforderungen dahingehend, dass Dienst- und Beziehungsinformationen um organisationale Aspekte angereichert und Beschreibungsmöglichkeiten für technische Parameter der Dienstkopplung geschaffen werden müssen:

- *Organisationsbezogene Informationen*
  Zunächst müssen in der Service-MIB entsprechende Modellierungskonzepte geschaffen werden, die es erlauben, Kompositionen

von Diensten über Organisationsgrenzen hinweg darzustellen. Dazu ist es erforderlich, Dienste und deren Abhängigkeiten mit organisationsbezogenen Informationen zu attributieren bzw. ihnen verantwortliche Rollen zuzuordnen. Somit können z.B. Leistungsstatistiken zu ausgelagerten Diensten verwaltet oder im Fehlerfall schnell Kontaktinformationen ermittelt werden. Grundsätzlich muss dabei allerdings beachtet werden, dass die Pflege von organisationsbezogenen Daten in der Regel außerhalb der Service-MIB erfolgt *(*z.B. in *Enterprise Databases)* und letztendlich nur in Form von entsprechenden Verweisen mitaufgenommen werden sollte.

- *Beschreibung von Koppeldiensten*
  Sollen Dienste interorganisational erbracht werden, wird eine Kopplung von (meist verschiedenen) Architekturen und Werkzeugen notwendig. Dies umfasst beispielsweise die Festlegung von geeigneten Schnittstellen oder eine Abbildung von Dienstgütemerkmalen, um die Qualitätseigenschaften eines Dienstes über Organisationsgrenzen hinweg sicherzustellen [Roe05]. Diesbezügliche Funktionalitäten können wiederum selbst als *Koppeldienst* aufgefasst werden und sollten konsequenterweise innerhalb der Service-MIB beschrieben und somit für das Service Management erfassbar gemacht werden.

Nachdem mit der Betrachtung interorganisationaler Gesichtspunkte der erste Teil der Informationsanalyse abgeschlossen wurde, erfolgt nun eine Zusammenfassung der gewonnenen Ergebnisse.

5.2.4 Zusammenfassung der Informationsanforderungen aus Dienstorientierungsaspekten

In den vorhergehenden Abschnitten die sich mit der Analyse von Dienstorientierungsaspekten beschäftigten, wurde eine Reihe von Informationsanforderungen identifiziert und ausführlich beschrieben. Im Hinblick auf eine Spezifikation von Dienstmanagementinformationen ist es nun notwendig, diese Anforderungen dahingehend zu konsolidieren, dass sie einfach in Managementobjekte überführt werden können.

Gegenüberstellung von Entitäten und Anforderungen

Dieser Zielsetzung dient der in den Tabellen 5.1 und 5.2 dargestellte Katalog. Er entstand durch eine Einordnung der abgeleiteten Informationsanforderungen entlang von Entitäten und dient als Grundlage für die Definition konkreter Dienstmanagementinformationen in Kapitel 6. Im Katalog vorhandene Entitäten weisen ferner eine strukturelle Ähnlichkeit zu dem MNM-Dienstmodell auf, beziehen sich aber ebenso auf anhand der Analyse von Lebenszyklus und organisationalen Aspekten gewonnenen Anforderungen. Insbesondere zu erwähnen sind dabei die folgenden Gesichtspunkte:

Unterscheidung zwischen Dienst und Dienstspezifikation Wegen der Bedeutung einer Dienstpezifikation für die Planungs- und Verhandlungsphase eines Dienstes erfolgte die Einführung einer eigenen Entität. Sie soll die erwarteten Eigenschaften des Dienstes beschreiben und somit dessen Implementierung unterstützen. Wie aus der Beschreibung von Lebenszyklusaspekten weiterhin deutlich wurde, gilt es dabei, zwischen allgemeinen und kundenspezifischen Dienstspezifikationen zu unterscheiden.

Berücksichtigung von Abhängigkeiten zwischen Dienst und Komponenten Wie aus der Analyse von Lebenszyklusaspekten hervorgeht, stellt die Erfassung von Abhängigkeiten zwischen dem Dienst und den dienstrealisierenden Komponenten eine wichtige Prämisse für die Unterstützung der Betriebsphase dar. Mit der Hinzunahme der Entität Dienstkomponente wurde die Voraussetzung dafür geschaffen, die genannten Relationen darzustellen.

Entität	Beschreibung
Dienst	Diese Entität soll die in der Bereitstellungsphase enstehende konkrete Implementierung eines Dienstes repräsentieren und aktuell gemessene Werte für Diensteigenschaften aufnehmen. Weiterhin müssen die Lebenszyklusphase, in der sich der Dienst befindet, festgehalten und Gründe für eine Änderung der Phase dokumentiert werden. *Relationen:* Es müssen Beziehungen zu den Entitäten Dienstspezifikation, SLA, QoS, Dienstkunde- und -nutzer beschrieben werden. Darüber hinaus ist es nötig, Abhängigkeiten zu anderen Diensten und zu dienstrealisierenden Komponenten zu dokumentieren. In diesem Zusammenhang erscheint eine Unterscheidung zwischen funktionalen, organisatorischen, topologischen und temporalen Abhängigkeiten wünschenswert.
Dienstspezifikation	Mit der Dienstspezfikation soll eine Vorlage zur Implementierung eines Dienstes geschaffen werden. Sie soll gewünschte Eigenschaften des Dienstes beschreiben und somit insbesondere während der Planungsphase entstehende Informationen aufnehmen. Insbesondere soll somit ein Soll-Ist Vergleich während des laufenden Betriebes des Dienstes ermöglicht werden. Zu unterscheiden gilt es hierbei allgemeine und kundenspezifische Dienstspezifikationen.

	Relationen: Es sollten Relationen zu der Spezifikation von Qualitätseigenschaften des Dienstes hergestellt werden. Im Falle einer kundenspezifischen Dienstspezifikation müssen ferner Dienstkunden und -nutzer assoziiert werden. Um den gewünschten Soll-Ist Vergleich zu ermöglichen, sollte weiterhin ein Bezug zu dem Dienst, der die entsprechende Spezifikation implementiert geschaffen werden.
Dienstzugangspunkt (SAP)	Der Dienstzugangspunkt beschreibt die Schnittstelle zwischen dem Dienstnutzer und dem Dienst. Dies umfasst neben technischen Parametern des SAP, organisatorische Informationen (z.B. Ansprechpartner). Ferner muss gleichermaßen die CSM-Schnittstelle berücksichtigt werden. *Relationen:* Die Beziehungen zwischen Dienst, Dienstzugangspunkt und Nutzer bzw. Kunde müssen erfasst werden.
Dienstrealisierende Komponenten	Obwohl die Pflege komponentenorientierter Managementinformation nicht als Aufgabe der Service-MIB angesehen wird, muss sie doch ermöglichen, entsprechende Verweise auf diese Daten zu erstellen. *Relationen:*Die Abhängigkeiten zwischen dem Dienst und den dienstrealisierenden Komponenten müssen erfasst werden. Hierbei sollte ebenfalls zwischen funktionalen, organisatorischen, topologischen und temporalen Abhängigkeiten unterschieden werden.

Tabelle 5.1: Konkretisierung der Informationsanforderungen in Entitäten

Hinzunahme von Organisationen Die Möglichkeit, den Dienst mit organisationsbezogenen Informationen zu attributieren, bildet eine zentrale Informationsanforderung der Analyse organisationaler Aspekte. Entsprechend wurden Entitäten geschaffen, die derartige Zusammenhänge darstellbar machen (siehe Tabelle 5.2).

Entität	Beschreibung
Organisation	Organisationsbezogene Informationen sollen als Verweise in die Service-MIB aufgenommen und damit dokumentiert werden können, von welcher Organisation ein Dienst erbracht wird bzw. welcher Organisation die Rollen Dienstnutzer und -kunde angehören. *Relationen:* Eine Zuordnung zu den Rollen Provider, Dienstkunde- und -nutzer sollte möglich sein.
Dienstkunde	Die Rolle Dienstkunde, die innerhalb der vertraglichen Vereinbarungen mit dem Provider auftritt und einer bestimmten Organisation zugeordnet ist, muss beschrieben werden. *Relationen:* Die Beziehungen zu Dienst und SLA müssen beschrieben und der Dienstkunde mit der entsprechenden Organisation verknüpft werden. Ferner sollte eine Zuordnung zu der Dienstschnittstelle möglich sein.
Dienstnutzer	Die Nutzer des Dienstes müssen erfasst und ebenfalls einer Organisation zugeordnet werden. *Relationen:* Die Nutzung der Dienstschnittstelle muss dokumentiert und die Entitäten Dienstnutzer und Organisation müssen miteinander verknüpft werden.

Tabelle 5.2: Konkretisierung der Informationsanforderungen in Entitäten (*Fortsetzung*)

Auf die Darstellung von Informationsanforderungen zu den Entitäten QoS und SLA wurde an an dieser Stelle bewusst verzichtet. Sie werden ausführlich in der Analyse von Managementprozessen behandelt und entsprechend in der Zusammenfassung der daraus gewonnenen Ergebnisse erläutert.

Der in Abschnitt 5.1 skizzierten Vorgehensweise folgend wird nun der zweite Teil der Informationsanalyse beschritten und somit eine Verfeinerung der ermittelten Informationsanforderungen bzw. des soeben vorgestellten Katalogs vorgenommen.

5.3 Ableitung des Informationsbedarfs aus Managementprozessen

Die wohl vorrangigste Zielsetzung der Service-MIB besteht darin, den Betrieb eines Dienstes umfassend zu unterstützen. Entsprechend muss sich die Informationsanalyse maßgeblich an der Fragestellung ausrichten, welche Dienstinformationen zur Erfüllung von Managementaufgaben benötigt werden. In Abschnitt 5.2.2 wurden bereits eine Reihe von Aufgaben im Kontext des Dienstlebenszyklus genannt, ohne allerdings Anspruch auf Vollständigkeit zu erheben. Jedoch stellt exakt diese Vollständigkeit eine wichtige Anforderung für die Service-MIB dar, sollen doch die darin enthaltenen Informationen möglichst viele Aspekte des Dienstmanagements abdecken.

Gefragt ist also an dieser Stelle eine möglichst vollständige und gleichzeitig generische Beschreibung von Dienstmanagementaufgaben, auf deren Grundlage weitere Informationsanforderungen abgeleitet werden können. Die grundlegende, in dieser Arbeit verfolgte Idee zur Erreichung dieser Zielsetzung stützt sich auf Spezifikationen von Managementprozessen: Mit Prozessen verknüpfte Aktivitäten werden als Quelle für Dienstmanagementaufgaben bzw. Anwendungsfälle betrachtet und dienen zur systematischen Ableitung von Dienstmanagementinformationen (siehe Abbildung 5.6).

Die für diesen Ansatz benötigte Sammlung von Managementprozessen kann wiederum aus den in Kapitel 3 vorgestellten Prozessrahmenwerken gewonnen werden: Sie stellen Referenzprozesse zur Verfügung, weisen

zudem ein hohes Maß an Technologieunabhängigkeit auf und adressieren ein großes Spektrum an Dienstmanagementaufgaben. Der Bewertung in Abschnitt 3.4 folgend, wird in dieser Arbeit – vor allem wegen der erwähnten, größeren Verbreitung – *ITIL* der Vorzug gegenüber *eTOM* gegeben. Weitere ausführliche Begründungen für eine Wahl zugunsten ITIL finden sich in [Bre07].

Dieses Vorgehen bietet mehrere Vorteile hinsichtlich der in Abschnitt 2 identifizierten Anforderungen. Einerseits kann mit Hilfe des ITIL-Prozessrahmenwerks allgemeingültige und vollständige Managementinformation auf nachvollziehbare Weise ermittelt werden (KMI 1, KMI 3 und KMI 5), die zudem einheitlich bezüglich der verwendeten Terminologie und des Ursprungs ist (KMI 4). Andererseits erfüllt dieser Ansatz inhärent die geforderte Einbettung in Managementprozesse (KMI 2), da eine Planung, Ausführung, Überwachung und Nachjustierung von Managementprozessen ermöglicht wird.

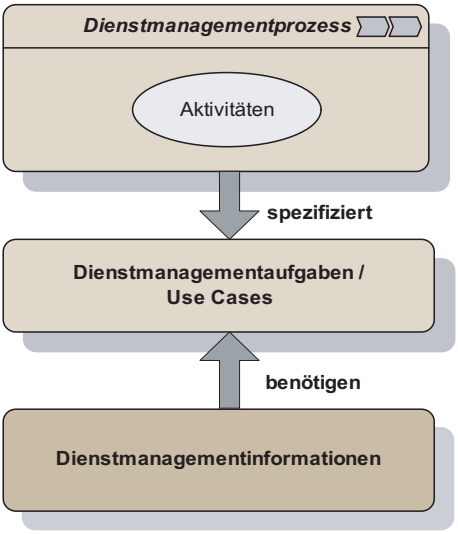

Abbildung 5.6: Ableitung von Dienstmanagementinformationen aus Prozessen

An dieser Stelle darf eine weitere Möglichkeit zur Ableitung von Dienstmanagementinformationen nicht unerwähnt bleiben. Im Rahmen der Habilitationsschrift von Dreo Rodosek [DR02] (siehe auch Abschnitt 3) wurden für einen Provider anfallende Dienstmanagementaufgaben anhand des Funktionsmodells der OSI-Managementarchitektur [ISO89a] strukturiert. Diese Klassifizierung unterscheidet zwischen Fehler-, Konfigurations-, Abrechnungs-, Leistungs- und Sicherheitsmanagement (englischsprachige Abkürzung: FCAPS) und erlaubt so eine Einordnung von Managementaufgaben. Obwohl diese funktionsspezifische Einordnung ursprünglich für das Netz- und Systemmangement konzipiert wurde, weisen die zugrundeliegenden Konzepte ein hohes Maß an Generizität auf, so dass eine Adaption auf das Dienstmanagement im Rahmen der genannten Arbeit erfolgreich durchgeführt werden konnte.

Da die durch ITIL vermittelten Managementaufgaben Ähnlichkeiten mit den FCAPS aufweisen [DSgF07], kann der in der vorliegende Arbeit gewählte Ansatz als Erweiterung von [DR02] angesehen werden. Dies umfasst insbesondere die durch ITIL einhergehende stärkere Verflechtung von IT-Management mit den Geschäftszielen einer Unternehmung.

Zur weiteren Konkretisierung des gewählten Ansatzes werden im Folgenden zunächst die für eine Ableitung von Managementinformationen relevanten Teile von Prozessbeschreibungen vorgestellt.

5.3.1 Relevante Teile von Prozessbeschreibungen

Teilaktivitäten des Prozesses ein

Um den Informationsbedarf von IT-Prozessen zu bestimmen bzw. Informationsanforderungen an eine Service-MIB zu charakterisieren, ist es zunächst erforderlich, den allgemeinen Aufbau eines Prozesses zu betrachten. Grundsätzlich kann ein Prozess als eine logisch zusammenhängende Gruppierung von Aktivitäten zur Erreichung eines vorab definierten Ziels angesehen werden [BKP02]. Er legt also fest, welche Tätigkeiten ausgeführt werden sollen, welche Ergebnisse zu erwarten sind, wie festgestellt wird, ob das anvisierte Ergebnis erreicht wird oder wie die Ergebnisse einen anderen Prozess beeinflussen. Unter Berücksichtigung der genannten Aspekte lassen sich drei Teilaktivitäten unterscheiden

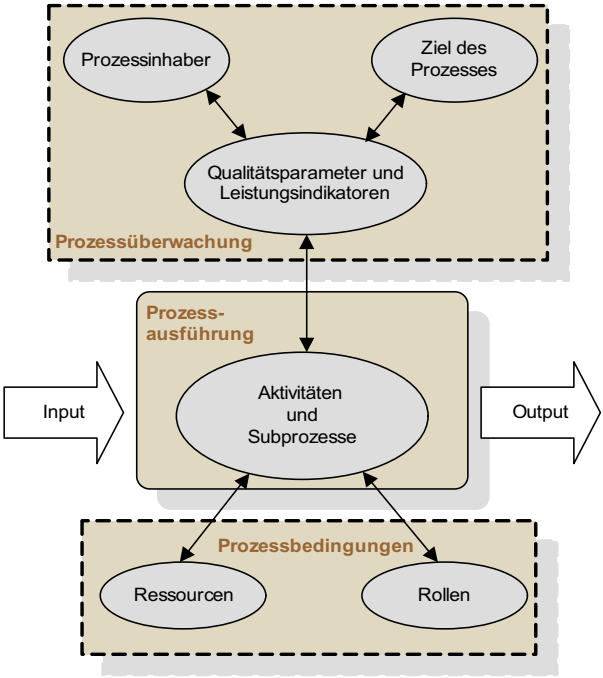

Abbildung 5.7: Generisches ITIL-Prozeßmodell nach [Off01]

(siehe Abbildung 5.7), die bei der Ermittlung von Informationsanforderungen Berücksichtigung finden müssen:

- *Prozessausführung*
  Die Prozessausführung umfasst Aktivitäten und Subprozesse, die einen bestimmten Input in einen festgelegten Output überführen. Am Beispiel des in Abschnitt 2.2.1 vorgestellten Web-Hosting Szenarios könnte eine Aktivität darin bestehen, dass basierend auf einer Fehlermeldung hinsichtlich des Ausfalls einer Switch-Komponente die betroffenen SLAs ermittelt und eine Priorisierung des Fehlers vorgenommen wird. Entscheidend für den pro-

zessorientierten Ansatz ist hierbei, dass Ein- und Ausgabe genau spezifiert werden, um somit die erzielten Ergebnisse bewerten zu können.
Zur Ausführung des Prozesses benötigte, dienstbezogene Informationen sollen über die Service-MIB zur Verfügung gestellt werden. In dem obigen Beispiel würde dies eine logische Beschreibung der Infrastruktur darstellen, anhand derer sowohl der von der Switch-Komponente abhängige Dienst bzw. der damit verknüpfte SLA ermittelt werden können. Denkbar wäre allerdings auch der umgekehrte Fall: Der Output eines Prozesses führt zu einer Aktualisierung der Service-MIB und wird so anderen, darauf zugreifenden Prozessen zur Verfügung gestellt. Damit bildet die Prozessausführung den wichtigsten Bestandteil einer Prozessbeschreibung hinsichtlich der Ableitung von Informationsanforderungen.
Besonders beachtet werden muss in diesem Zusammenhang, dass Prozessartefakte (z.B. Incident Records) zwar zur Prozessausführung notwendige Bestandteile darstellen, aber nicht in einer Service-MIB Berücksichtigung finden. Vielmehr wird, der Argumentation in Abschnitt 3.1.4 folgend, die Bereitstellung dieser Artefakte als Aufgabe einer CMDB angesehen.

- *Prozessüberwachung*
  Den Prozess im Hinblick auf die erwünschten Qualitätseigenschaften zu steuern stellt die Aufgabe der Prozessüberwachung dar. Dafür ist es zunächst erforderlich, geeignete *Qualitätsparameter* und *Leistungsindikatoren (Key Performance Indicators, KPIs)* festzulegen, die eine Beurteilung der Prozessausführung und somit eine weitergehende Optimierung ermöglichen. Der *Prozessinhaber (process owner)* trägt dabei die Verantwortung für die Ergebnisse des Prozesses.
  Die Aufgabe der Service-MIB in der Prozessüberwachung besteht darin, umfassende Basisdaten bereitzustellen, auf denen basierend, eine weiterführende Berechnung der Leistungsindikatoren vorgenommen werden kann. Konkret bedeutet das in obigem Beispiel, dass Informationen in der Service-MIB zur Gesamtanzahl an Dienstfehlern während eines bestimmten Zeitraums zur Bestimmung des KPIs Lösungsquote herangezogen werden können.

- *Prozessbedingungen*
  Unter Prozessbedingungen versteht man zur Ausführung des Prozesses notwendige Ressourcen und Rollen. Ressourcen können beispielsweise in Form von Personal, Finanzmittel oder Infrastruktur vorliegen, typische Rollen sind *Prozessverantwortlicher (process manager)* oder *Prozessausführender*.
  Da die Service-MIB eine Informationsbasis für Managementinformationen darstellt, kann sie als Ressource bzw. Prozessbedingung verstanden werden.

Mit den Bestandteilen einer Prozessbeschreibung wurden gleichzeitig die wesentlichen Ansatzpunkte für eine Ableitung von Informationsanforderungen vorgestellt: Insbesondere die mit der Prozessausführung und -überwachung verknüpften Dienstmanagementaufgaben eignen sich in diesem Zusammenhang für eine weiterführende Analyse. Unter dieser Prämisse werden im Folgenden ITIL Referenzprozesse untersucht und somit der zweite Teilschritt der Analysephase durchgeführt. Zunächst erfolgt allerdings eine Auswahl von ITIL-Prozessen für dieses Vorhaben.

Auswahl von ITIL-Prozessen zur exemplarischen Anwendung

Berücksichtigt man den Umfang des ITIL-Rahmenwerks (siehe Abschnitt 3.1.4), wird offensichtlich, dass eine vollständige Analyse aller Prozessbeschreibungen den Rahmen dieser Arbeit sprengen würde. Aus diesem Grund wird eine Auswahl bestimmter Prozesse notwendig, anhand derer die entwickelte Ableitungsmethodik exemplarisch Anwendung findet. Mit dem Ziel möglichst elementare Prozesse auszuwählen, fiel die Entscheidung dabei auf jeweils einen Vertreter aus dem *Service Support* und *Service Delivery*.

Mit dem Incident-Management wurde ein Prozess ausgewählt, der sehr häufig am Anfang einer ITIL-Umsetzung steht [Gar04a] und zugleich ein hohes Automatisierungspotential und somit eine Eignung zur Werkzeugunterstützung bietet [Bre06]. Zudem stellen Störungen in der Regel äußerst geschäftskritische Vorgänge für den Provider dar, da eine Verletzung in dem SLA vereinbarter Qualitätsparameter oftmals mit Strafzahlungen belegt ist und überdies zu Kundenunzufriedenheit führt.

Die Auswahl des Service-Level Managements innerhalb der exemplarischen Analyse wird durch seine vitale Rolle in der Umsetzung einer qualitätsorientierten Sichtweise auf den Dienst gerechtfertigt. Er bildet die Grundlage für Kunden- und Dienstorientierungsbestrebungen und trägt überdies entscheidend zu einem *IT-Business Alignment* bei.

Insgesamt zeigt sich bei der Betrachtung von ITIL Rahmenwerks, dass, mit Ausnahme des Service-Level Managements, die Service Support Prozesse ein größeres Potential zur Ableitung von Informationsanforderungen bieten. Dies betrifft insbesondere den *Configuration Management* Prozess, der eine wichtige Rolle in der Dienstbereitstellung einnimmt und deshalb nochmal im Rahmen der Beschreibung weiterführender Forschungsfragestellungen (Kapitel 8) aufgegriffen wird.

5.3.2 Incident-Management

Der Incident-Management Prozess trägt dazu bei, aufgetretene Störungen (*Incidents*) möglichst schnell zu beheben. Es soll somit die Funktionalität eines Dienstes umgehend wiederhergestellt und eine negative Auswirkung auf Geschäftsprozesse minimiert werden. Als Störung wird in diesem Zusammenhang ein Ereignis verstanden, welches nicht zum standardmäßigen Betrieb eines Dienstes gehört und tatsächlich oder potentiell eine Unterbrechung oder eine Minderung der Service-Qualität verursacht [IT 02]. Dieser Prozess zielt damit auf eine Verbesserung der Verfügbarkeit eines Dienstes und, damit verbunden, auf eine Erhöhung der Anwenderproduktivität und Kundenzufriedenheit ab. Entsprechend steht nicht zwingend die Behebung der einer Störung zugrundeliegenden Ursache im Vordergrund, sondern erscheint in manchen Fällen das Finden einer Behelfslösung (*Work around*) als ausreichend.

Neben der Behandlung von Störungen besteht eine weitere Aufgabe des Incident-Managements darin, *service requests* zu bearbeiten. Hierunter werden u.a. Anfragen eines Anwenders hinsichtlich der Handhabung oder Funktionalität eines Dienstes (RFA), der Passwortrücksetzung (RFI) oder Installation neuer Hard- oder Software (RfC) verstanden.

5.3.2.1 Teilprozesse des Incident-Managements

Der Incident-Management Prozess umfasst sechs Teilprozesse, die in Abbildung 5.8 grafisch dargestellt und im Folgenden kurz erläutert werden.

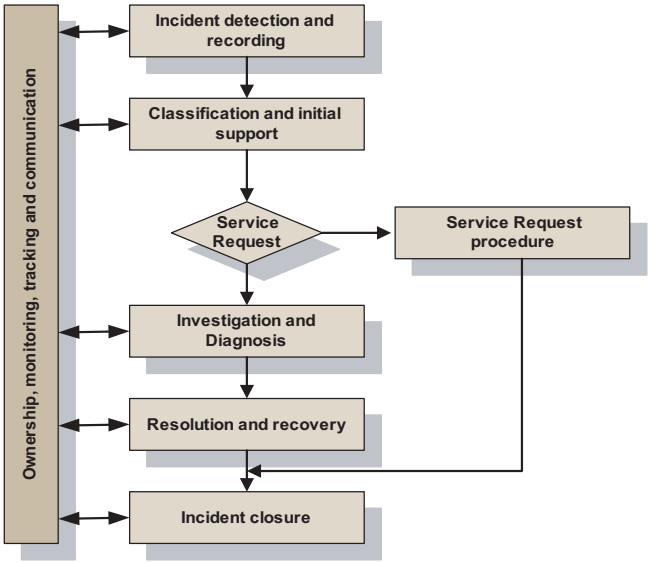

Abbildung 5.8: Der Incident-Management Prozess nach [Off00]

Incident Detection and Recording Eine Meldung über eine tatsächliche oder potentielle Störung bzw. das Auftreten eines *service requests* bilden den Auslöser für den ersten Subprozess *Incident Detection and Recording*. Werden Störungen von den Anwendern bemerkt und gemeldet, erfolgt in der Regel eine Annahme durch das Service Desk, welches als *Single-Point-Of-Contact* fungiert. Eine weitere Variante besteht darin, Störungen automatisiert mit Hilfe geeigneter Management- und

Überwachungssysteme zu erkennen und entsprechend an den Incident-Management Prozess weiterzuleiten.

In jedem Fall muss eine genaue Beschreibung der Störung erfasst und in Form eines Incident Records dokumentiert werden. Dies umfasst u.a. eine textuelle Beschreibung der Störung und der betroffenen Dienste oder Kontaktdaten des Meldenden bzw. Kunden.

Incident Record

Classification and Initial Support Um eine effiziente Weiterbearbeitung zu gewährleisten, wird in diesem Subprozess eine Klassifizierung und Priorisierung von Störungen vorgenommen. Während mit der Klassifizierung das Ziel verfolgt wird, ein Auffinden von bekannten Lösungen bzw. eine Zuordnung zu Spezialistengruppen zu erleichtern, dient die Priorisierung vorwiegend dazu, eine Bearbeitungsreihenfolge für Störungen festzulegen. Anhaltspunkte für eine Priorisierung stellen dabei zu erwartende Auswirkungen (*Impact*) und die Dringlichkeit (*Urgency*) einer Störung dar. Abhängig davon, ob es sich bei dem zu bearbeitenden Vorgang um eine Störung oder *service request* handelt, werden danach folgende Aktivitäten durchgeführt:

- *Service Request procedure*
  Für die Behandlung von *service requests* werden vorgesehene Prozeduren initiiert und von dem damit beauftragten Personal durchgeführt (z.B. Rücksetzen eines Passworts).
- *Incident Matching*
  Innerhalb dieser Aktivität wird überprüft, ob eine ähnliche Störung bereits aufgetreten ist und ob für diese eine Lösung bzw *Workaround* existiert. Dies dient dazu, Störungsmeldungen mit gleicher Symptomatik zu verknüpfen und als Einheit für die weitere Lösungsfindung zu betrachten. Weiterhin fließen in der Vergangenheit aufgetretene Störungen und deren Lösungen in das *Incident Matching* mit ein. Dieses Vorgehen spiegelt die vorrangige Zielsetzung des Incident-Managements wider, von einer Störung betroffene Dienste schnellstmöglich wiederherzustellen.

Wurde innerhalb des *Initial Supports* eine (temporäre) Lösung ermittelt, so wird diese dem Subprozess *Resolution and Recovery* zugeführt, andernfalls wird mit *Investigation and Diagnosis* weiter verfahren.

Investigation and Diagnosis Kann im Rahmen des *Initial Supports* keine Lösung bzw. *Workaround* ermittelt werden oder stimmt der Anwender dieser Lösung nicht zu, muss auf spezialisierte Supporteinheiten zurückgegriffen werden. Dieser, als funktionale Eskalation bezeichnete Vorgang, kann ebenfalls notwendig werden, falls vereinbarte Reaktions- oder Bearbeitungszeiten überschritten wurden.

Charakteristisch für den Subprozess *Investigation and Diagnosis* stellt sich dabei eine iterative Vorgehensweise dar, d.h. – falls notwendig – erfolgt eine mehrfache Eskalation zur nächstspezialisierten Supporteinheit, um letztendlich eine Lösung oder einen *Workaround* zu ermitteln. Dies beinhaltet auch eine Beantragung zusätzlicher Ressourcen oder Befugnisse (*hierarchische Eskalation*). In der Regel werden dabei bisher gesammelte, störungsbezogene Informationen einer tiefergehenden Analyse unterzogen und wiederum Wissendatenbanken (*Known-Error* und *Problem Records*) konsultiert.

Resolution and Recovery Die Umsetzung von in den vorhergenden Subprozessen gefundenen Lösungen oder *Workarounds* stellt die Aufgabe von *Resolution and Recovery* dar. In vielen Fällen resultieren daraus Änderungen der IT-Infrastruktur, die entsprechend mit den *Change-* und *Configuration-Management* Prozessen koordiniert werden müssen. Letzteres erfolgt nach den ITIL-Vorgaben im Rahmen eines *Requests for Change (RfCs)*, der eine Beschreibung der geplanten Änderungen enthält und zur Autorisierung an das *Change-Management* übergeben wird. Durch dieses Vorgehen soll insbesondere sichergestellt werden, dass durch die Lösungsbehebung vermittelte Änderungen entsprechend in Infrastrukturmodelle abgebildet und somit deren Konsistenz gewahrt wird.

Incident Closure Nachdem aus Sicht des Incident-Managements die Störung erfolgreich behoben werden konnte, wird der betroffene Anwender darüber in Kenntnis gesetzt. Stimmt er der vorgenommenen Lösung zu, kann die Störung abgeschlossen und archiviert werden (*Incident Closure*). In diesem Zusammenhang sollte insbesondere auf eine lückenlose Dokumentation geachtet werden, um ein *Incident Matching* bzw. eine Analyse zukünftiger Störfälle zu erleichtern.

Ownership, monitoring, tracking and communication Entlang der bisher genannten Subprozesse ist nach ITIL zusätzlich eine Aktivität zur übergreifenden Koordination der Störungsbehandlung vorgesehen. Ziel hierbei ist es, den Fortschritt der Störungsbehandlung zu überwachen, gegebenfalls durch geeignete Maßnahmen zu steuern und somit eine Einhaltung der im SLA vereinbarten Supportzeiten zu gewährleisten. Ein häufig gewähltes Mittel, um in den Verlauf der Strörungsbehandlung einzugreifen, stellt dabei die bereits genannte funktionale und hierarchische Eskalation dar.

Eine weitere wichtige Aufgabe von *Ownership, monitoring, tracking and communication* besteht in der Berechnung von Prozesskennzahlen, die zur Bewertung der Effizienz und Qualität des Incident-Management Prozesses Verwendung finden (vgl. Abschnitt 5.3.1). Sie bilden die Grundlage für Erstellung von *Reports* gegenüber anderen Prozessen (z.B. *Service-Level Management*) und Kunden des Dienstes. Für den Incident Management Prozess stellen die Gesamtzahl an Störungen, die durchschnittliche Lösungszeit, die Durchschnittskosten je Störung oder die Erstlösungsquote (Prozentsatz der direkt vom First-Level-Support gelösten Störungen) typische KPIs dar.

Es darf nicht unerwähnt bleiben, dass die genannten Aspekte nur die wesentlichen Bestandteile des Incident-Managements darstellen, die insbesondere für die im nächsten Abschnitt aufgezeigte Ableitung von Dienstmanagementinformationen relevant sind. Für eine umfassendere Ausführung wird an dieser Stelle auf [Off00, IT 02] oder [Bre07] verwiesen.

5.3.2.2 Ableitung von Dienstmanagementinformationen für das Incident-Management

Der entwickelten Ableitungsmethodik (siehe Abschnitt 5.3.1) folgend, werden nun Informationsanforderungen an eine Service-MIB zur Unterstützung des Incident-Management Prozesses abgeleitet. Anforderungen treten dabei sowohl in einem prozessübergreifenden Kontext auf, d.h. sie erstrecken sich auf mehrere Teilprozesse, können aber auch spezifisch für eine bestimmte Aktivität des Incident-Managements erscheinen.

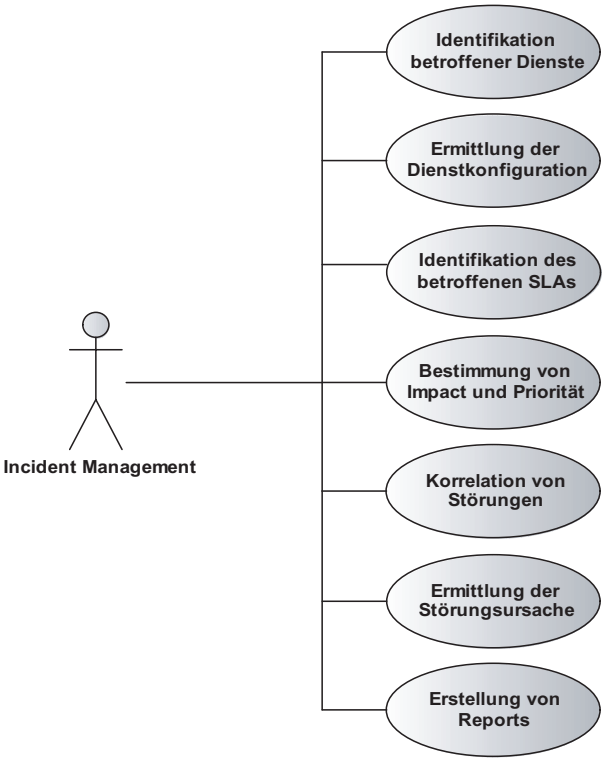

Abbildung 5.9: Betrachtete Anwendungsfälle des Incident-Managements

Aus den Prozessbeschreibungen extrahierte Anwendungsfälle werden in Abbildung 5.9 grafisch dargestellt. Sie repräsentieren im Rahmen des Incident-Managements auftretende Dienstmanagementaufgaben und legen gleichzeitig die Struktur der Ableitung fest: Schrittweise werden die verschiedenen Anwendungsfälle betrachtet und die zu ihrer Ausführung benötigte Dienstmanagementinformation bestimmt.

Weiterhin gilt es zu beachten, dass mit dem Incident-Management ver-

knüpfte Prozessartefakte (z.B. Incident Record) zwar in der Ableitung Berücksichtigung finden, aber nicht als Bestandteil einer Service-MIB verstanden werden (siehe Begründung in Abschnitt 3.1.4). Ebenfalls werden *service requests* nicht betrachtet, da damit assoziierte Anwendungsfälle als Teil des *Change Managements* angesehen werden. Entsprechend der oben genannten Unterscheidung werden nun zunächst prozessübergreifende Informationsanforderungen ausgeführt:

- **Identifikation betroffener Dienste**
  Eine Zuordnung zwischen Störungen und den davon betroffenen Diensten zu verwalten, stellt eine die gesamte Störungsbearbeitung übergreifende Aufgabe dar und wird entsprechend als notwendiger Bestandteil eines Incident Records angesehen. Die Grundlage hierfür bildet ein umfassender Dienstekatalog, der ein schnelles Auffinden und somit Referenzieren des betroffenen Dienstes erlaubt.
  Für die Service-MIB erwächst daraus die Notwendigkeit, Dienstbeschreibungen mit Attributen auszustatten, die eine eindeutige Identifikation ermöglichen (z.B. Dienst-ID) und es zugleich dem Supportpersonal gestatten, die Funktionalität des Dienstes (z.B. in Form einer textuellen Beschreibung) einfach nachzuvollziehen. Gleichzeitig sollte eine Einteilung in Dienstkategorien erfolgen und somit eine Strukturierung des entstehenden Dienstekatalogs vorgenommen werden.

- **Emittlung der Dienstkonfiguration**
  Service-MIB muss Konfigurationsbeschreibung reflektieren
  Um Störungen geeignet zu klassifizieren und im weiteren Verlauf des Incident-Managements die verursachenden Faktoren zu ermitteln, wird ein Zugriff auf die aktuelle Konfiguration des Dienstes benötigt und mit dem Incident Record verknüpft. Ebenso gilt es, im Anschluss an eine erfolgreiche Störungsbehebung, den daraus resultierenden Konfigurationszustand des Dienstes zu dokumentieren. Entscheidend ist hierbei eine zeitnahe Anpassung der Konfigurationsbeschreibung im Änderungsfall, da davon abhängige Managementprozesse eine möglichst aktuelle und konsistente Sicht auf die Dienstkonfiguration benötigen.
  Eine vorrangige Zielsetzung der Service-MIB besteht darin, eine aktuelle Managementsicht auf den Dienst zur Verfügung zu

stellen. Dies umfasst insbesondere eine Konfigurationsbeschreibung von statischen Aspekten des Dienstes (z.B. Name oder Ansprechpartner), funktionale und leistungsbezogene Merkmale (z.B. Qualitätseigenschaften), Zeitstempel der letzten Änderungen und Beziehungsinformationen (z.B. Abhängigkeiten zu Komponenten). Letztere repräsentieren in erster Linie die Beschreibung der Diensttopologie (siehe Abbildung 5.5) und stellen damit eine unverzichtbare Prämisse für effektives Incident-Management dar.

Nach der Vorstellung allgemeiner Informationsanforderungen efahren nun Anwendungsfälle Berücksichtigung, die spezifisch im Rahmen eines bestimmten Subprozesses des Incident-Managements auftreten:

- **Identifikation des betroffenen SLAs**
  Innerhalb des *Classification and Initial Support* Subprozesses besteht eine Aktivität darin, die von einer Störung betroffene Dienstvereinbarung (SLA) zu ermitteln und in den Incident Record mitaufzunehmen. Damit sollen mit dem Kunden vereinbarte Ausfall- und Wiederherstellungszeiten in die Störungsbearbeitung miteinbezogen, eine Berechnung der Dringlichkeit unterstützt (siehe nächsten Anwendungsfall) und eventuelle Eskalationszeiten festgelegt werden.
  Um Auswirkungen einer Störung auf vertragliche Vereinbarungen zu antizipieren, muss die Service-MIB eine Zuordnung zwischen SLA und Dienst ermöglichen [HSS05b]. Weiterhin Beachtung finden muss in diesem Zusammenhang, dass Störungen des Dienstes in den meisten Fällen als Konsequenz von Störungen in den dienstrealisierenden Netz- und Applikationskomponenten auftreten. Es ergibt sich also die Notwendigkeit, Relationen zu schaffen, die es ermöglichen, ausgehend von einem Komponentenfehler den betroffenen Dienst und den damit verknüpften SLA zu ermitteln [HSS04a].

Zuordnung zwischen SLA und Dienst

- **Bestimmung von Impact und Urgency**
  Treten mehrere Störungen gleichzeitig auf oder ist das mit der Störungsbehebung betraute Personal ausgelastet, stellt sich für

den Provider oftmals die Frage, welche Aktionen er zuerst ausführen soll. Zur Beantwortung dieser Fragestellung sieht der Subprozess *Classification and Initial Support* eine Prioritätsberechnung für Störungen vor, die sich an den zu erwartenden Auswirkungen (*Impact*) und der Dringlichkeit (*Urgency*) orientiert. Während zur Abschätzung des Impacts wiederum in der Dienstvereinbarung festgelegte Ausfall- und Reparaturzeiten bzw. die Höhe der Strafzahlungen im Falle einer Nichteinhaltung dieser Zeiten berücksichtigt werden müssen, erfordert die Dringlichkeitsberechnung eine Abschätzung des Nutzerverhaltens [HSS05b]. Einen weiteren Aspekt in der Impactanalyse bildet die Tragweite einer Störung: Beispielsweise kann der Impact als relativ hoch eingestuft werden, falls durch einen Dienstausfall eine Reihe weiterer Subdienste betroffen sind.
Zusätzlich zu den im letzten Anwendungsfall identifizierten Anforderungen hinsichlich der Zuordnung von Diensten und SLAs, wird es weiterhin erforderlich, Strafzahlungen als Teil des SLAs abzubilden und somit eine *Impact-Analyse* zu unterstützen. Ebenfalls müssen in diesem Zusammenhang das Dienst- und Abhängigkeitsmodell mit Indikatoren angereichert werden, die Auswirkungen von Störungen quantifizieren (z.B. welche Auswirkungen der Ausfall einer Komponente auf den Zustand des Dienstes hat) und somit zur Berechnung des Impacts herangezogen werden können bzw. eine computergestützte *Impact-Analyse* [Sch07] unterstützen.

- **Korrelation von Störungen**
  Gleichartige Störungen miteinander zu verknüpfen, stellt die primäre Aufgabe des Incident Matchings dar. Dadurch kann die – oftmals sehr hohe – Anzahl von Störungsmeldung dahingehend reduziert werden, dass nur noch die aus der Korrelation resultierende Störungmeldungen betrachtet werden muss. Wurde schließlich eine Lösung für die korrelierte Meldung gefunden, können meistens auch die damit verknüpften Einzelmeldungen als gelöst angesehen werden. Zur Automatisierung der genannten Vorgänge bietet sich insbesondere eine Toolunterstützung in Form von *Event-Korrelationswerkzeugen* an [Han07].
  Die wohl entscheidendste Voraussetzung für eine effiziente Stö-

rungskorrelation besteht in einer strukturierten Abhängigkeitsbeschreibung. Strukturiert drückt an dieser Stelle aus, dass verschiedene Arten von Abhängigkeiten erfasst und unterscheidbar gemacht werden. Dies umfasst neben Abhängigkeitsbeziehungen zwischen Diensten auch Abhängigkeiten, die auf Komponentenebene auftreten, sowie Beziehungen zwischen Diensten und dienstrealisierenden Komponenten (siehe auch Anforderung DEP 1). Aus Sicht der Störungskorrelation ist dabei eine möglichst feingranulare Beschreibung der genannten Abhängigkeitstypen wesentlich, d.h. im besten Fall eine separate Abhängigkeitsbeschreibung für jede Funktionalität des Dienstes. Nicht zuletzt soll es somit möglich werden, Korrelationsregeln automatisch aus dem Dienstmodell zu generieren [Han07]. Wegen des damit verbundenen hohen Modellierungs- und Pflegeaufwands gilt es allerdings, in der Service-MIB einen geeigneten Mittelweg zu finden und beispielsweise hohe Granularitätsstufen nur für die gängigsten Dienstfunktionalitäten vorzusehen.

- **Ermittlung der Störungsursache**
  Die Ursachen einer Störung zu ermitteln, bildet die Grundlage für eine schnelle Wiederherstellung betroffener Dienste dar. Dieses sollte unter der Zielsetzung geschehen, möglichst schnell einen *Workaround* zu ermitteln – kann die unterliegende Ursache nicht identifiziert werden, wird ein *Problem Record* erstellt und an den *Problem Management* Prozess weitergeleitet. ITIL sieht zur Bewältigung dieser Aufgabe vor, bestehende Wissensbasen (Incident-, Problem und Known-Error-Daten) in die Analyse miteinzubeziehen und somit einen Wissensaustausch zu gewährleisten. Als charakteristisch für Dienststörungen gilt dabei die kausale Abhängigkeit zu Störungen in Netz- und Applikationskomponenten: Erkennen von Dienststörungen bedeutet primär, Fehlerzustände dienstrealisierender Komponenten hinsichtlich ihrer Auswirkungen auf den Dienst zu erfassen.

  Dienstattribute werden benötigt

  Die Service-MIB sollte eine unterstützende Rolle in der Ermittlung von Störungsursachen einnehmen und somit als weitere Wissensbasis fungieren. In diesem Zusammenhang zeigt sich vor allem ein Bedarf an Dienstattributen, die Auskunft zu dem gegenwärtigen Zustand des Dienstes geben: Zunächst ist es für das mit der

Ermittlung der Störungsursache betraute Personal wichtig, sich einen Überblick über auftretende Fehler in Dienstausführungen und Verbindungsversuchen zu verschaffen. Entsprechende Attribute sollten deshalb sowohl zur Beschreibung der aktuellen Anzahl als auch der relativen Häufigkeit derartiger Fehler vorhanden sein.
Um Störungsursachen in dienstrealisierenden Komponenten (*Dienstelemente*) auszumachen, sind ferner Attribute notwendig, die deren Zusammenspiel reflektieren. Neben Informationen zur durchschnittlichen Fehlerrate und Auslastung der Dienstelemente ist es hierbei vor allem von Interesse, welches Element gegenwärtig die höchste Auslastung bzw. Fehlerrate besitzt. Ebenfalls sollte der Status und die Fehlerrate in (Netz-)Verbindungen zwischen Dienstelementen berücksichtigt und in entsprechenden Attributen abgebildet werden.

- **Erstellung von Reports**
  Um eine Übersicht über die erreichte Effizienz und Qualität des Incident-Managements zu erlangen, werden über ein festgelegtes Zeitintervall hinweg gesammelte Daten zu Prozesskennzahlen (KPIs) aufbereitet und zu einem Report zusammengefasst. Dieser Bericht dient einerseits zur Information des Kunden hinsichtlich seines abonnierten Dienstes, liefert aber auch wichtige Eingangsdaten für interne Planungsaktivitäten.
  In die Erstellung eines Reports fließen oftmals Informationen hinsichtlich der Ausprägung und Anzahl in einem Intervall aufgetretener Störungen mit ein. Zur Unterstützung dieses Vorgangs sollte die Service-MIB geeignete statistische Daten zu Dienststörungen führen und somit Basisinformationen für die Berichtserstellung vermitteln. Dies umfasst insbesondere Attribute, die Auskunft zur Anzahl einem Dienst zugeordneter Incidents und deren Bearbeitung geben. Hierbei ist es vor allem von Interesse, wieviele Störungen in der vorgegebenen Zeit gelöst werden konnten, mehrfach bearbeitet wurden oder ohne Feststellung einer Störungsursache geschlossen wurden.

Wie aus den obigen Ausführungen hervorgeht, kommt der Service-MIB eine wichtige Rolle bei der Bearbeitung von Störfällen zu. Besonders

im Hinblick auf eine weitgehende Automatisierung und Werkzeugunterstützung der genannten Prozessschritte stellt sie die Grundlage für einen strukturierten Wissensaustausch zwischen verschiedenen Managementanwendungen dar. Welche zusätzlichen Informationsanforderungen dabei durch die Betrachtung des Service-Level Management Prozesses auftreten, wird im nächsten Abschnitt weiter ausgeführt.

5.3.3 Service-Level Management

Eine tragende Rolle in der Umsetzung von Dienst- und Kundenorientierungsaspekten kommt dem Service-Level Management Prozess zu. Er fungiert als Brücke zwischen Kunden und Provider und dokumentiert in dieser Funktion Dienste und dessen Qualitätsmerkmale in einer für den Kunden verständlichen Weise. Dabei entstehende *Service Level Agreements (SLAs)* stellen wichtige Instrumente zur Vereinbarung und Nachweisbarkeit der Güte eines Dienstes dar.

Obwohl der Service-Level Management Prozess in einer, im Vergleich zu dem im vorigen Abschnitt behandelten Incident-Management, in einer weniger strukturierten Form vorliegt [Bre06], lassen sich doch eine Reihe von Teilaufgaben identifizieren, die im nächsten Abschnitt besprochen werden.

5.3.3.1 Teilaufgaben des Service-Level Managements

Grundsätzlich können innerhalb des Service-Level Management Prozesses fünf Teilaktivitäten unterschieden werden. Die erarbeiteten Ergebnisse werden in Abbildung 5.10 auf der rechten Seite dargestellt und im Folgenden, zusammen mit den Aufgaben einer Aktivität, erläutert:

Identify Zunächst werden Erwartungen der Kunden hinsichtlich der Güte des Dienstes erfasst und in Form von *Service Level Requirements (SLR)* festgehalten. Letztere bilden die Basis für eine anschließende Definition des SLAs.

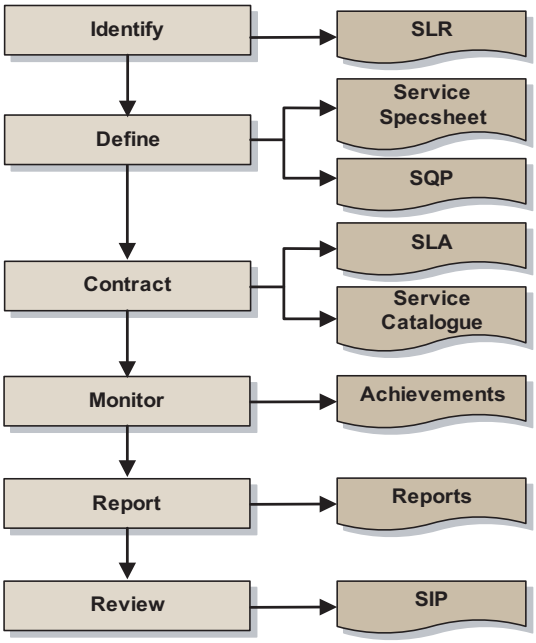

Abbildung 5.10: Der Service-Level Management Prozess nach [Off00]

Define Im Hinblick auf eine Implementierung des Dienstes im Einklang mit den identifizierten Qualitätsanforderungen werden nun SLRs in technische Spezifikationen *(Service Specification Sheet)* abgebildet und zusätzlich ein *Service Quality Plan (SQP)* erstellt. Letzterer soll Planungsschritte zur Umsetzung der vereinbarten Dienstqualität beinhalten und vorwiegend als provider-internes Dokument dienen.

Contract Basierend auf den zuvor erarbeiteten Dokumenten (SLR, Specsheet, SQP) wird nun ein rechtsgültiger Vertrag zwischen Kunde und Provider geschlossen. Dabei vorgenommene Änderungen bezüglich des Dienstes sollten in den Dienstekatalog des Provider (*Service*

Catalogue) einfliessen. Weiterhin sieht ITIL innerhalb dieser Aktivität Vereinbarungen mit internen (*Operational Agreements*) und externen (*Underpinning Contracts*) Dienstleistern vor. Damit soll sichergestellt werden, dass die in einem SLA enthaltenen Zielvorgaben auf alle an der Diensterbringung beteiligten Parteien projiziert werden.

Monitor Die Monitor-Aktivität setzt sich mit einer Erfassung der erreichten Qualitätsstufe des Dienstes (*service level*) auseinander. Dafür werden im SLA enthaltene Parameter gemäß der vereinbarten Modalitäten (z.B. Messmethode, Häufigkeit) gemessen und in *Service Level Achievements* konsolidiert.

Report Um Aussagen zum Erfüllungsgrad des SLA zu treffen, werden nun die erreichten *service levels* mit den Zielvorgaben des SLA verglichen, in einen Bericht (*Report*) überführt und dem Kunden vorgelegt.

Review Gegenstand der *review*-Aktivität bilden Auswertungen der zuvor erstellten Berichte in Zusammenarbeit mit dem Kunden und evtl. die Veranlassung von Änderungen bei Nichteinhaltung des SLAs. Dabei sollen insbesondere OLAs, UCs und das SQP Berücksichtigung finden. Um die Dienstqualität kontinuierlich zu verbessern schlägt ITIL weiterhin vor, ein *Service Improvement Program (SIP)* einzuführen, wobei auch Anregungen und Veränderungswünsche seitens des Kunden einfließen sollen.

5.3.3.2 Ableitung von Dienstmanagementinformationen für das Service-Level Management

Zur Ableitung von Informationsanforderungen aus dem Service-Level Management Prozess wurden wiederum eine Reihe von Anwendungsfällen betrachtet (siehe Abbildung 5.11). Sie repräsentieren Aufgabengebiete aus den im vorhergehenden Abschnitt betrachteten Teilaktivitäten, die insbesondere durch eine Service-MIB unterstützt werden sollen.

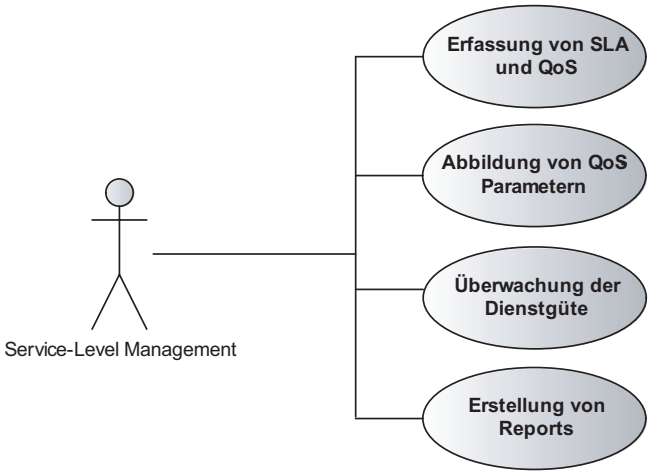

Abbildung 5.11: Betrachtete Anwendungsfälle des Service-Level–Managements

- **Erfassung von SLAs und QoS-Parametern**
  Zunächst ist es erforderlich, SLAs, OLAs und UCs formal zu erfassen und in der Service-MIB abzubilden. Dabei sollten in diesen Dokumenten enthaltene Zielvorgaben, zusammen mit aus einer Verletzung dieser Vorgaben resultierenden Pönalen, dargestellt werden.
  Weiterhin erfolgt die Festlegung von Qualitätseigenschaften eines Dienstes aus technischer Sicht mit Hilfe von QoS-Parametern, die innerhalb der Teilaktivität *Define* spezifiziert werden. Diese adäquat zu erfassen und mit den Zielvorgaben des SLAs zu assoziieren stellt eine weitere Informationsanforderung an die Service-MIB dar. Um einen späteren Vergleich zu gestatten, ist es dabei erforderlich, einen Zugriff sowohl auf aktuell gemessene als auch Soll-Werte für QoS-Parameter zu gewährleisten. Ferner muss es die Modellierungstruktur der Service-MIB erlauben, die Charakteristika von QoS-Parametern darzustellen: Dies umfasst insbesondere die Semantik, Messvorschriften, erlaubte

Wertemengen, Garantietyp und QoS-Kategorie eines Parameters [Roe05].
Während konkrete QoS-Parameter üblicherweise basierend auf die Anforderungen des Kunden spezifiziert werden, sollte die Service-MIB dennoch Basisattribute beinhalten, aus denen eine Berechnung derartiger Parameter möglich erscheint. Dies umfasst Attribute zur Beschreibung der Anzahl von Dienstausführungen und -anfragen während eines bestimmten Zeitraumes, das übertragene Datenvolumen zu und vom Dienst oder Dauer und Erfolgsrate von Verbindungsaufbauversuchen zum Dienst.

Basisattribute zu Leistungseigenschaften des Dienstes

- **Abbildung von QoS-Parametern**
  Dienstbezogene QoS-Parameter reflektieren in erster Linie vom Kunden an der Dienstschnittstelle feststellbare Qualitätseigenschaften des Dienstes (z.B. Sprachqualität bei VoIP). Sie können vom Provider oftmals nicht direkt beeinflusst werden, sondern resultieren vielmehr aus der Konfiguration der dienstrealisierenden Komponenten[1]. Um die gewünschte Dienstqualität zu erreichen, muss dementsprechend eine Abbildung dienstorientierter QoS-Parameter auf Leistungsparameter von Komponenten (z.B. Jitter) vorgenommen werden, was auch als *vertikales QoS-Abbildungsproblem* bezeichnet wird [HAN99].
  Aufgrund der Zielsetzung, eine zentrale Managementsicht auf Dienste zu repräsentieren, wird es für die Service-MIB erforderlich, neben dienstorientierten QoS-Parametern auch Dienste zu dokumentieren, die für deren Umsetzung zuständig sind (z.B. DiffServ [BBC+98]).
  Weiterhin sollten in diesem Zusammenhang Attribute in der Service-MIB mitaufgenommen werden, die Leistungseigenschaften von dienstrealisierenden Komponenten (Dienstelemente) repräsentieren und somit zur Abbildung von QoS-Parametern herangezogen werden können. Hierbei sind weniger Leistungsparameter einzelner Komponenten von Interesse, sondern vielmehr sind Attribute gefragt, die das Zusammenspiel zwischen dienstrealisierenden Komponenten beschreiben. Darunter fallen die kumulierte

Leistung der dienstrealisierenden Komponenten

[1] In [DR02] wird hierfür der Begriff Quality-of-Device (QoD) geprägt.

Verfügbarkeit und der durchschnittliche Durchsatz aller Dienstelemente oder die Geschwindigkeit bzw. das Volumen der Datenübertragung zwischen Dienstelementen.

- **Überwachung der Dienstgüte**
  Um feststellen zu können, ob ein Dienst in Einklang mit den im SLA enthaltenen Vorgaben operiert, muss ein Zugriff auf die aktuellen Qualitätseigenschaften des Dienstes ermöglicht werden. Dies stellt eine Aufgabe der Dienstüberwachung dar, die, anhand geeigneter Werkzeuge, kontinuierliche Messungen der QoS-Parameter durchführt, und bei Überschreiten festgelegter Schwellwerte entsprechende Alarmmechanismen auslöst. Eine besondere Schwierigkeit stellt hierbei wiederum die Abbildung von QoS-Parametern dar: Verminderte Qualitätseigenschaften des Dienstes werden in der Regel durch Leistungsengpässe in den dienstrealisierenden Komponenten verursacht (z.B. hohe CPU-Auslastung). Der Rückschluss von komponentenorientierten Leistungsparametern auf den Dienst muss somit von der Dienstüberwachung vollzogen werden können.
  Die Service-MIB sollte die Dienstüberwachung dahingehend unterstützen, dass verwaltete Abhängigkeitsbeziehungen auf einfache Weise zur Konfiguration von Monitoringwerkzeugen genutzt werden können. Durch die daraus resultierende, automatisierte Konfiguration kann erreicht werden, dass sich Monitoringwerkzeuge flexibel an Änderungen in der Dienstbereitstellung anpassen können.

- **Erstellung von Reports**
  Um in der *review*-Aktivität anfallende Aufgaben zu unterstützen, sollte die Service-MIB Attribute beinhalten, die als Vorlage für Berichte gegenüber Kunden dienen. Dies umfasst beispielsweise den Erfüllungsgrad im SLA vereinbarter Zielvorgaben, den Schweregrad von SLA-Verstößen oder daraus resultierende Strafzahlungen.

Nachdem die Ableitung von Informationsanforderungen anhand des Incident und Service-Level Management Prozesses demonstriert wurde, erfolgt nun eine Zusammenfassung der gewonnenen Ergebnisse.

5.3.4 Konkretisierung der Informationsanforderungen aus Managementprozessen

Mit der Betrachtung von Managementprozessen wurden eine Reihe neuer Anforderungen abgeleitet und somit der Informationsbedarf an dienstorientierter Managementinformation weiter konkretisiert. Um strukturierte Vorgaben für die Spezifikationsphase zu vermitteln, ist es nun erforderlich, die neu gewonnnenen Erkenntnisse mit den im ersten Teil der Analyse ermittelten Informationanforderungen zusammenführen. Dazu wird der in Abschnitt 5.2.4 vorgestellte Katalog hinsichtlich benötigter Attribute und Beziehungsinformationen verfeinert.

Zunächst wurde als weitere Informationsanforderung an die Entität Dienst die Einordnung in einen Dienstekatalog eingeführt (siehe Tabelle 5.3). Zusammen mit einer Beschreibung der Funktionalität des Dienstes soll so eine einfache Identifizierbarkeit des Dienstes gewährleistet werden (vgl. Abschnitt 5.3.2.2).

Wie aus der Ableitung von Informationsanforderungen aus dem Service-Level Management Prozesses ersichtlich wurde, gilt es weiterhin die Entiäten SLA und QoS zu berücksichtigen. Während hier insbesondere im Zusammenhang mit SLAs die Notwendigkeit zur Beschreibung der enthaltenen Zielvorgaben und Pönalen ermittelt wurde, sind es die mögliche Wertemengen und die Semantik von QoS-Parametern, die als Anforderung der Entität QoS gegenübergestellt werden. Ferner fallen in diesen Bereich die Beschreibung von QoS- und Koppeldiensten, wie bereits in Abschnitt 5.2.3 hervorgehoben.

Entität	Beschreibung
Dienst	Ein schnelles Auffinden des Dienstes sollte über eine Typisierung des Dienstes und Einordnung in einen Dienstekatalog gewährleistet werden. Darüberhinaus muss eine einfach nachvollziehbare Funktionalitätsbeschreibung erstellt werden, die elementare funktionale Bausteine des Dienstes aufzeigt.
QoS	Primär sollen Qualitätseigenschaften des Dienstes mit Hilfe dieser Entität erfasst werden. Insbesondere soll ein Vergleich zwischen aktuell gemessenen und im SLA vereinbarten Werten möglich sein und somit wiederum eine Aufteilung in Spezifikation und Implementierung gegeben sein. Ferner müssen QoS-Parameter beschrieben und anhand ihrer Semantik und Wertemenge sowie ihres Garantietyps charakterisiert werden. Im Zusammenhang mit der Entität QoS müssen ferner Koppeldienste und Dienste zur Durchsetzung von QoS-Eigenschaften beschrieben werden können. *Relationen:* Es muss ein Bezug zu SLA und Dienst hergestellt werden.
SLA	Die Entität SLA soll die zwischen Kunde und Provider vereinbarten Zielvorgaben hinsichtlich der Dienstbereitstellung repräsentieren und insbesondere Strafzahlungen im Falle einer Nichteinhaltung des SLAs dokumentieren. Die Verwendung von SLA-Templates soll unterstützt und eine Unterscheidung der verschiedenen Subtypen von SLAs (OLAs, UCs) unterschieden ermöglicht werden.

	Relationen: Der SLA muss mit dem betreffenden Dienst und dem Dienstkunden verknüpft werden. Ferner muss ein Bezug zwischen im SLA enthaltenen Zielvorgaben und QoS-Parametern hergestellt werden.

Tabelle 5.3: Konkretisierung der Informationsanforderungen in Entitäten

Informationsanforderungen für Dienstattribute

Neben entitätenbezogenen Anforderungen wurden weiterhin Informationsanforderungen ermittelt, die sich auf Attribute eines Dienstes beziehen. Während bei der Analyse des Incident Management Prozesses vor allem der Bedarf an Managementinformationen im Bereich Dienststörungen evident wurde, sind es hier vor allem leistungsbezogene Attribute des Dienstes, die zur Ausführung des Service-Level Management Prozesses benötigt werden. Dabei muss beachtet werden, dass derartige Informationsanforderungen jeweils eine Basismenge benötigter Attribute beschreiben, die zur weiteren szenariospezifischen Verfeinerung und als Grundlage zur Berechnung weiterer Größen gedacht sind. Damit sollte vermieden werden, dass eine ausufernde Anzahl von Attributen entsteht, wie dies z.B. bei Internet-MIBs der Fall war.

Drei Arten von Dienstattributen

Übergreifend zeigte sich dabei die Notwendigkeit zur Unterscheidung verschiedener Arten von Dienstattributen: Neben Attributen, die auf einer holistischen Betrachtung des Dienstes fußen, umfasst dies Attribute zur Beschreibung des Zusammenspiels zwischen dienstrealisierenden Komponenten und zur Berichterstellung (*Reporting*) gegenüber Dienstkunden.

Der genannten Aufteilung entsprechend werden zunächst in Tabelle 5.4 Informationsanforderungen dargestellt, die ihren Ursprung in der Analyse des Incident Management Prozesses haben. Maßgeblich zeigte sich dabei der Bedarf nach Dienstattributen bei der Ermittlung der Störungsursache (Abschnitt 5.3.2.2): Neben der Beschreibung von Fehlerhäufigkeiten während der Dienstausführung und des Verbindungsaufbaus zum Dienst wurde hier weiterhin auf eine Beschreibung der Auslastung und Fehleranfälligkeit in dienstrealisierenden Komponenten Wert

gelegt. Ferner sollten Basisgrößen, wie z.B. die Anzahl dem Dienst zugeordneter Incidents, in geeignete Attributsdefinition überführt werden.

Art	Informationsanforderung
Ganzheitliche Betrachtung des Dienstes	• Gegenwärtiger Status des Dienstes einschließlich Degradierungen der Dienstqualität • Anzahl, Durchschnittswert und relative Häufigkeit fehlerhafter Dienstausführungen • Fehler im Verbindungsaufbau zum Dienst
Fehler im Zusammenspiel dienstrealisierender Komponenten	• Ausfallrate von Dienstelementen • Durchschnittliche Fehlerrate der Dienstelemente • Durchschnittliche Auslastung der Dienstelemente • Dienstelement mit höchster Fehlerrate/Auslastung • Status, Fehlerrate und Auslastung der (Netz-)Verbindungen zwischen Dienstelementen
Prozessreporting	• Anzahl dem Dienst zugeordneter Incidents • Incidents, die in vorgebener Zeit gelöst wurden • Incidents, die mehrfach bearbeitet wurden • Geschlossene Incidents, bei denen kein Problem festgestellt wurde

Tabelle 5.4: Informationsanforderungen hinsichtlich von Dienstattributen

Leistungsbezogene Größen des Dienstes

Der nächste Block an Informationsanforderungen geht auf die Analyse des Service-Level Management Prozesses zurück. Hier wurde, insbesondere in der Teilaktivität *Monitoring and Reporting*, ein Bedarf

nach Dienstattributen zu leistungsbezogenen Größen des Dienstes festgestellt. Wie aus Tabelle 5.5 ersichtlich, findet hierbei wiederum die Aufteilung in drei Arten von Dienstattributen Anwendung.

Art	**Informationsanforderung**
Ganzheitliche Betrachtung des Dienstes	• Gegenwärtige und durchschnittliche Anzahl von Dienstausführungen • Bearbeitungszeit von Dienstausführungen • Anzahl an den Dienst gestellter Anfragen • Verfügbarkeit des Dienstes • Übetragenes Datenvolumen zu und von dem Dienst • Benötigte Zeit zum Verbindungsaufbau mit dem Dienst • Anzahl und Erfolgsrate von Verbindungsversuchen
Leistung der dienstrealisierenden Komponenten	• Verfügbarkeit der Dienstelemente • Aktueller und durchschnittlicher Durchsatz der Dienstelemente • Geschwindigkeit der Datenübertragung zwischen Dienstelementen • Zwischen Dienstelementen übertragenes Datenvolumen
Prozessreporting	• Anteil der im SLA enthaltenen Zielvorgaben die erfüllt wurden • Schweregrad der SLA-Verstöße • Resultierende Strafzahlungen aus SLA-Verstößen

Tabelle 5.5: Informationsanforderungen hinsichtlich von Dienstattributen (Fortsetzung)

Abschließend zur Analysephase werden nachfolgend die wichtigsten Ergebnisse zusammengefasst und einer Bewertung unterzogen.

5.4 Zusammenfassung

In diesem Kapitel wurde ein zweistufiges Vorgehen gewählt, um den Informationsbedarf an dienstorientierten Managementinformationen zu bestimmen. Zunächst stellten Dienstorientierungsaspekte den Mittelpunkt der Betrachtung dar: Anhand des MNM-Dienstmodells sowie unter Berücksichtigung von Dienstlebenszyklus und interorganisationalen Aspekten wurden Informationsanforderungen abgeleitet, anschließend konsolidiert und in einem Katalog abgebildet.

Der zweite Teil der Analysephase widmete sich Managementprozessen. Es wurde eine Ableitungsmethodik entwickelt, die es gestattet, Informationsanforderungen aus standardisierten Prozessbeschreibungen abzuleiten und auf diese Weise den im ersten Teil der Analyse erstellten Katalog weiter zu verfeinern. Letzterer repräsentiert somit grundlegende Informationsanforderungen an eine Service-MIB, zeigt den Bedarf an Entitäten, Attributen und Relationen auf und vermittelt so konkrete Vorgaben hinsichtlich der nachfolgenden Spezifikationsphase.

Wie aus der Zusammenfassung hervorgeht, stellen in diesem Kapitel entwickelte Lösungen einen ersten wichtigen Schritt hinsichtlich der Konzeption einer Service-MIB dar. Entsprechend erscheint es an dieser Stelle vernünftig, einen Zwischenabgleich mit den in Kapitel 3 identifizierten Anforderungen an einen Lösungsansatz durchzuführen.

Stellt man den Anforderungskatalog (Tabelle 2.1) den bisher erzielten Ergebnissen gegenüber, wird deutlich, dass durch Analysephase insbesondere die Beschaffenheit konkreter Managementinformation festgelegt wird. Zwar erfolgt eine tatsächliche Definition von Entitäten, Attributen und Relationen erst in der Spezifikationsphase, aber dieser Vorgang wird nachhaltig von den Ergebnissen des vorliegendes Kapitel beeinflusst. Unter diesem Gesichtspunkt werden deshalb im Folgenden Anforderungen aus dem Bereich *Definition konkreter Managementinformation* bewertet und in Tabelle 5.6 dargestellt:

- Zunächst zeichnet sich der gewählte Ansatz durch eine klare Ableitungsmethodik aus. So definierte Managementinformation bzw. deren Bestimmungszweck können somit einfach nachvollzogen werden (KMI 5). Weiterhin weist keine der gewählten Quellen

szenario- oder technologiespezifische Charakteristika auf, die Ableitung kann somit als allgemeingültig angesehen werden (KMI 1).

- Durch die Ausrichtung an den Vorgaben des ITIL-Prozessrahmenwerks wird inhärent die geforderte geschäftsorientierte Sichtweise (KMI 2) eingenommen. Ferner können so zwei wesentliche Vorteile von ITIL auf die Ableitung bzw. Definition von Managementinformationen transferiert werden: Einerseits beschreibt ITIL umfassend an IT-Provider gerichtete Managementaufgaben und trägt so zur Definition vollständiger Managementinformation bei (KMI 3). Andererseits erfolgt diese Beschreibung nach einem einheitlichen Schema – was auch auf die daraus abgeleitete Managementinformation zutrifft (KMI 4). Hierbei muss erwähnt werden, dass zwar innerhalb der exemplarischen Analyse des Incident und Service-Level Management Prozesses keine Vollständigkeit hinsichtlich aller Managementaufgaben erreicht wurde, die zugrundeliegende Methodik aber durchaus in der Lage ist, dieser Anforderung gerecht zu werden.

- Die beiden verbleibenden Anforderungen wurden insbesondere durch den ersten Teil der Analysephase adressiert: Mit der Berücksichtigung interorganisationaler Aspekte und der Hinzunahme einer Entität Organisation in den Informationskatalog konnte gleichzeitig Anforderung KMI 7 entsprochen werden. Weiterhin wurden anhand des MNM-Dienstmodells die Entitäten Dienstnutzer und Dienstkunde abgeleitet, mit dem Dienst in Beziehung gesetzt und somit die Grundlage für eine kundenspezifische Aufbereitung (KMI 6) geschaffen.

Gemäß der in Kapitel 4 vorgestellten Methodik wird nun mit der Spezifikationsphase weiter verfahren.

Allgemeine Anforderungen	
Anforderung	Bewertung
ALL 1 Dienstorientierung	✔
ALL 2 Berücksichtigung des Dienstlebenszyklus	✔

Definition konkreter Managementinformation	
Anforderung	Bewertung
KMI 1 Allgemeingültigkeit	✔
KMI 2 Ausrichtung an Geschäftszielen und -prozessen	✔
KMI 3 Vollständigkeit	(✔)
KMI 4 Einheitlichkeit	✔
KMI 5 Nachvollziehbarkeit	✔
KMI 6 Kundenspezifische Aufbereitung	✔
KMI 7 Berücksichtigung interorganisationaler Aspekte	✔

Beschreibung von Abhängigkeitsbeziehungen	
Anforderung	Bewertung
DEP 1 Differenziertheit	✔
DEP 2 Bezug zu SLAs und Kunden	✔
DEP 3 Berücksichtigung dynamischer Aspekte	✔
✔: erfüllt (✔): teilweise erfüllt **?**: keine Aussage möglich	

Tabelle 5.6: Erfüllung der Anforderungen durch die Analysephase

Kapitel 6

Spezifikation dienstbezogener Managementinformation

Nachdem im letzten Kapitel mit der Analysephase der erste Schritt zur Konzeption einer Service-MIB vollzogen wurde, schließt sich nun die Ausführung der Spezifikationsphase an. Das Ziel besteht hierbei darin, Konzepte zu schaffen, die eine Umsetzung der in Kapitel 4 dargelegten Methodik ermöglichen und somit, neben einer managementorientierten Beschreibung von Diensten, insbesondere die Abbildung von komponenten- auf dienstorientierte Managementinformationen adressieren. Zur Erreichung dieses Ziels werden eine Reihe von Erweiterungen für bestehende Ansätze vorgestellt bzw., wenn notwendig, neue Lösungen entwickelt.

Auf diese Weise sollen die vorab identifizierten Informationsanforderungen in geeignete Entitäten, Methoden und Beziehungen überführt und somit konkrete Managementinformation geschaffen werden. Das dabei entstehende Modell für Dienstmanagementinformationen kann als Kernstück der Service-MIB angesehen werden. Welches Vorgehen auf dem Weg dahin gewählt wird, stellt den Gegenstand des nächsten Abschnitts dar.

6.1 Vorgehen in der Spezifikationsphase

Spezifikationsphase verfolgt drei Ziele

Das Vorgehen in der Spezifikationsphase vereint drei miteinander verbundene Zielsetzungen: Es sollen dienstorientierte Managementinformationen definiert, Konzepte zur Abbildung von komponenten- auf dienstorientierter Managementinformation geschaffen und gleichzeitig der Übergang auf die nachfolgende Überwachungsphase berücksichtigt werden.

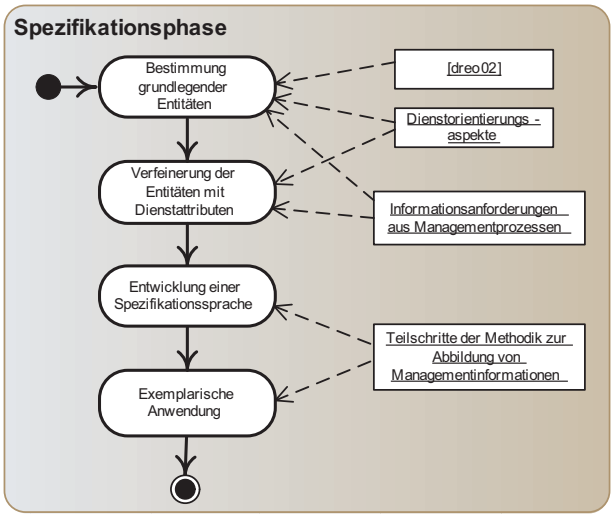

Abbildung 6.1: Vorgehen in der Spezifikationsphase

Stellt man dieses Vorhaben den in Abschnitt 2.1.2 dargelegten Grundlagen zur Repräsentation von *Managed Objects* gegenüber, wird deutlich, dass im Hinblick auf die anvisierte Spezifikation ein geeignetes Modell bzw. eine Datenmodellierungssprache zwingend erforderlich ist. Wie allerdings aus der Analyse verwandter Arbeiten hervorgeht (siehe Abschnitt 3.4), eignet sich keiner der darin vorgestellten Ansätze zur uneingeschränkten Anwendung in der Spezifikationsphase. Hierbei ist es vor allem die fehlende Berücksichtigung von Beziehungen zwischen

Diensten und Komponenten bzw. der fehlende Detailgrad hinsichtlich generischer Dienstattribute, welche eine unmittelbare Umsetzung der dieser Arbeit zugrundeliegenden Methodik erschwert.

Basismodell der Service-MIB

Prinzipiell gliedert sich das in diesem Kapitel gewählte Vorgehen (Abbildung 6.1) in zwei Blöcke. Zunächst steht dabei die Erstellung eines Basismodells der Service-MIB bzw. dessen Verfeinerung mit Dienstattributen im Vordergrund, was sich in den folgenden Teilschritten ausdrückt:

- *Bestimmung grundlegender Entitäten*
  Unter Einbeziehung in der Informationsanalyse ermittelter Dienstorientierungsaspekte werden zunächst für das Dienstmanagement grundlegende Managementobjekte definiert. Das dadurch entstehende Basismodell der Service-MIB orientiert sich dabei an verwandten Arbeiten: Insbesondere das in [DR02] entwickelte *Service Template Model* fließt in die Definition mit ein und wird zu diesem Zwecke an mehreren Stellen erweitert.

- *Erweiterung des Basismodells mit Dienstattributen*
  Den nächsten Schritt bei der Definition konkreter Managementinformation bildet die Verfeinerung des vorab erstellten Grundgerüsts mit Dienstattributen. Hierfür wird primär der in der Informationsanalyse identifizierte Informationsbedarf von Managementprozessen in geeignete Definitionen überführt und im Zuge dessen das Basismodell der Service-MIB erweitert. Die entstehenden Attribute stellen dabei gewissermaßen Templates dar, die durch Anreicherung mit Abbildungsvorschriften an ein bestimmtes Szenario angepasst werden können. Im Hinblick auf die vorgestellte Synthesemethodik wird dadurch der Teilschritt 1 (*Deklaration statischer Eigenschaften*) komplementiert und die Grundlage für eine Festlegung von Abbildungsvorschriften geschaffen.

Festlegung von Abbildungsvorschriften

Der zweite Block des Vorgehensmodells der Spezifikationsphase widmet sich der Abbildung von komponentenorientierter auf dienstorientierte Managementinformationen. Dies umfasst die Entwicklung einer Spezifikationsphase zur Festlegung von Abbildungsvorschriften und deren exemplarische Anwendung:

- *Entwicklung einer Spezifikationssprache*
  Mit der entwickelten Spezifikationssprache wird der formale Rahmen zur Definition konkreter Dienstmanagementinformationen geschaffen und insbesondere eine Spezifikation feingranularer Abbildungsvorschriften zwischen Komponenten- und Dienstmanagementinformation ermöglicht. Letztere stellt eine unverzichtbare Voraussetzung zur Durchführung der in Kapitel 4 besprochenen Methodik dar. Gleichzeitig fließen Konstrukte in den Entwurf der *Service Information Specification Language* (SISL) mit ein, die eine Steuerung der nachfolgenden Überwachungsphase (Kapitel 7) unterstützen.

Anwendung der erarbeiteten Lösungen wird demonstriert

- *Exemplarische Anwendung*
  Wie bereits in Kapitel 4 dargelegt wurde, sind Abbildungsvorschriften zu einem gewissen Teil immer szenariospezifisch und können deshalb i.d.R. nicht allgemein festgelegt werden. Um den Adaptionsprozess und damit die Durchführung der fünf Teilschritte der Spezifikationsphase demonstrieren wird deshalb ein einfaches, aus dem GRID-Szenario entlehntes Beispiel, zur exemplarischen Anwendung gewählt.

In der Vorgehensbeschreibung wird deutlich, dass in der Spezifikationsphase auf eine möglichst einfache Integration in vorhandene Managementlösungen Wert gelegt wird. Konkrete Managementinformation wird auf konzeptioneller Ebene spezifiziert und kann somit prinzipiell in verschiedene bestehende Datenmodelle überführt werden. Soll allerdings eine Abbildung von komponentenorientierter Information geschaffen werden, wird eine Anwendung der in Abschnitt 6.4 vorgestellten Spezifikationssprache notwendig.

6.2 Definition grundlegender Entitäten

Im Hinblick auf eine Definition konkreter Managementinformation stellt sich zunächst die Frage, welche Entitäten überhaupt innerhalb einer Service-MIB berücksichtigt werden müssen bzw. für das Dienstmanagement relevant sind. Eine Antwort darauf vermittelt vor allem die im

letzten Kapitel durchgeführte Informationsanalyse: Mit der Betrachtung von Grundkonzepten im Dienstmanagementumfeld und vor allem dem MNM-Dienstmodell wurden eine Reihe von Entitäten identifiziert, die aus Sicht des Dienstmanagements unverzichtbar erscheinen.

Bestehende Arbeiten werden erweitert

Desweiteren wurden in Kapitel 3 bereits bestehende Arbeiten vorgestellt, die Anhaltspunkte für die Definition konkreter Managementinformation geben: Insbesondere die in [DR02] vorgestellten *Service Template Models* beschreiben bereits grundlegende Entitäten und enstanden überdies unter Berücksichtigung des MNM-Dienstmodells. Es erscheint deswegen eine Erweiterung dieses Modells gemäß den identifizierten Informationsanforderungen sinnvoll. Nicht zuletzt kann somit eine Einbettung in verwandte Forschungsarbeiten des MNM-Teams [Han07, Sch07] erzielt werden.

Zur Modellierung von Entitäten und deren Querbezügen wird ein objektorientierter Ansatz gewählt. Dies motiviert sich aus den dadurch gegebenen Vorteilen hinsichtlich der Darstellung von Vererbungsbeziehungen, Relationen usw., die bereits in Kapitel 3.4 genannt wurden. Zudem verwenden insbesondere neuere Managementmodelle objektorientierte Modellierungstechniken, so dass eine Integration mit diesen Arbeiten vereinfacht wird. In der grafischen Darstellung der entstehenden Modelle wird dabei auf die *Unified Modelling Language (UML)* in der Version 2 [OMG04] zurückgegriffen, die sich als führender Standard in diesem Bereich etabliert hat.

Designpattern finden Verwendung

Weiterhin finden in der Definition dienstorientierter Managementinformationen aus dem Software-Engineering stammende Design-Patterns Verwendung. Sie repräsentieren eine Sammlung über Jahre hinweg bewährter Entwurfsmuster und zielen darauf ab, möglichst flexible und erweiterbare Lösungen zu schaffen [GHJV95, Fow96]. Obwohl Design-Patterns in bestehenden Managementmodellen nur wenig Berücksichtigung finden, eignen sie sich doch hervorragend für die Erstellung einer Service-MIB – nicht zuletzt, weil damit exakt die Anforderungen MOD 2, MOD 4 und MOD 5 adressiert werden. Bevor nun die einzelnen Entitäten sukzessive definiert werden, erfolgt zunächst die Vorstellung des verwendeten Beschreibungsschemas.

6.2.1 Beschreibungsschema für Entitäten

In dem gewählten Beschreibungsschema spiegeln sich zwei grundlegende Zielsetzungen wider. Einerseits soll ein Bezug zu den in Kapitel 3 vorgestellten Managementmodellen hergestellt und dadurch eine Integration erleichtert werden. Andererseits soll die Entitätsbeschreibung nachvollziehbar (Anforderung MOD 4) bzgl. des Prozesskontexts sein und beispielsweise Rückschlüsse dahingehend zulassen, für welchen Prozess die jeweilige Entität relevant ist. Daneben werden noch weitere, auch in bestehenden Arbeiten zu findende Beschreibungsmöglichkeiten berücksichtigt, wie im Folgenden dargelegt wird:

	Entität Beispielentität
Beschreibung	In dieser Form werden Entitäten beschrieben
Synonym	Name einer verwandten Klasse in SID oder CIM
Prozessbezug	ITIL-Prozesse, die auf diese Entität unmittelbar zugreifen
Verwandte Entitäten	Bezug zu anderen Entitäten innerhalb der Service MIB

Tabelle 6.1: Entität Beispielentität

Name Mit diesem Feld wird die Entität mit einer eindeutigen Bezeichnung versehen, um sie so innerhalb der Service-MIB identifizierbar zu machen.

Beschreibung Mit der Beschreibung soll es Anwendern der Service-MIB ermöglicht werden, den Zweck der vorliegenden Entität nachzuvollziehen.

Synonym Das Synonym-Feld dient einer einfachen Integration mit bestehenden, standardisierten Informationsmodellen. Es beinhaltet die Bezeichnungen äquivalenter Entitäten in den Managementmodellen SID

und CIM, sofern diese vorhanden sind. Die Auswahl dieser beiden Modelle motiviert sich vor allem durch deren Bezug zu Dienstorientierungskonzepten (siehe Abschnitt 3.4).

Prozessbezug Aus den genannten Nachvollziehbarkeitsgründen werden in diesem Feld ITIL-Prozesse dargestellt, die unmittelbaren Bezug zu der vorliegenden Entität haben.

Verwandte Entitäten Um die Zusammenhänge zwischen den verschiedenen Bestandteilen der Service-MIB nachvollziehbar erscheinen zu lassen, werden in diesem Feld verwandte Entitäten genannt. Damit soll vor allem den Nachteilen begegnet werden, die im Rahmen der Bewertung von CIM in Abschnitt 3.1.1 genannt wurden.

Eine Anwendung des Beschreibungsschemas und damit eine Definition grundlegender Entitäten der Service-MIB wird im Folgenden, beginnend mit der Root-Entität, vorgestellt.

6.2.2 Die Wurzelentität ManagedServiceElement

ManagedService-Element fungiert als Superklasse

Als erstes Element des entstehenden Modells der Service-MIB wird zunächst eine Wurzelentität geschaffen, die als gemeinsamer Vorfahre bzw. Superklasse aller weiteren Entitäten auftritt. Dadurch wird eine Grundregel objektorientierten Designs befolgt, die Kohärenz zwischen den verschiedenen Modellteilen sichergestellt und insbesondere ein einfaches Traversieren des Modells von der Wurzel ausgehend ermöglicht.

Die Root-Entität *ManagedServiceElement* dient somit dazu, Eigenschaften zu repräsentieren, die über alle Entitäten der Service-MIB hinweg gleich sind (Abbildung 6.2). Dies umfasst, neben dem Namen der Entität, eine textuelle Beschreibung des Verwendungszwecks sowie die Versionsnummer, in der sich die Entität befindet. Letztere dient vor allem dazu, Änderungen in der Infrastruktur nachvollziehbar zu dokumentieren (vgl. ITIL Change Management in Abschnitt 3.1.4).

Entität ManagedServiceElement	
Beschreibung	Diese Entität repräsentiert die Wurzelentität der Service-MIB. Sie ist ein gemeinsamer Vorfahre aller Entitäten und beschreibt Eigenschaften, die über alle Entitäten hinweg gleich sind.
Synonym	Einem ähnlichen Zweck dienen die Klassen *ManagedElement* in CIM und *RootEntity* in SID.
Prozessbezug	Kein unmittelbarer Bezug vorhanden
Verwandte Entitäten	Fungiert als Superklasse für alle Entitäten der Service-MIB

Tabelle 6.2: Entität ManagedServiceElement

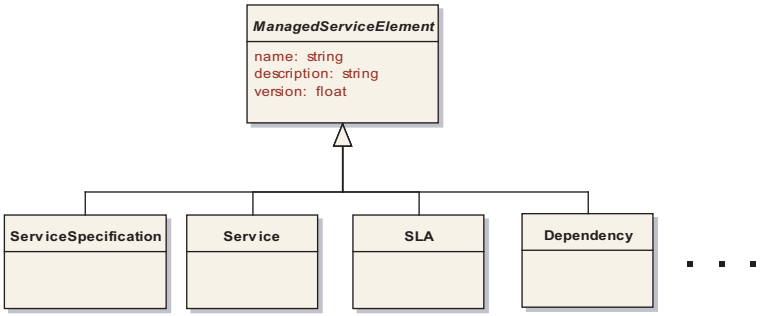

Abbildung 6.2: Klassenmodell der WurzelEntität

6.2.3 Dienst und Dienstspezifikation

Mit der Definition dienstbezogener Entitäten erfolgt nun die Festlegung der zentralen Bestandteile einer Service-MIB, die zugleich den Ausgangspunkt für die weitere Modellierung bilden. Entsprechend der in Abschnitt 5.2.4 dargestellten Informationsanforderungen besteht dabei ein wichtiges Kriterium darin, die Spezifikation und Implementierung des Dienstes strikt zu trennen. Auf das Service-MIB Modell abgebildet,

resultieren aus dieser Anforderung zwei grundlegenden Entitäten, die im Folgenden vorgestellt werden.

Dienstspezifikation schafft Basis für Implementierung

Die Aufgabe, gewünschte Eigenschaften und Funktionalitäten des Dienstes zu beschreiben, fällt der *Dienstspezifikation* zu. Sie fungiert damit als Vorlage (*Template*), auf deren Basis die Implementierung konkreter Dienste erfolgen kann und stellt gemeinsame Eigenschaften einer Klasse von Diensten dar. Gemäß den in [DR02] vorgestellten *Template Models* müssen dabei weiterhin ein *Service-*, *Provider-* und *Customer-Centric Part* berücksichtigt werden (vgl. Abschnitt 3.2.1). Entsprechend dieser Aufteilung entsteht die folgende Hierarchie von Entitäten, die gleichzeitig die schrittweise Verfeinerung einer Dienstspezifikation repräsentieren:

- **BasicServiceTemplate**
  Das *BasicServiceTemplate* repräsentiert die generische Spezifikation einer Klasse von Diensten, zusammen mit typischen Diensteigenschaften. Beispielsweise könnte ein derartiges Template für einen Web-Hosting Dienst beschreiben, dass dieser Dienst aus Web-, Applikations- und Datenbankserver besteht und dass die Anzahl von möglichen Transaktionen pro Sekunde einen wichtigen Leistungsparameter darstellt.

- **ProviderCentricTemplate**
  Spezifika eines Providers werden berücksichtigt

  Aus einer Verfeinerung des *BasicServiceTemplates* im Hinblick auf Provider-spezifische Randbedingungen geht das *ProviderCentricTemplate* hervor. Es stellt das Ergebnis der Planungsphase dar und konkretisiert die vom *BasicServiceTemplate* vererbten Diensteigenschaften dahingehend, dass für den jeweiligen Provider typische Ausprägungen bzw. Wertebereiche festgelegt werden (Abschnitt 5.2.2). Für den Web-Hosting Dienst am LRZ könnte z.B. ein Wertebereich von fünf- bis zehntausend Transaktionen pro Sekunde definiert werden.

- **ServiceSpecification**
  Service-Specification ermöglicht Soll-Ist Vergleich

  Als Resultat der Verhandlungsphase mit dem Kunden entsteht schließlich die *ServiceSpecification*. Sie drückt eine Verfeinerung der im *ProviderCentricTemplate* enthaltenen Diensteigenschaften hinsichtlich der von einem Kunden gewünschten Werte aus und

schafft somit die Voraussetzung für einen Soll-Ist Vergleich während der Betriebsphase des Dienstes. Beispielsweise könnte eine derartige Spezifikation für den Web-Hosting Dienst am LRZ vorsehen, dass mindestens sechstausend Transaktionen pro Sekunde unterstützt werden müssen.

Aus Übersichtlichkeitsgründen wird an dieser Stelle nur die Spezifikation der Entität *ServiceSpecification* (Tabelle 6.3) dargestellt, die weiteren genannten Entitäten finden sich in Anhang A.

Entität ServiceSpecification	
Beschreibung	Diese Entität entsteht durch eine Verfeinerung des *BasicServiceTemplates* bzw. des *ProviderCentricTemplates* und stellt die Basis für die Implementierung des Dienstes für einen bestimmten Kunden dar. Sie beschreibt gewünschte Diensteigenschaften und ermöglicht so einen Soll-Ist Vergleich.
Synonym	Dieses Konzept ist in CIM nicht vorhanden; in SID existiert eine Klasse ServiceSpecification, allerdings wird nicht zwischen den verschiedenen Arten einer Spezifikation unterschieden
Prozessbezug	Wegen der zentralen Bedeutung dieser Entität besteht ein Bezug zu allen Prozessen.
Verwandte Entitäten	ManagedServiceElement (Superklasse), BasicServiceTemplate, ProviderCentricTemplate, Service, Category

Tabelle 6.3: Entität ServiceSpecification

Entität Service reflektiert Implementierung

Die zweite grundlegende Entität *Service* (Tabelle 6.4) repräsentiert den auf der Basis einer Dienstspezifikation implementierten Dienst. Sie zielt darauf ab, eine aktuelle Sicht auf den Dienst bzw. die aktuell gemessenen Werte für Diensteigenschaften zu reflektieren. Letztere werden insbesondere in Abschnitt 6.3 weiter konkretisiert.

Entität Service	
Beschreibung	Repräsentiert die Implementierung eines Dienstes gemäß einer *ServiceSpecification* und umfasst aktuell gemessene Werte für Diensteigenschaften. Weiterhin ist der Dienst bestimmten Kategorien zugeordnet und befindet sich in einer definierten Lebenszyklusphase.
Synonym	CIM definiert eine Klasse Dienst, diese trägt aber eine unterschiedliche Semantik. In SID ist eine Klasse Dienst ebenfalls vorhanden, es wird allerdings eine Unterscheidung zwischen *ResourceFacingService* und *CustomerFacingService* eingeführt.
Prozessbezug	Wegen der zentralen Bedeutung dieser Entität besteht ein Bezug zu allen Prozessen.
Verwandte Entitäten	ManagedServiceElement (Superklasse), ServiceSpecification, QoS, Category, LifecycleStatus

Tabelle 6.4: Entität Service

Zur Überführung der definierten Entitäten in ein geeignetes Modell wurde auf Specification-Pattern von Fowler [Fow96] zurückgegriffen, das gleichzeitig eine Erweiterung des Strategy-Pattern [GHJV95] von Gamma et al. darstellt. Die grundlegende Idee beider Pattern besteht dabei darin, eine Trennung von Spezifikation und Implementierung zu gewährleisten – eine Designvorgabe, die ebenfalls von dem MNM-Dienstmodell abgeleitet wurde. In Abbildung 6.3 werden diese beiden Teilbereiche verdeutlicht: Der jeweilige Stereotyp (*ServiceSpecification* oder *ServiceImplementation*) zeigt an, welcher Aufgabe die betreffende Entität untergeordnet ist. Während die verschiedenen Arten der Spezifikation in einer Vererbungshierachie angeordnet werden, weist die Relation *Specifies* darauf hin, dass eine *ServiceSpecification* einem oder mehreren Diensten zugeordnet werden kann.

Einordnung in Dienstkategorien

Weiterhin wurden in dem Modell eine Reihe von Informationsanforderungen (Abschnitte 5.2.4 und 5.3.4) umgesetzt: Um ein Auffinden des

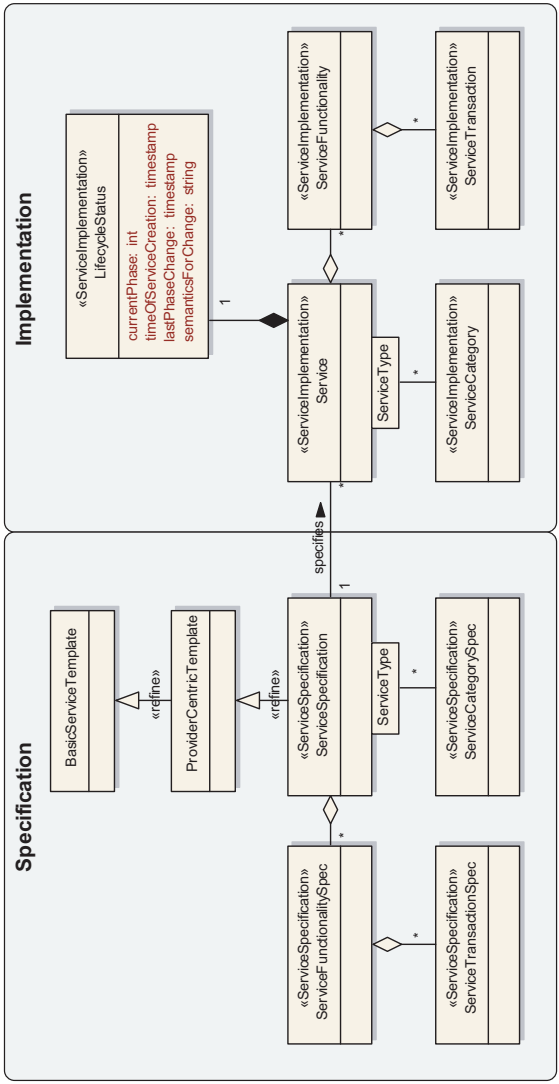

Abbildung 6.3: Klassenmodell von Dienst und Dienstspezifikation

Dienstes bzw. der Dienstspezifikation zu unterstützen, erfolgte zunächst die Einführung eines *ServiceTypes* und einer Entität *ServiceCategory*. Den Entitäten *Service* und *ServiceSpecification* kann so ein bestimmter Diensttyp zugewiesen und mit Hilfe dessen eine Einordnung in Dienstkategorien vorgenommen werden, wie durch die qualifizierten Assoziationen in Abbildung 6.3 ausgedrückt. Dies erlaubt die Einbettung in den Dienstkatalog eines Providers und vereinfacht die Teilaktivität *Initial Classification and Recording* des Incident-Management Prozesses.

Zur Beschreibung der Dienstfunktionalität dient die Entität *ServiceFunctionality*. Sie steht in Aggregationsbeziehung mit dem *Service* bzw. der *ServiceSpecification* und besteht selbst wiederum aus einer Reihe von *ServiceTransactions*. In dieser Weise können funktionale Bausteine eines Dienstes (vgl. die in [DR02] eingeführten *FunctionalBB*) in *ServiceFunctionalities* gegliedert und anhand ihrer Transaktionen weiter detailliert werden. *ServiceTransactions* charakterisieren dabei generische Dienstprimitive (z.B. Aufruf einer Webseite) und abstrahieren somit effektiv Details der Implementierungstechnologie (z.B. http-Aufrufe) – ein Mechanismus, der beispielsweise auch innerhalb der Application-MIB [KS98] Anwendung findet. Wie sich in Abschnitt 6.3 weiterhin zeigen wird, können mit Hilfe dieser Abstraktion eine Reihe generischer Dienstattribute festgelegt werden.

Beschreibung der Lebenszyklusphase

Als weitere Informationsanforderung fand die Beschreibung der Lebenszyklusphase eines Dienstes Berücksichtigung. Dazu wurde die Entität *LifecycleStatus* eingeführt und per Komposition untrennbar mit dem *Service* verbunden. Sie verfügt über Attribute zur Beschreibung der aktuellen Lebenszyklusphase des Dienstes (*currentPhase*), des Zeitpunktes, an dem der Service kreiert wurde (*timeOfServiceCreation*), des Zeitpunktes der letzten Änderung der Lebenszyklusphase (*lastPhaseChange*) und der Begründung für diese Änderung (*semanticsForChange*).

6.2.4 Service Access Point

Die Entität *ServiceAccessPoint (SAP)* dient dazu, die Schnittstelle zwischen Dienst und Dienstnutzer bzw. Dienstkunde zu beschreiben. Dies

wird notwendig, da der SAP zum einen den Übergang von Verantwortlichkeiten in der Dienstbereitstellung verkörpert und zum anderen einen potentiellen Messpunkt für Diensteigenschaften repräsentiert (siehe z.B. [Gar04b]).

Verschiedene Arten von SAPs werden berücksichtigt

Sowohl in den im vorhergehenden Kapitel ermittelten Informationsanforderungen (Tabelle 5.1) als auch in [DR02] wird dabei auf verschiedene Arten von Dienstzugangspunkten hingewiesen: Ein SAP tritt sowohl bei der Nutzung von Dienstfunktionalität als auch bei dem Zugriff auf die Customer Management Funktionalität des Dienstes auf. Beide Fälle können als Verfeinerung eines SAPs angesehen werden und finden deshalb in der Service-MIB entsprechend Berücksichtigung.

Weiterhin werden SAPs üblicherweise durch bestimmte Komponenten (z.B. *Customer Premises Equipment*, *CPE*) realisiert, die im Modell durch das Attribut *hostedOn* referenziert werden. Sie stellen wiederum verschiedene Zugangsarten zum SAP bereit, die sowohl in Kombination als auch exklusiv auftreten können und durch die folgenden Entitäten repräsentiert werden:

- **NetworkAccessProperties**
  Stützt sich der Zugriff auf den SAP auf Netzkomponenten, so werden in diesem Zusammenhang relevante Managementinformationen mit Hilfe der Entität *NetworkAcessProperties* ausgedrückt. Dies umfasst beispielsweise Parameter des Übertragungsprotokolls, den verwendeten Fehlerkorrekturalgorithmus, die VLAN-ID oder den Standort der entsprechenden Netzkomponente. Wegen des Fokus auf höherwertige Dienste werden diese Eigenschaften nicht näher in der Spezifikation berücksichtigt, es ist vielmehr eine Einbeziehung entsprechender komponentenorientierter Managementmodelle, wie das in Abschnitt 3.1.3 vorgestellte Modell der Internet Managementarchitektur, vorgesehen.

- **ApplicationAccessProperties**
  Managementinformationen hinsichtlich der an der Realisierung des SAPs beteiligter Applikationskomponenten werden in der Entität *ApplicationAccessProperties* subsummiert. Beispiele hierfür stellen Verschlüsselungsverfahren oder Protokollports dar. Analog

zu den *NetworkAccessProperties* ist diese Entität mehr als Verweis auf Objekte aus applikationsmanagementorientierten Modellen (siehe z.B. *CIM_Application* in Abschnitt 3.1.1) zu verstehen.

- **OrganizationalContact**
  Die letzte Entität umfasst für den SAP relevante organisatorische Managementinformationen, wie Ansprechpartner, Emailadresse oder Telefonnummer. Dies ist beispielsweise in Fällen wichtig, in denen die Customer-Management-Schnittstelle als Telefoncenter realisiert wird.

Kontaktinformationen werden vermerkt

Die im Mittelpunkt der genannten Aspekte stehende Entität *ServiceAccessPoint* wird in Tabelle 6.5 nocheinmal zusammengefasst, für die Spezifikation der weiteren Entitäten wird wiederum auf Anhang A verwiesen.

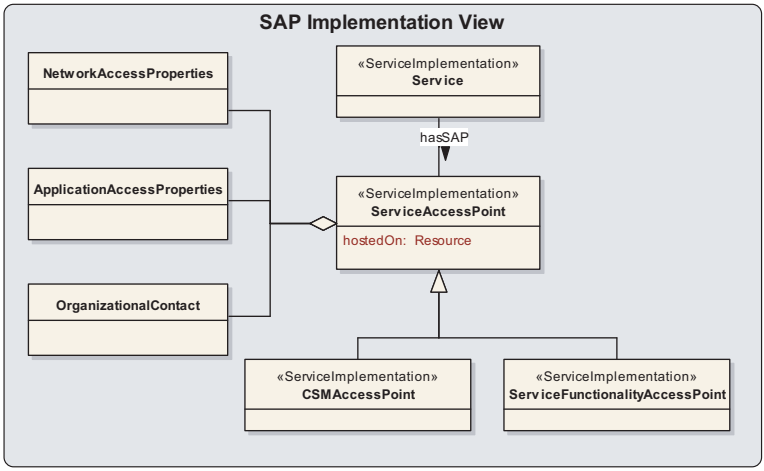

Abbildung 6.4: Klassendiagramm zu der Entität ServiceAccessPoint

Die Abbildung der genannten Entitäten in eine Klassenstruktur kann in diesem Falle einfach durchgeführt werden (siehe Implementierungsteil in Abbildung 6.4): *ServiceFunctionality*- und *CSMAccessPoint* bilden

SAP wird Dienst zugeordnet

Entität ServiceAccessPoint	
Beschreibung	Beschreibt die Schnittstelle zwischen Kunde und Provider und wird weiterhin in *ServiceFunctionalityAccessPoint* und *CSMAccessPoint* verfeinert. Mit dem SAP verbundene technische Managementinformationen werden nach *Network-* und *ApplicationAcessProperties* unterschieden; organisatorische Informationen finden sich in der Entität *OrganizationalContact*
Synonym	Eine Klasse *ServiceAccessPoint* ist in CIM vorhanden, die genannten Unterscheidungen werden allerdings nicht berücksichtigt. In SID wird keine Klasse mit ähnlicher Semantik definiert.
Prozessbezug	Vorwiegend Service-Level, Incident, Problem und Change Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), ServiceSpecification, Service, NetworkAccessProperties, ApplicationAccessProperties

Tabelle 6.5: Entität ServiceAccessPoint

jeweils eine Verfeinerung der Klasse *ServiceAccessPoint*. Um die Zuordnung eines SAPs zu einem Service vornehmen zu können, wurde ferner die Assoziation *hasSAP* eingeführt. Hierbei gilt es zu beachten, dass in Abbildung 6.4 nur der Implementierungsteil eines SAPs (Stereotyp *ServiceImplementation*) dargestellt wird, aufgrund der in Abschnitt 6.2.3 dargelegten Symmetrie aber die beschriebenen Entitäten und Assoziationen gleichermaßen in einer *ServiceSpecification* auftreten.

Weiterhin stehen die Klassen *NetworkAccessProperties*, *ApplicationAccessProperties* und *OrganizationalContact* jeweils in einer Aggregationsbeziehung zu dem *ServiceAccessPoint*. Wie bereits erwähnt, stellen sie vorwiegend Verweise auf in anderen Managementmodellen gepflegte Daten dar, welche durch entsprechende Gateways integriert werden müssen (siehe Kapitel 7).

6.2.5 Quality-of-Service (QoS)

Die Einführung eines Qualitätsbegriffes für die erbrachten Dienste (Quality-of-Service) stellt eine elementare Herausforderung für IT-Provider dar. Entsprechend müssen in der Servie-MIB Entitäten eingeführt werden, die eine QoS-Spezifikation abbilden und einen Vergleich aktuell gemessener Qualitätsparameter mit den Zielvorgaben für diese Parameter erlauben (siehe Informationsanforderungen in Tabelle 5.1).

Spezifikation und Implementierung wird unterschieden

In der Spezifikation qualitätsbezogener Eigenschaften wird deshalb konsequent das mit der Entität Dienst eingeführte Designprinzip, Implementierung und Spezifikation zu trennen, weiter verfolgt. Dies mündet in zwei grundlegenden Entitäten, die jeweils mit dem *Service* bzw. der *ServiceSpecification* in Beziehung stehen: Zunächst bildet die *ServiceQualitySpecification* (Tabelle 6.6) während der Verhandlungsphase zwischen Kunde und Provider entstehende Zielvorgaben hinsichtlich der Dienstqualität ab. Sie besteht aus einer Reihe von QoS-Parametern, die wiederum einzelne QoS-Eigenschaften des Dienstes beschreiben. Für die Modellierung von QoS-Parametern wird dabei auf eine Kombination von [Roe05] und [DR02] zurückgegriffen und somit die gewünschte Einbettung in die Forschungslandschaft des MNM-Teams gewährleistet (vgl. Abschnitt 6.2). Demnach werden folgende Bestandteile von QoS-Parametern berücksichtigt:

- **Beschreibung der Semantik**
  Für eine eindeutige und unmissverständliche Spezifikation von QoS-Parametern stellt zunächst die Beschreibung der Semantik (Entität *SemanticDescription)* einen unverzichtbaren Bestandteil dar. Dies wird notwendig, da Parameter mit dem gleichen Bezeichner prinzipiell grundlegend verschiedene Sachverhalte ausdrücken können: Beispielsweise lässt sich die Verfügbarkeit eines Dienstes einerseits über die Anzahl an Dienstausfällen während eines bestimmten Zeitraums definieren, andererseits kann dafür das Verhältnis von beantworteten Anfragen zu allen gestellten Anfragen herangezogen werden. Weiterhin bestimmt die Auswahl der Messpunkte maßgeblich die Semantik eines QoS-Parameters und sollte deshalb in einer formalen Spezifikation berücksichtigt werden [Gar04b].

- **Garantietyp**
  Mit der Entität *GuarenteeType* wurde weiterhin eine Beschreibungsmöglichkeit hinsichtlich der Garantiegüte für die Einhaltung von QoS-Parametern eingeführt. Sie erlaubt – über weitere Spezialisierung – eine Unterscheidung zwischen dem Garantietyp *Best Effort*, der keine Zusicherungen enthält sowie relativen und absoluten Garantien. Während relative Garantien typischerweise Zusicherungen in der Form von Relationen (z.B. "besser als") enthalten, drücken absolute Garantien entweder aus, dass sie zu jedem Zeitpunkt (*deterministische Garantie*) oder über ein statistisches Mittel (*statistische Garantie*) eingehalten werden (siehe auch [Roe05]).

- **Wertemenge**
  Neben dem Typ des QoS-Parameters, der als Attribut der Entität QoS-Parameter beschrieben wird, ist es zusätzlich noch notwendig, mögliche Wertemengen (*ValueRange*) für den betreffenden Parameter festzulegen. Wertemengen können dabei u.a. als Intervalle oder Liste möglicher Belegungen auftreten und somit z.B. vom Provider eingeführte Qualitätsstufen (z.B. Gold, Silber, Bronze) abbilden. In welcher Weise die einzelnen Werte dieser Menge interpretiert werden müssen, legt dabei wiederum die *SemanticDescription* fest.

Vom Provider eingeführte Qualitätsstufen können abgebildet werden

Ein weiterer Modellierungsaspekt, der in [Roe05] eingeführt wird, betrifft die Art des QoS-Parameters: Hier wird zwischen Volumen-, Geschwindigkeits- und Korrektheitsparametern unterschieden. Diese Aufteilung reflektiert allerdings die spezifischen Gegebenheiten von Netz-QoS Architekturen und wurde deshalb im vorliegenden Modell nicht berücksichtigt.

Weiterhin gilt es zu beachten, dass einem gegebenen Dienst mehrere *ServiceQualitySpecifications* zugeordnet werden können. Eine QoS-Spezifikation repräsentiert somit eine QoS-Kategorie, die Parameter mit bestimmten Eigenschaften subsummiert. In dieser Weise können vom Provider vorgenommene Paketierungen hinsichtlich QoS-Eigenschaften (z.B. Gold-Paket für Netz-QoS) abgebildet und einfach wiederverwendet werden (Attribut *qoSPackage*).

Entität ServiceQualitySpecification	
Beschreibung	Diese Entität repräsentiert Zielvorgaben hinsichtlich der Qualitätseigenschaften eines Dienstes. Sie setzt sich aus mehreren QoS-Parametern zusammen und wird von einem bestimmten QoS-Dienst realisiert. Da die Spezifikation mehrere QoS-Parameter bündelt, dient sie u.a. zur Modellierung von QoS-Kategorien.
Synonym	Eine entsprechende Klasse ist weder in CIM noch in SID existent. In CIM können QoS-Eigenschaften anhand des *CIM_Metrics*-Modells dargestellt werden, in SID bietet sich dafür die Klasse *ServiceCharacteristic* an.
Prozessbezug	Service-Level Management, Capacity Management, Continuity Management
Verwandte Entitäten	ServiceSpecification, ServiceQuality, ManagedServiceElement (Superklasse), QoSParameter

Tabelle 6.6: Entität ServiceQualitySpecification

ServiceQuality stellt Implementierung dar

Die nächste grundlegende Entität zur Beschreibung qualitätsbezogener Eigenschaften bildet die *ServiceQuality*. Sie repräsentiert die aktuell gemessenen Qualitätseigenschaften eines Dienstes, stellt die Implementierung einer bestimmten *ServiceQualitySpecification* dar und umfasst ebenfalls eine Reihe von QoS-Parametern. Letztere verkörpern dementsprechend eine Verfeinerung der vorab spezifizierten QoS-Parameter hinsichtlich der konkreten Dienstinstanz, d.h. sie setzen sich aus den gleichen Bestandteilen zusammen, reflektieren aber die technischen Spezifika der Dienstimplementierung: Durch die Abbildung auf QoS-Mechanismen einer bestimmten QoS-Architektur können sich so beispielsweise abweichende Wertemengen für einen Parameter ergeben. Ebenso denkbar ist eine Konkretisierung dahingehend, dass technische Umsetzungsaspekte in die Beschreibung des QoS-Parameters aufgenommen werden. Im Hinblick auf [DR02] bildet diese Verfeinerung von QoS-Parametern den Übergang von dem *customer-centric service template*

auf die entsprechende *service template instance* ab. Tabelle 6.7 fasst die Spezifikation der Entität *ServiceQuality* zusammen.

Entität ServiceQuality	
Beschreibung	Die Entität repräsentiert die Implementierung einer bestimmten *ServiceQualitySpecification* und beschreibt die aktuell gemessene Dienstqualität. Sie setzt sich ebenfalls aus QoS-Parametern zusammen, die eine Verfeinerung der spezifizierten Parameter hinsichtlich technischer Implementierungsdetails darstellen.
Synonym	In CIM wird eine Klasse *QoS* als Teil des *CIM_Network* Modells definiert, sie bezieht sich aber vorwiegend auf Netz-QoS Dienste. SID enthält momentan keine Entität mit ähnlicher Semantik, es bietet sich jedoch eine Modellierung über die Klasse *ServiceCharacteristic* an.
Prozessbezug	Service-Level Management, Capacity Management, Continuity Management
Verwandte Entitäten	Service, QoSService, ServiceQualitySpecification, ManagedServiceElement (Superklasse), QoSParameter

Tabelle 6.7: Entität ServiceQuality

Beschreibung von QoS-Diensten und deren Kopplung

Eine weitere Anforderung, die innerhalb der Informationsanalyse identifizert wurde (siehe Tabelle 5.1), betrifft die Beschreibung von QoS- und Koppeldiensten. Während QoS-Dienste die technische Implementierung der Dienstqualität mit Hilfe einer bestimmten QoS-Architektur repräsentieren (z.B. DiffServ [BBC+98]), stellen Koppeldienste den Übergang zwischen verschiedenen QoS-Architekturen bzw. QoS-Diensten her [Roe05]. Zur Einbettung beider Dienste in die Service-MIB werden zwei, miteinander in Beziehung stehende Entitäten eingeführt: Die Entität *QoSService* entsteht durch eine Verfeinerung der Entität *Service*. Sie beschreibt die technische Umsetzung der *ServiceQuality*, was über die mit dem Stereotyp *realize* versehene Relation ausgedrückt wird.

Koppeldienste werden ihrerseits als Assoziationen zwischen zwei QoS-Diensten modelliert und durch die Entität *QoSMappingService* repräsentiert. Letztere entsteht ebenfalls durch eine Verfeinerung der Entität *Service*.

Entität QoSService	
Beschreibung	Repräsentiert die technische QoS-Implementierung mit Hilfe einer bestimmten QoS-Architektur und stellt eine Verfeinerung der Entität *Service* dar. Die Abbildung zwischen verschiedenen QoS-Diensten wird durch die Entität *QoSMappingService* näher umschrieben.
Synonym	Sowohl in CIM als auch SID ist eine gleichlautende Klasse mit ähnlicher Semantik vorhanden. Die in diesen Modellen definierten Verfeinerungen können entsprechend einfach in die Service-MIB integriert werden und umgekehrt.
Prozessbezug	Service-Level Management
Verwandte Entitäten	Service, QoSMappingService, ManagedSystemElement (Superklasse)

Tabelle 6.8: Entität QoSService

Nachfolgend findet sich die Spezifikation der Entität *QoSService* in Tabelle 6.8, auf die entsprechende Darstellung des *QoSMappingService* in Anhang A wird verwiesen.

Aufteilung in Spezifikation und Implementierung wird beibehalten

Die Zusammenführung der in diesem Abschnitt beschriebenen Entitäten in eine Klassenstruktur veranschaulicht das in Abbildung 6.5 dargestellte QoS-Modell. Zunächst wird die symmetrische Aufteilung in Spezifikations- und Implementierungsteil augenscheinlich: Die Entitäten *ServiceQualitySpecification* und *ServiceQuality* stehen miteinander über die *specifies*-Assoziation in Beziehung und stellen einen Teil der *ServiceSpecification* bzw. des *Service* dar. Sie können einem bestimmten QoS-Paket zugeordnet werden (*qoSPackage*) und bestehen aus einer Reihe von QoS-Parametern, die sich wiederum aus *GuaranteeType*, *SemanticDescription* und *ValueRange* zusammensetzen. Zu beachten ist

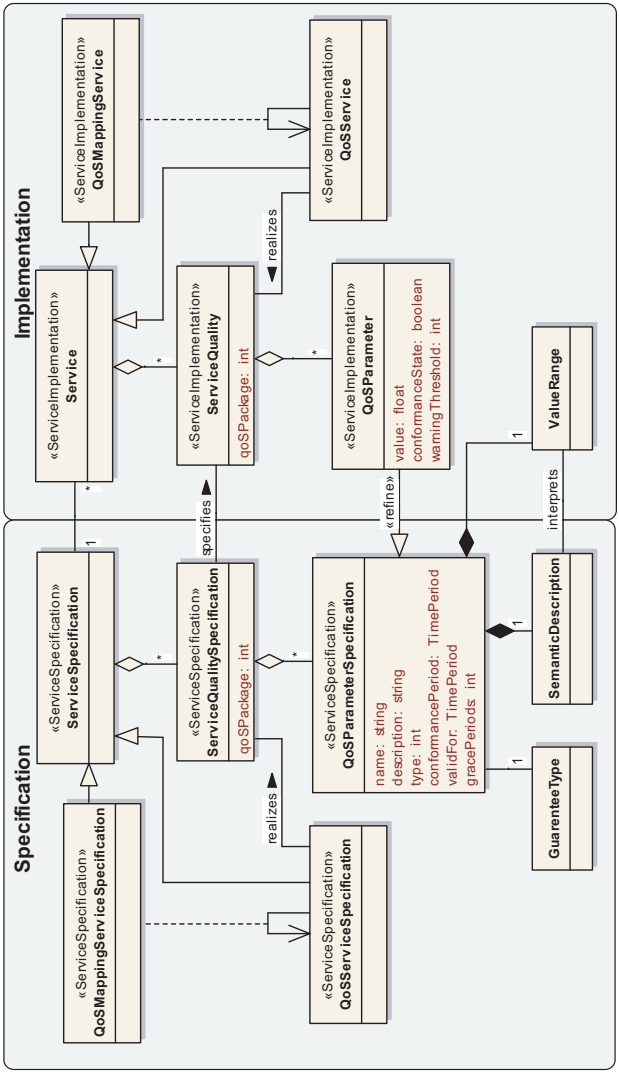

Abbildung 6.5: Klassendiagramm zu der Entität QoS

hierbei, dass diese Aufteilung nur für die QoS-Spezifikation im Modell Darstellung findet, aber durch die Verfeinerungsbeziehung (*refine*) effektiv auch für den Implementierungsteil vorhanden ist.

Spezifikations- und Implementierungsteil des *QoSService* repräsentieren jeweils eine Verfeinerung der Entitäten *ServiceSpecification* bzw. *Service*. Um auszudrücken, dass dieser Dienst für die Umsetzung einer bestimmten Dienstqualität zuständig ist, wurde eine entsprechende Assoziation mit dem Stereotyp *realize* eingeführt. Schließlich finden Koppeldienste (*QoSMappingService*) als Assoziationsklassen zwischen QoS-Diensten Einzug in das Modell; sie gliedern sich ebenfalls in Spezifikations- und Implementierungsteil (Stereotypen *ServiceSpecification* und *ServiceImplementation*).

6.2.6 Service Level Agreement

In enger Beziehung mit der Dienstqualität steht das zwischen Kunde und Provider vereinbarte *Service Level Agreement*: Es legt Zielvorgaben bzgl. der QoS-Parameter fest und bestimmt somit letztendlich deren Ausprägung. Wie aus der Informationsanalyse des Service-Level Managementprozesses (Abschnitt 5.3.3) hervorgeht, ist es weiterhin notwendig, die aus einer Nichteinhaltung der Zielvorgaben resultierenden Pönalen als Teil der Service-MIB aufzufassen.

SLATemplate stellt Ausgangslage dar

Um die genannten Sachverhalte darzustellen wurden drei grundlegende Entitäten geschaffen, die im Folgenden vorgestellt werden. Zunächst bildet die Entität *SLATemplate* den Ausgangspunkt für die weitere Modellierung. Sie fungiert als Container für generische und unparametrisierte SLA-Beschreibungen, die dazu benutzt werden können, gemeinsame Eigenschaften von SLAs für eine bestimmte Klasse von Diensten zu erfassen (siehe auch Abschnitt 8). Ausgehend von dem *SLATemplate* entsteht durch Verfeinerung das konkrete *SLA* (siehe Tabelle 6.9), das entsprechend mit einem bestimmten *Service* und *ServiceCustomer* (siehe Abschnitt 6.2.7) verknüpft wird und zudem eine Gültigkeitsperiode (*applicableDuring*) aufweist. Gemäß der in Abschnitt 5.3.4 identifizierten Informationsanforderungen ergibt sich zusätzlich die Notwendigkeit, den Typ des SLAs zu dokumentieren, wofür die Entität *AgreementType* dient. Sie beschreibt, ob es sich um ein *Operational Level Agree-*

SLA wird mit Dienst und Dienstkunde verknüpft

Entität SLA	
Beschreibung	Diese Entität beschreibt die vertraglichen Vereinbarungen zwischen Provider und Dienstnutzer hinsichtlich eines konkreten Dienstes. Ein SLA trägt einen bestimmten Typus (*AgreementType*) und besteht aus einer Reihe von Zielvorgaben (*Objectives*) und Pönalen, falls diese nicht eingehalten werden können.
Synonym	In CIM ist ein entsprechendes Konzept nicht vorhanden; die Modellierung in SID weist Ähnlichkeiten mit der vorliegenden auf, eine Klasse *ServiceLevelAgreement* ist vorhanden.
Prozessbezug	Service-Level Management, Incident Management, Problem Management, Availability Management, Capacity Management, Financial Management, Continuity Management
Verwandte Entitäten	SLATemplate, AgreementType, ServiceLevelObjective, ServiceLevelPenalties, Service, ServiceCustomer

Tabelle 6.9: Entität SLA

ment oder einen *Underpinning Contract* handelt und ob der SLA als *Service-based*, *Customer-based* oder *Multi-level SLA* konzipiert ist (vgl Abschnitt 5.3.3).

Spezifikation von QoS-Parametern basiert auf SLOs

Einen wesentlichen Bestandteil des SLAs stellen die mit der Entität *ServiceLevelObjectives* (Tabelle 6.10) repräsentierten Zielvereinbarungen hinsichtlich der Dienstbereitstellung dar. Sie legen u.a. die erwarteten Qualitätseigenschaften des Dienstes fest und bilden somit die Basis für eine Spezifikation von QoS-Parametern, was in dem Modell durch eine entsprechende Assoziation (*MappedUpon*) ausgedrückt wird. Nach welchem Mechanismus *ServiceLevelObjectives* auf QoS-Parameter abgebildet werden (*vertikales QoS-Abbildungsproblem*), führt man dabei mit Hilfe der Entität *SLOToQoSMapping* weiter aus. Sie wurde zum gegenwärtigen Zeitpunkt als abstrakte Entität definiert, eine weitere

Entität ServiceLevelObjective	
Beschreibung	Verkörpert die in einem SLA enthaltenen Zielvorgaben hinsichtlich der Qualitätseigenschaften eines Dienstes. Zur technischen Realisierung werden SLOs nach einem mit der Entität *SLOToQoSMapping* beschriebenen Schema auf QoS-Parameter abgebildet. Sie stehen in Beziehungen zu *ServiceLevelPenalties*, die Konsequenzen einer Nichteinhaltung von SLOs beschreiben und sind mit einer BusinessCriticality versehen.
Synonym	Nicht vorhanden in CIM; in SID existiert eine Klasse mit gleichem Namen und ähnlicher Semantik
Prozessbezug	Service-Level Management, Incident Management, Problem Management, Availability Management, Capacity Management, Financial Management, Continuity Management
Verwandte Entitäten	SLA, QoSParameter,SLOToQoSMapping, ServiceLevelPenalties

Tabelle 6.10: Entität ServiceLevelObjective

Verfeinerung stellt einen Gegenstand für weiterführende Arbeiten dar. Objectives werden ferner durch ihre Beschreibung, ihre Gültigkeitsdauer (*validFor*) und ihren Stellenwert für die Geschäftsziele (*businessCriticality*) charakterisiert. Letzterer kann insbesondere mit Hilfe der im nächsten Abschnitt beschriebenen Entität berechnet werden.

Beschreibung von Pönalen ermöglicht Impact Analyse

Als letzter Bestandteil des SLA-Teilmodells der Service-MIB wurde die Entität *ServiceLevelPenalty* eingeführt. Sie repräsentiert Pönalen, die aus einer Nichteinhaltung eines *Service Level Objectives* resultieren (*UnmetResultsIn*) und beschreibt somit die Auswirkungen einer SLA-Verletzung auf die Geschäftsvorgänge des Unternehmens. Zu welchen Aktionen der Provider in diesem Fall verpflichtet ist, wird dabei durch das Attribut *action* charakterisiert; denkbar sind Strafzahlungen, Ersatzleistungen oder Preisnachlässe. In dieser Weise wird der gewünschte

Entität ServiceLevelPenalty	
Beschreibung	Diese Entität repräsentiert die Auswirkungen von Nichteinhaltungen eines *SLAs* bzw. *ServiceLevelObjectives*. Jeder ServiceLevelPenalty ist eine bestimmte Aktion zugeordnet, die die Form der Ausgleichsleistung beschreibt. Ferner stellt diese Entität die Basis für die Berechnung einer *businessCriticality* von *ServiceLevelObjectives* dar.
Synonym	Ein ähnliches Konzept wurde in CIM nicht definiert. Das Pendant zu dieser Entität in SID bildet die Klasse *ServiceLevelSpecConsequence*.
Prozessbezug	Service-Level, Incident, Problem, Availability, Capacity, Financial und Continuity Management
Verwandte Entitäten	SLA, ServiceLevelObjective

Tabelle 6.11: Entität ServiceLevelPenalty

Bezug zu der Teilaktivität *Classification and Initial Support* des Incident Management Prozesses hergestellt und somit eine Impact Analyse ermöglicht (siehe Tabelle 6.11).

Die Einbettung der genannten Aspekte in das Service-MIB Modell veranschaulicht Abbildung 6.6, die im Folgenden, beginnend mit der Entität SLA, erläutert wird. Die oben genannten Beziehungen zwischen *SLA*, *Service* und *ServiceCustomer* wurden mit Hilfe der *underlies* bzw. *agreesUpon* Assoziation modelliert. Die Entitäten *AgreementType*, *ServiceLevelPenalty* und *ServiceLevelObjective* sind als Kernbestandteile des SLAs anzusehen und werden entsprechend als Aggregationen dargestellt. Um die Abbildung von *ServiceLevelObjectives* auf *QoSParameter* zu modellieren, wurde die Entität *SLOToQoSMapping* als (abstrakte) Assoziationsklasse eingeführt.

Weiterhin bleibt zu erwähnen, dass die vorliegende Modellierung in Einklang mit dem in Abschnitt 3 vorgestellten SLA Management Handbook [GB905] vorgenommen wurde, was die Interoperabilität mit bestehenden Ansätzen auf diesem Gebiet begünstigt.

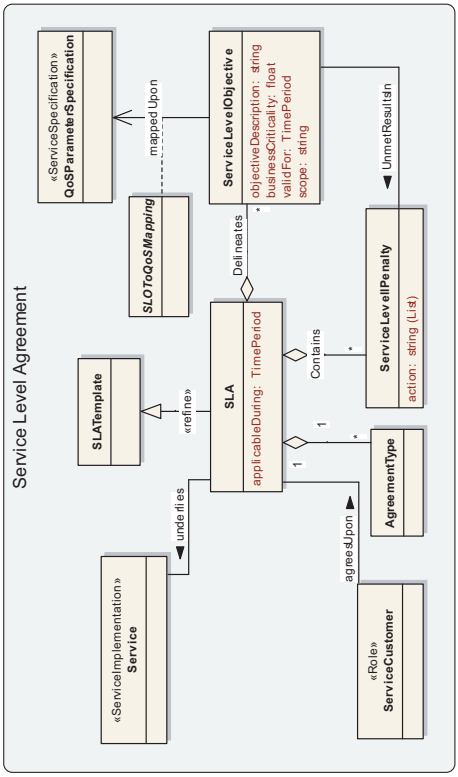

Abbildung 6.6: Klassendiagramm zu der Entität SLA

6.2.7 Dienstnutzer und Dienstkunden

Im letzten Abschnitt wurde mit dem Dienstkunden bereits eine wichtige Rolle erwähnt und in Beziehung zu dem SLA gesetzt. Im weiteren Verlauf erfolgt nun die umfassende Spezifikation rollenbezogener Entitäten und deren Einbettung in das Service-MIB Modell. Wie aus Tabelle 5.1 hervorgeht, gilt es dabei Informationsanforderungen dahingehend zu

berücksichtigen, dass eine Zuordnung kundenbezogener Daten zu den elementaren Rollen Dienstnutzer, Dienstkunde und Provider ermöglicht wird. Weiterhin wurde angemerkt, dass die Pflege von Kontaktinformationen nicht als Aufgabe der Service-MIB verstanden wird, sondern vielmehr Verweise auf die entsprechenden Daten geschaffen werden müssen.

Party-Pattern findet Anwendung

Um organisationsbezogene Entitäten möglichst flexibel in das Service-MIB Modell zu integrieren, findet das Party Designpattern von [Fow96] Verwendung. Dessen grundlegende Idee besteht darin, Personen und Organisationen als *Party* zu abstrahieren, wie auf der linken Seite in Abbildung 6.7 dargestellt. Für die weitere Modellierung bietet dies den Vorteil, Rollen einer *Party* zuordnen zu können und keine Unterscheidung zwischen Personen und Organisationen durchführen zu müssen. Entstprechend wird, wie in dem Modell ersichtlich, die Rolle, die eine bestimmte *Party* einnimmt, mit der Entität *Role* repräsentiert und über die *HasRole* Assoziation verknüpft.

Grundlegende Rollen: Dienstnutzer, -kunde und Provider

Als fundamentale Rollen unterscheidet man – gemäß den Informationsanforderungen – in Dienstnutzer, Dienstkunde und Provider. Diese werden durch die Entitäten *ServiceUser, ServiceCustomer* und *ServiceProvider* repräsentiert, welche wiederum eine Verfeinerung der Entität *Role* darstellen und im Kontext eines bestimmten Dienstes erscheinen (Aggregation *boundTo*). In der Rolle *ServiceUser* (Tabelle 6.12) treten dabei Personen oder Organisationen auf, die die Nutzungsfunktionalität eines Dienstes an einem bestimmten *ServiceFunctionalityAccessPoint* konsumieren, wie mit der Assoziation *usesFunctionality* ausgedrückt wird.

Dienstkunde steht mit SLA und CSM in Beziehung

Wie bereits im letzten Absatz aufgezeigt wurde, tritt die Rolle *ServiceCustomer* vorwiegend im Zusammenhang mit den vertraglichen Vereinbarungen für den Dienst auf und wird entsprechend über die *agrees* Assoziation mit der Entität *SLA* in Beziehung gesetzt (siehe Abbildung 6.6). Weiterhin aufgenommen in das Modell wurde der Zugriff des Dienstkunden auf die Managementfunktionalität des Dienstes, was sich in der *usesCSM* Assoziation zwischen *CSMAccessPoint* und *ServiceCustomer* widerspiegelt. Die genannten Aspekte fliessen in die Definition letztgenannter Entität, die in Tabelle 6.13 dargestellt, mit ein.

Provider wird zusätzlich mit Dienst verknüpft

Wie aus den Informationsanforderungen hervorgeht, ist es insbesondere im Falle verketteter Dienste erforderlich, die verschiedenen beteiligten

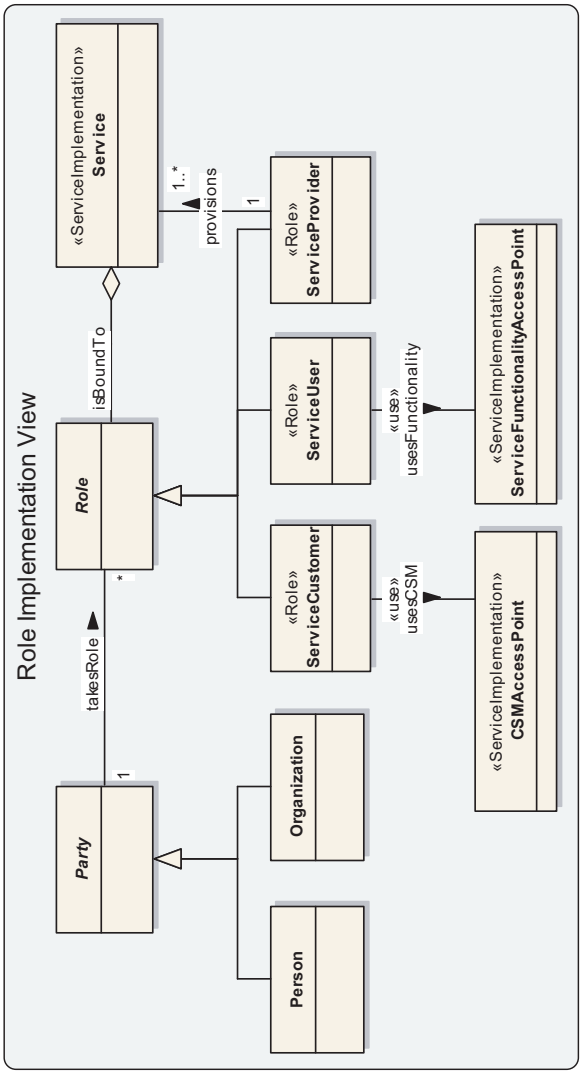

Abbildung 6.7: Klassendiagramm rollenbezogener Entitäten

Entität ServiceUser	
Beschreibung	Repräsentiert den Dienstnutzer, der die Nutzungsfunktionalität des Dienstes mit Hilfe eines *SAP* in Anspruch nimmt. Der *ServiceUser* tritt dabei als Rolle einer bestimmten *Party* auf, über die die tatsächlichen Kontaktinformationen bereitgestellt werden.
Synonym	CIM verfügt nicht über die Party-Abstraktion, es wird aber eine Klasse *User* im *CIM_User* Modell definiert, die noch der Klasse *Role* zugeordnet werden müsste. In SID werden Nutzer und Kunden nicht unterschieden, beide Entitäten werden unter der Klasse *Customer* zusammengefasst.
Prozessbezug	Wird von allen Prozessen benötigt.
Verwandte Entitäten	Role, Party,ServiceFunctionalityAccesPoint

Tabelle 6.12: Entität ServiceUser

Provider bzw. die von ihnen bereitgestellten (Sub-)Dienste identifizieren zu können. Dieser Aufgabe kommt die Entität *ServiceProvider* nach. Sie ist zusätzlich über die *provisions* Assoziation mit dem Dienst verknüpft, was die gewünschte Nachverfolgbarkeit gestattet. Abschließend zu der Definition organisationsbezogener Entitäten soll noch einmal darauf hingewiesen werden, dass die Pflege der eigentlichen Kontaktdaten von *ServiceUser, ServiceCustomer* und *ServiceProvider* gemäß gängiger Unternehmenspraxis nicht als Aufgabe der Service-MIB angesehen wird. Entsprechend ist die Entität *Party* als Verweis auf ein externes Verwaltungssystem (z.B. ein Identity Management System [Hom05]) zu verstehen.

6.2.8 Resource

Entscheidend für die Erfüllung von Dienstmanagementaufgaben ist, wie in Abschnitt 5.3.4 erläutert, die Kenntnis der Querbezüge zwischen dem Dienst und den Komponenten, die diesen Dienst realisieren. Wäh-

	Entität ServiceCustomer
Beschreibung	Diese Entität beschreibt den Dienstkunden, der in vertragliche Vereinbarungen(*SLAs*) hinsichtlich des Dienstes eintritt (*agrees* Assoziation). Weiterhin besteht eine *usesCSM* Beziehung zu dem *CSMAccessPoint*, die eine Nutzung der Customer Service Management Funktionalität reflektiert.
Synonym	In CIM existiert zwar eine Klasse *Role*, die Zuordnung zu einer *OrganizationalUnit* müsste allerdings noch definiert werden. In SID weist die Klasse *Customer* Ähnlichkeit zu dieser Entität auf.
Prozessbezug	Wird von allen Prozessen benötigt.
Verwandte Entitäten	Role, Party, CSMAccessPoint

Tabelle 6.13: Entität ServiceCustomer

rend die Modellierung dieser Abhängigkeitsbeziehungen Gegenstand des nächsten Abschnitts darstellt, ist es zunächst notwendig, eine geeignete Repräsentation für Komponenten in das Modell zu integrieren. Diese Aufgabe erfüllt die in Tabelle 6.14 dargestellte (abstrakte) Entität *Resource*. Sie dient als Basisentität, aus der durch Spezialisierung detaillierte Klassen zur Beschreibung von Netz- und Applikationskomponenten entstehen und mit dem *Service* assoziiert werden können.

Resource stellt Verweis auf bestehende Mgmt.-Modelle dar

Da bestehende Managementmodelle (siehe Kapitel 3) bereits umfassende und in der Praxis etablierte Beschreibungen von Netz- und Applikationskomponenten zur Verfügung stellen, wird in dem Service-MIB Modell die Modellierung von Ressourcen nicht weiter konkretisiert. Vielmehr soll die Entität Resource einen Platzhalter für den Informationsübergang aus bestehenden Modellen (z.B. CIM) repräsentieren. Mit welchen Mechanismen dieser Übergang realisiert werden kann, wird in Abschnitt 7.2 weiter vertieft.

Entität Resource	
Beschreibung	Diese Entität repräsentiert Netz- und Applikationskomponenten, die an der Bereitstellung eines Dienstes beteiligt sind. Sie dient als Verweis auf entsprechende Informationen aus anderen Managementmodellen und wird nicht innerhalb der Service-MIB gepflegt bzw. konkretisiert.
Synonym	Eine vergleichbare Semantik trägt die Klassen *ManagedSystemElement* im *CIM_Core* Modell, bzw. die Klasse *Resource* in SID.
Prozessbezug	Wird von allen Prozessen benutzt.
Verwandte Entitäten	ManagedServiceElement (Superklasse), Service, ServiceResourceDependency

Tabelle 6.14: Entität Resource

6.2.9 Abhängigkeitsbeziehungen zwischen Diensten und Ressourcen

Als letzter Aspekt in der Definition grundlegender Entitäten werden in diesem Abschnitt Abhängigkeitsbeziehungen zwischen Diensten sowie zwischen Diensten und Ressourcen betrachtet. Entsprechend der in Tabelle 5.3 dargestellten Informationsanforderungen gilt es dabei eine Differenzierung zwischen verschiedenen Typen von Abhängigkeitsbeziehungen vorzunehmen: Dies betrifft zunächst die Unterscheidung, ob Abhängigkeiten organisatorische oder funktionale Beziehungen ausdrücken. Während organisatorische Abhängigkeiten vorwiegend bei Auslagerung von Teilen des Dienstes an Drittanbieter zum Tragen kommen, stellt die Beschreibung funktionaler Abhängigkeiten eine wichtige Prämisse für die Nachverfolgung von Dienstfehlern dar. Wie in Abschnitt 5.3.2 begründet wurde, ist es dabei weiterhin von Relevanz, ob funktionale Abhängigkeiten zwischen Diensten oder zwischen Diensten und Ressourcen auftreten.

Organisatorische und funktionale Abhängigkeiten werden unterschieden

Die genannten Differenzierungsmerkmale spiegeln sich im Aufbau des Klassenmodells für Abhängigkeitsbeziehungen (Abbildung 6.8) wider.

Dependency stellt Oberklasse dar

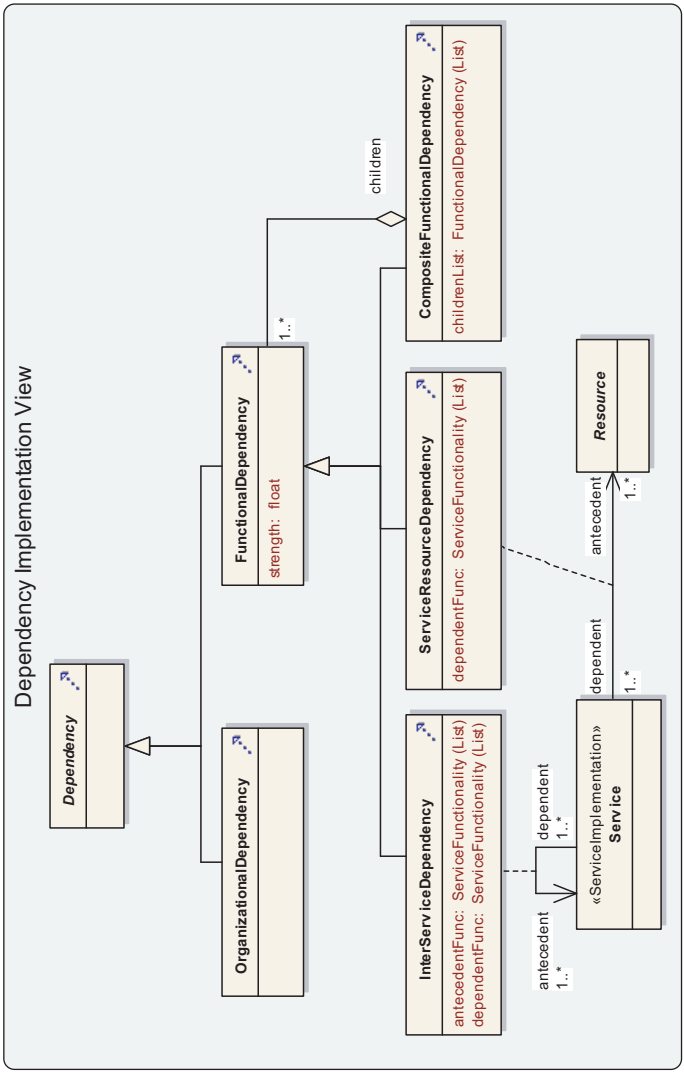

Abbildung 6.8: Klassendiagramm für Abhängigkeiten

Zunächst wurde eine abstrakte Basisentität *Dependency* eingeführt, die als gemeinsamer Vorfahre aller zur Abhängigkeitsmodellierung verwandter Entitäten fungiert. Aus einer Verfeinerung der Basisentität entstehen die Entitäten *OrganizationalDependency* und *FunctionalDependency*; sie reflektieren die genannte Unterscheidung zwischen funktionalen und organisatorischen Abhängigkeiten. Weiterhin wurde als gemeinsame Eigenschaft aller funktionaler Abhängigkeiten in Anlehnung an [HMSS06] das Attribut *strength* einführt. Es beschreibt die Wichtigkeit der Abhängigkeit für den operativen Zustand der Entität, von der die Abhängigkeitsbeziehung ausgeht und dient somit zur Berechnung des Impacts von Störungen (vgl. Abschnitt 5.3.2).

Entität InterServiceDependency	
Beschreibung	Dient zur Modellierung funktionaler Abhängigkeitsbeziehungen zwischen Diensten. Der abhängige Dienst tritt dabei in der Assoziationsrolle *dependent* auf, die (Sub-)Dienste zu denen die Abhängigkeit besteht, in der Rolle *antecedent*. Weiterhin zeigt das von der Superklasse FunctionalDependency geerbte Attribut strength an, inwiefern sich ein Ausfall des *antecedent*-Dienstes auf die Funktionalität des *dependent*-Dienstes auswirkt.
Synonym	In CIM besitzt die Klasse *ServiceServiceDependency* aus dem *CIM_Core* Modell eine ähnliche Semantik. SID sieht eine vergleichbare Abhängigkeitsmodellierung nicht vor, am nächsten kommt die Assoziation *CFServiceRequiresRFService*.
Prozessbezug	Incident, Problem, Capacity, Configuration Management
Verwandte Entitäten	Service, Dependency, FunctionalDependency

Tabelle 6.15: Entität InterServiceDependency

Ausgehend von der Entität *FunctionalDependency* wurde eine weitere

Verfeinerung hinsichtlich der zueinander in Beziehung stehenden Entitäten vorgenommen:Während Abhängigkeiten zwischen Diensten durch die Entität *InterServiceDependency* (Tabelle 6.15) repräsentiert wird, konzeptualisiert die Entität *ServiceResourceDependency* (Tabelle 6.16) Abhängigkeiten zwischen Diensten und Ressourcen.

Funktionale Abhängigkeiten werden weiter differenziert

Entität ServiceResourceDependency	
Beschreibung	Diese Entität repräsentiert funktionale Abhängigkeiten zwischen Diensten und Ressourcen. Letztere bekleiden dabei die Assoziationsrolle *antecedent*, der abhängige Dienst tritt als Rolle *dependent* auf. Potentielle Auswirkungen eines Ressourcenausfalls werden mit dem vererbten Attribut *strength* beschrieben.
Synonym	Das Pendant in CIM stellt die Klasse *DeviceServiceDependency* aus dem *CIM_Core* Modell dar. Eine ähnliche Funktion erfüllt in SID die Assoziation *LogicalResourcesImplementRFS*. Allerdings unterscheidet SID zwischen Customer- und Ressource-Facing Service, wodurch die Abhängigkeitsmodellierung inhärent eine abweichende Semantik besitzt.
Prozessbezug	Incident, Problem, Capacity, Configuration Management
Verwandte Entitäten	Service, Resource, Dependency, FunctionalDependency

Tabelle 6.16: Entität ServiceResourceDependency

Zusätzlich wurde in der Modellierung ein weiterer Fall berücksichtigt: Oftmals erweist es sich als nützlich, mehrere Abhängigkeitsbeziehungen als Einheit zu betrachten, um so beispielsweise Fehler zu erkennen, die von einer bestimmten Ressource verursacht werden. Ermöglicht wird dies durch die Entität *CompositeFunctionalDependency*, welche als Container für funktionale Abhängigkeiten dient und in ihrem Aufbau dem Composite-Pattern [GHJV95] folgt. In dem Attribut *childrenList*

Abhängigkeiten können als Einheit betrachtet werden

wird dabei eine Liste aller Bestandteile des Kompositums verwaltet. Die Querbezüge zwischen den neu definierten Entitäten *InterServiceDependency* und *ServiceResourceDependency* und den bereits vorhandenen Entitäten *Service* und *Resource* werden im unteren Teil von Abbildung 6.8 dargestellt. Sowohl die *InterServiceDependendency* als auch die *ServiceResourceDependency* wurden als Assoziationsklassen modelliert und attributieren in dieser Funktion die Assoziation zwischen verschiedenen *Service* Entitäten bzw. zwischen *Service* und *Resource* Entitäten. Als Assoziationsrollen treten in beiden Fällen Bezugselement (*antecedent*) und Abhängiger (*dependent*) auf.

Es soll nocheinmal darauf hingewiesen werden, dass die vorliegende Abhängigkeitsmodellierung insbesondere auf eine Unterstützung der *Service Event Correlation* und *Impact Analyse* ausgerichtet ist (vgl. Incident Management in Abschnitt 5.3.2). Es wurde deshalb Wert darauf gelegt, eine strukturelle Ähnlichkeit zu den in [Han07] und [DR02] vorgestellten Modellen zu erzielen.

6.2.10 Basismodell der Service-MIB

Nachdem in den vorhergehenden Abschnitten die Teilkonzepte des Service-MIB Modells dargelegt wurden, erfolgt nun eine Zusammenfassung der definierten Entitäten in Form eines Basismodells. Der Fokus liegt dabei auf einer Darstellung der Zusammenhänge zwischen den verschiedenen Modellteilen – entsprechend wird in Abbildung 6.9 nur ein Ausschnitt aller definierten Entitäten dargestellt.

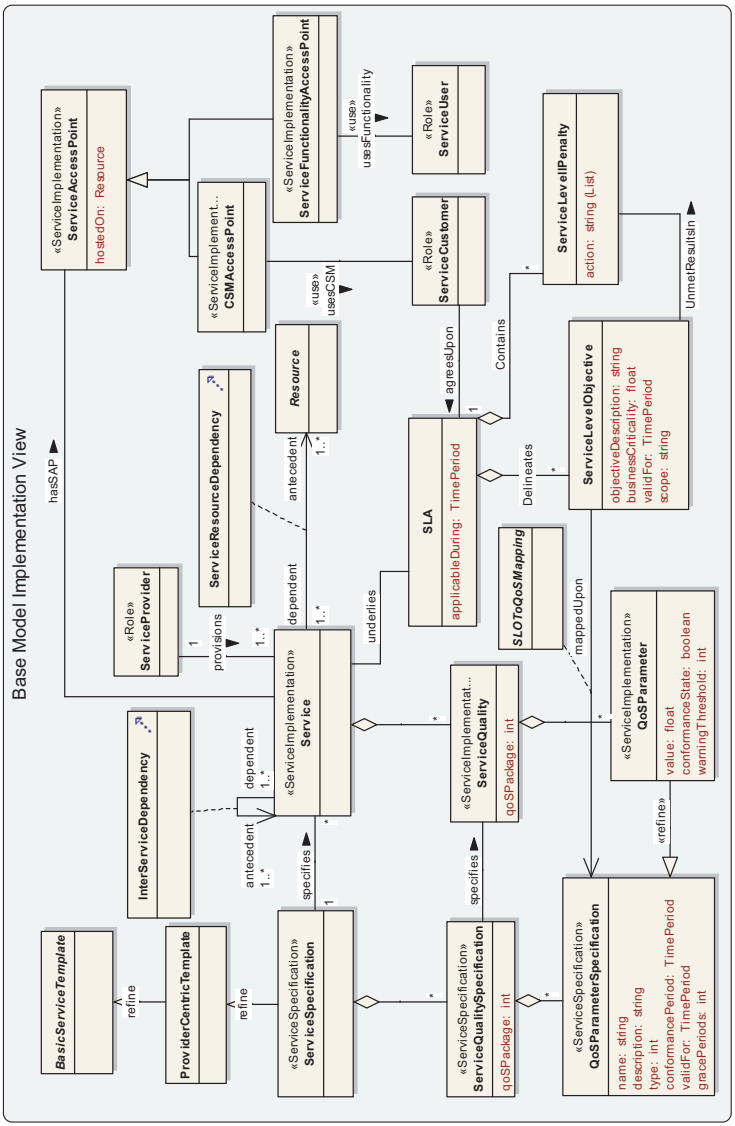

Abbildung 6.9: Basismodell der Service-MIB (Implementierungsteil)

6.3 Erweiterung des Basismodells mit Dienstattributen

Templates für Dienstattribute werden geschaffen

Das Fehlen konkreter Dienstattribute wurde in Abschnitt 3.4 als übergreifendes Defizit bestehender Managementmodelle hervorgehoben und gleichzeitig deren Wichtigkeit für eine Interoperabilität verschiedener Dienstmanagementanwendungen dargelegt. Dieser Aspekt wird nun im Folgenden mit der Durchführung des Teilschritts I der Methodik adressiert und somit eine Erweiterung des Basismodells der Service-MIB vorgenommen. In dieser Weise werden Templates für Dienstattribute geschaffen, die durch eine Anreicherung mit Aggregationsvorschriften (Abschnitt 6.5) auf ein bestimmtes Szenario adaptiert werden können. Entsprechend bestand dabei das Ziel, eine möglichst kompakte Basismenge von Dienstattributen zu schaffen, die somit weiter verfeinert und als Grundlage zur Berechnung weiterer Größen dienen kann.

Festlegung orientiert sich an der Analyse von Mgmt-Prozessen

Die Festlegung von Dienstattributen orientiert sich dabei vorwiegend an dem aufgrund der Analyse von Managementprozessen ermittelten Informationsbedarf (Abschnitt 5.3.4). Zusätzlich fliessen Ergebnisse aus einem Vorschlag des TeleManagement Forums zur Definition von Kennzahlen für *Wireless Services* mit ein [GB904b]. Es werden nun im Folgenden zunächst verschiedene Kategorien von Dienstattributen (z.B. Informationen zu Dienstfehlern) vorgestellt und danach entlang dieser Kategorien sukzessive die jeweiligen Dienstattribute definiert.

6.3.1 Kategorien von Dienstattributen

Erweiterbarkeit und Übersichtlichkeit stehen im Vordergrund

Um Dienstattribute in das Basismodell der Service-MIB einzubetten, müssen weitere Entitäten geschaffen und mit dem Dienst bzw. der Dienstspezifikation in Beziehung gesetzt werden. Ein wichtiges Ziel, das dabei verfolgt wird ist es, eine möglichst einfache Erweiterbarkeit mit zusätzlichen Attributen zu gewährleisten und gleichzeitig die Übersichtlichkeit des Service-MIB Modells zu wahren.

ServiceParameter repräsentieren Dienstattribute

Aus diesem Grund wurden Dienstattribute von der Entität Dienst entkoppelt und desweiteren in Kategorien bzw. Klassen von Attributen strukturiert. Dabei zeigte sich, dass eine Einteilung gemäß der OSI-Funktionsbereiche (FCAPS) eine klare Zuordnung zwischen Attributen und Kategorien ermöglicht. Abbildung 6.10 stellt diesen Sachverhalt

dar: Kategorien werden durch die Entitäten *ServiceFault, ServicePerformance, ServiceStatistics*, *ServiceConfiguration*, *ServiceSecurity* und *ServiceAccounting* repräsentiert und als Bestandteile des *Service* bzw. der *ServiceSpecification* modelliert (Kompositionsbeziehung). Sie bestehen jeweils wiederum aus *ServiceParameters*, die zur Beschreibung der eigentlichen Dienstattribute dienen. Gegenstand der in den folgenden Abschnitten durchgeführten Festlegung von Dienstattributen sind dabei insbesondere die folgenden Kategorien:

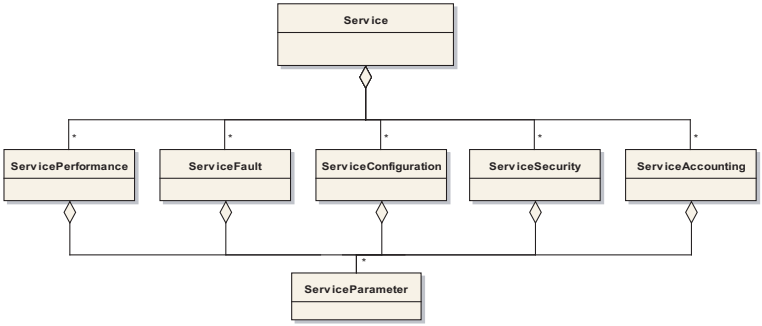

Abbildung 6.10: Kategorien für Diensteigenschaften

ServiceFault Attribute, die zur Erkennung von Dienstfehlern dienen, werden innerhalb dieser Kategorie zusammengefasst. Hinsichtlich der Analyse von Managementprozessen erscheinen in dieser Kategorie somit vorwiegend Attribute, die aus dem *Incident Management* Prozess abgeleitet wurden.

ServicePerformance Mit Attributen, die Auskunft über die Leistung des Dienstes geben, setzt sich die Kategorie *ServicePerformance* auseinander. In die exemplarische Festlegung von Attributen in den folgenden Kapiteln finden in diese Kategorie primär Attribute Einzug, die auf Basis des Service-Level Management Prozesses ermittelt wurden.

Für die restlichen Kategorien werden aufgrund der – in Abschnitt 5.3.1 begründeten – Einschränkung der Analyse von Managementprozessen

auf das *Incident* und *Service-Level Management* im weiteren Verlauf dieser Arbeit keine Attribute definiert. Sie sind vielmehr für zukünftige Versionen der Service-MIB vorgesehen und werden im Folgenden kurz erläutert:

ServiceConfiguration Innerhalb der Beschreibung des Basismodells wurden bereits eine Reihe von Entitäten vorgestellt, die als Teil der Konfigurationsbeschreibung eines Dienstes angesehen werden können (z.B. Abhängigkeiten als Beschreibung der Diensttopologie). Für alle darüberhinausgehenden Konfigurationseigenschaften des Dienstes wurde die Kategorie *ServiceConfiguration* geschaffen. Ihr Prozessbezug stellt vorwiegend das *Configuration* und *Change Management* dar.

ServiceSecurity Mit der Beschreibung von Sicherheitsaspekten des Dienstes betraute Attribute werden in dieser Kategorie zusammengefasst. In diesen Bereich fallen beispielsweise Statistiken zu Angriffsversuchen oder die Anzahl der gescheiterten Authentifizierungsversuche. Als Quelle für derartige Attribute kommt vor allem der *Security Management* Prozess in Frage.

ServiceAccounting Zur Abrechnung eines Dienstes dienende Attribute werden in der Kategorie *ServiceAccounting* aufgeführt. Dies kann beispielsweise die Festlegung von diskret abzählbaren *Abrechnungseinheiten* zur Beschreibung des in Anspruch genommenen Nutzungsvolumens des Dienstes [Rad03] umfassen. Vorgaben hierzu bietet der *Financial Management* Prozess.

Unterkategorisierung einfach möglich

Zusammenfassend kann festgestellt werden, dass durch die vorgenommene Modellierung eine grundlegende Strukturierung von Dienstattributen erreicht werden konnte. Gleichzeitig erscheint eine Erweiterung einfach durchführbar: Sollte mit der Hinzunahme weiterer Attribute der Bedarf nach Unterkategorien entstehen, so können diese als Unterklassen der jeweiligen Hauptkategorie in das Modell integriert werden.

6.3.2 Beschreibungsschema für Dienstattribute

Das zur Beschreibung von Dienstattributen verwendete Schema folgt der Ausrichtung des in diesem Kapitel vorgestellten Basismodells der Service-MIB: Hier liegt der Fokus auf der Erstellung eines Informationsmodells, das gemäß der in Abschnitt 2.1 beschriebenen Grundlagen auf der konzeptionellen Ebene angesiedelt ist. Entsprechend erfolgt die Beschreibung von Attributen in einer implementierungsneutralen Form – die Überführung in ein beliebiges Datenmodell soll damit so einfach wie möglich gestaltet werden. Dies zeigt sich vor allem in den Beschreibungselementen Datentyp und Werte, bei denen auf die Angabe konkreter und somit implementierungsabhängiger Datentypen bewusst verzichtet wurde. Im Folgenden werden die einzelnen Elemente des Schemas kurz vorgestellt:

Beschreibung erfolgt implementierungsneutral

Name Mit dem Namen wird ein eindeutiger, für den Anwender verständlicher, Bezeichner für das Dienstattribut festgelegt.

Beschreibung Die Angabe des Zwecks des Dienstattributs soll in diesem Feld in einer für den Anwender nachvollziehbaren textuellen Beschreibung erfolgen.

Datentyp Um die Abbildung des jeweiligen Dienstattributs in ein Datenmodell zu erleichtern, wird an dieser Stelle die Beschreibung eines generischen Datentyps für das entsprechende Attribut vorgenommen. Soll beispielsweise das Dienstattribut einen prozentualen Wert ausdrücken, so wird ihm der Datentyp *float* zugeordnet. Welche konkrete Ausprägung dieser Datentyp trägt (z.B. Genauigkeit), ist Teil der Implementierung und wird deshalb an dieser Stelle nicht weiter konkretisiert.

Werte Ebenfalls zur Vereinfachung der Implementierung eines Dienstattributs werden mit diesem Feld gültige Wertebereiche definiert. So kann beispielsweise festgelegt werden, dass ein Attribut zur Beschreibung der Verfügbarkeit Werte zwischen 0 und 1 annehmen kann.

Optional Mit diesem Feld wird angezeigt, ob die Implementierung des Dienstattributs zwingend erforderlich ist.

In welcher Weise Datentypen und Werte eines Attributs in ein konkretes Datenmodell überführt werden können, wird in Abschnitt 7.2 anhand von CIM demonstriert. Zunächst erfolgt jedoch die Festlegung von Attributen für die Kategorien *ServiceFault*, *ServicePerformance* und *ServiceStatistics*.

6.3.3 Attribute in der Kategorie Service Fault

Die schnellstmögliche Erkennung von Fehlerzuständen in der Diensterbringung stellt eine wichtige Herausforderung für IT-Provider dar, wie bereits in der Vorstellung des *Incident Management* Prozesses betont wurde. In diesem Abschnitt definierte Dienstattribute sind auf eine Unterstützung der Emittlung von Fehlerursachen (siehe Abschnitt 5.3.2) ausgelegt. Wie aus den in Tabelle 5.3 dargestellten Ergebnissen der Informationsanalyse hervorgeht, müssen dabei drei verschiedene Aspekte beachtet werden:

- Holistische Betrachtung des Dienstes
- Fehler in dienstrealisierenden Komponenten
- Informationen zur Berichtserstellung (*Reporting*)

Dieser Aufteilung folgend werden nun zunächst Attribute vorgestellt, denen eine ganzheitliche Betrachtung des Dienstes zu Grunde liegt (Tabelle 6.17). An erster Stelle steht dabei das, den akutellen Zustand des Dienstes reflektierende Attribut *operationalStatus*. Es zeigt an, ob die Funktionalität des Dienstes momentan in vollem Umfang zur Verfügung steht oder, im Falle von Störungen, in welchem Maße die Funktionalität eingeschränkt ist (*Degradierung*).

Zur Beurteilung des Ausmaßes an Dienstfehlern dienen die Attribute *FaultyTransactions* und *FaultyTransactionRate*. Um eine Bezugsgröße für funktionale Bestandteile des Dienstes zu schaffen, wurde auf die in Abschnitt 6.2.3 entwickelten Konzepte zurückgegriffen: Die Abstraktion

der Dienstfunktionalität in *Transactions* ermöglicht eine generische Beschreibung funktionaler Einheiten des Dienstes. Entsprechend beziehen sich die genannten Attribute auf die Anzahl fehlerhafter Transaktionen bzw. deren Anteil an der Gesamtmenge von Transaktionen. Zur Vervollständigung dieser Betrachtung wurde ferner das Attribut *ServiceTransactionSuccessRate* definiert: Es beschreibt das Verhältnis von erfolgreich durchgeführten Transaktionen zu der Gesamtzahl an Transaktionen.

Ebenfalls aufgenommen wurden in Tabelle 6.17 Attribute, die Aussagen zu Fehlern zulassen, die während eines Verbindungsversuchs des Nutzers auftraten. Während das Attribut *serviceAttachFaults* deren Gesamtzahl reflektiert, repräsentiert das Attribut *serviceAttachFaultsRate* wiederum die relative Häufigkeit derartiger Fehler. Das letzte Attribut in dieser Tabelle *serviceConnectionFailureRate* gibt schließlich Aufschluss darüber, welcher Anteil der Verbindungen mit dem Dienst nach einem erfolgreichen Verbindungsaufbau abgebrochen sind.

Attribute für die Kategorie ServiceFault

Name	Beschreibung	Datentyp	Werte	Optional
Operational Status	Zeigt an, ob der Dienst momentan ordnungsgemäß funktioniert. Werte kleiner als 1 weisen dabei auf eine Degradierung der Dienstfunktionalität hin.	float	$0 \leq OS \leq 1$	nein 0
Faulty Transactions	Die Gesamtanzahl fehlerhafter Transaktionen des Dienstes	Integer	≥ 0	nein 1
Faulty Transaction Rate	Das Verhältnis fehlerhafter Transaktionen zu der gesamten Anzahl von Transaktionen des Dienstes	float	$0 \leq FTR \leq 1$	ja 2

Attribute für die Kategorie ServiceFault (*Fortsetzung)*				
Name	**Beschreibung**	**Datentyp**	**Werte**	**Optional**
Service Transaction Success Rate	Stellt den Anteil erfolgreicher Transaktionen von und zu dem Dienst im Verhältnis zu deren Gesamtzahl dar	Float	$0 \leq TSR \leq 1$	nein 3
Service Attach Faults	Repräsentiert die Anzahl fehlgeschlagener Verbindungsaufbauversuche zum Dienst bzw. SAP.	Integer	≥ 0	nein 4
Service Attach Fault Rate	Stellt das Verhältnis von fehlgeschlagenen zu erfolgreichen Verbindungsaufbauversuchen zum Dienst bzw. SAP dar.	Float	$0 \leq AFR \leq 1$	nein 5
Service Connection Failure Rate	Beschreibt den Anteil der Verbindungen mit dem Dienst, die nach einem erfolgreichen Verbindungsaufbau abgebrochen sind	float	$0 \leq CFR \leq 1$	nein 6

Tabelle 6.17: Attribute zur ganzheitlichen Betrachtung fehlerbezogener Aspekte des Dienstes

Dem zweiten Aspekt (Fehler in dienstrealisierenden Komponenten) zugehörige Attribute dienen vor allem einer schnellen Ermittlung der Störungsursache innerhalb der Teilaktivität *Investigation and Diagnosis* des *Incident Management* Prozesses. Hierbei liegt die Feststellung zu Grunde, dass Dienstfehler oftmals als Resultat von Fehlern in den

dienstrealisierenden Komponenten auftreten und somit durch Nachverfolgung dieser Zusammenhänge ausfindig gemacht werden können. Entsprechend konzentrieren sich die in Tabelle 6.18 definierten Attribute darauf, dem für die Störungsfindung verantwortlichen Personal einen Überblick über den Zustand der dienstrealisierenden Komponenten zu vermitteln und sie in die Lage zu versetzen, möglichst schnell die Ursache des Fehlers zu identifizieren. Aus diesem Grund werden auch Attribute berücksichtigt, die auf den ersten Blick nicht unmittelbar der Kategorie *ServiceFault* zugehörig erscheinen, aber für die genannte Zielsetzung benötigt werden (die Zuordnung eines Attributs zu mehreren Kategorien wird gemäß Abschnitt 6.3.1 explizit unterstützt).

Das erste Attribut in Tabelle 6.18 (*SEHighestErrorRate*) enthält den Namen der dienstrealisierenden Komponente, die gegenwärtig die höchste Fehlerrate aufweist und gibt somit einen Hinweis auf den möglichen Verursacher der Dienststörung. Der Wert des Attributs reflektiert dabei einen Verweis auf das diese Komponente beschreibende Managementobjekt, wie durch die Wertemenge *{ref}* angedeutet. Neben der Fehlerrate wurde in diesem Zusammenhang weiterhin die Auslastung von Dienstelementen in die Betrachtung aufgenommen, da sie auf Leistungsengpässe in der Diensterbringung hinweisen kann. Entsprechend zeigt das Attribut *SEHighestUtilization* das Dienstelement mit der höchsten Auslastung an. Weiterhin ist es für die Erkennung der Störungsursache oftmals nützlich, die durchschnittliche Fehlerrate bzw. Auslastung aller Dienstelemente zu betrachten (siehe Tabelle 5.3 aus der Informationsanalyse), was durch die Attribute *SEAverageErrorRate* und *SEAverageUtilization* ermöglicht wird.

Bei vielen Störungen des Dienstes tritt der Fall auf, dass die einzelnen Dienstelemente für sich genommen zwar fehlerfrei funktionieren, aufgrund von Fehlern in den (Netz-)Verbindungen allerdings in ihrer Kommunikation eingeschränkt sind. Um diesem Umstand Rechnung zu tragen, wurden die verbleibenden drei Attribute in Tabelle 6.18 eingeführt: Während der *InterSEConnectivityStatus* beschreibt, ob sich die (Netz-)Verbindungen zwischen Dienstelementen gegenwärtig in einem funktionsfähigen Zustand befinden, werden Fehlerrate und Auslastung dieser Verbindungen durch die Attribute *InterSETransferErrors* bzw. *InterSE LinkUtilization* repräsentiert.

Attribute für die Kategorie ServiceFault				
Name	**Beschreibung**	**Datentyp**	**Werte**	**Optional**
SE HighestErrorRate	Beinhaltet den Namen des Dienstelements, das gegenwärtig die höchste Fehlerrate aufweist	String	{*ref*}	nein 7
SE AverageErrorRate	Dieses Attribut beschreibt die durchschnittliche Fehlerrate aller Dienstelemente	Float	$0 \leq AER \leq 1$	nein 8
SE Highest Utilization	Verweist auf das Dienstelement mit der gegenwärtig höchsten Auslastung	String	{*ref*}	nein 9
SE Average Utilization	Enthält die durchschnittliche Auslastung aller Dienstelemente	Float	$0 \leq AER \leq 1$	nein 10
Inter SE Connectivity Status	Zeigt an, ob sich alle (Netz-)Verbindungen zwischen Dienstelementen in funktionsfähigem Zustand befinden. Werte unter 1 deuten auf eine Degradierung hin.	Float	$0 \leq ICS \leq 1$	nein 11
Inter SE Transfer Errors	Dieses Attribut beschreibt die Anzahl von Fehlern in den (Netz-)Verbindungen zwischen Dienstelementen.	Integer	≥ 0	nein 12

Attribute für die Kategorie ServiceFault (*Fortsetzung)*				
Name	**Beschreibung**	**Datentyp**	**Werte**	**Optional**
Inter SE Link Utilization	Der Wert dieses Attributs entspricht der gegenwärtigen Auslastung der (Netz-) Verbindung zwischen Dienstelementen.	Float	$0 \leq ILU \leq 1$	nein 13

Tabelle 6.18: Attribute im Kontext von Fehlern in dienstrealisierenden Komponenten

Abschließend zu der Kategorie *ServiceFaults* werden nun Attribute betrachtet, die zur Berichterstellung gegenüber dem Dienstkunden genutzt werden können. Die Notwendigkeit für derartige Informationen geht insbesondere auf die Teilaktivität *Ownership, monitoring, tracking and communication* des *Incident Management* Prozesses zurück und wurde entsprechend als Informationsanforderung in Tabelle 5.3 aufgenommen.

Im Hinblick auf eine mögliche Bewertung der Effektivität und Effizienz der Störungsbehandlung stellen die Attribute in Tabelle 6.19 einen Bezug zwischen dem Dienst und des diesen Dienst betreffenden *Incident Records* her. Ausgehend von dem Attribut *NumberOfIncident Records*, das die Anzahl der Incident Records in einem bestimmten Zeitraum enthält, können so weitere Statistiken zu dem Verlauf der Bearbeitung dieser Records erstellt werden. Maßgeblich ist hierbei zunächst der Anteil an Incidents für einen Dienst, der nicht innerhalb des vereinbarten Zeitraums gelöst werden konnte *(IncidentsFixedinResolutionTime)*, da er zur Berechnung der Effektivität des Prozesses herangezogen werden kann. Um Aussagen über die Effizienz der Störungsbehandlung zu ermöglichen, wurde weiterhin der Anteil an Incident Records, die erneut geöffnet wurden, mit dem Attribut *RepeatedIncidents* dokumentiert. Einen Zusammenhang zwischen der Geamtzahl an geschlossenen Incident Records und den Records, bei denen sich kein Problem feststellen ließ, zeigt schließlich das Attribut *IncidentsWithNoProblemFound*.

Attribute für die Kategorie ServiceFault				
Name	**Beschreibung**	**Datentyp**	**Werte**	**Optional**
NumberOf Incident Records	Anzahl von Incident Records, die für den Dienst während eines bestimmten Zeitintervalls geöffnet wurden	Integer	$\geq$0	nein 14
Incidents Fixedin Resolution Time	Anteil der Incident Records, die nicht innerhalb der anvisierten Zeit geschlossen werden konnten	Float	0$\leq FRT \leq$1	nein 15
Repeated Incidents	Anteil der Incident Records, die erneut geöffnet wurden	Float	0 $\leq RI \leq$1	nein 16
Incident WithNo Problem Found	Anteil der Incident Records, die geschlossen wurden und bei denen kein Problem identifiziert werden konnte	Float	0$\leq INP \leq$1	nein 17

Tabelle 6.19: Attribute zur Berichterstellung gegenüber Dienstkunden im Bereich Incident Management

Wegen der verhältnismäßig hohen Anzahl denkbarer Attribute in diesem Bereich wurde in Tabelle 6.19 nur ein Exzerpt an Basisgrößen vorgestellt. Eine dienstkunden- und szenariospezifische Erweiterung wird als Teil der Nutzungsphase einer Service-MIB angesehen und in Abschnitt 7.2 weiterführend behandelt.

6.3.4 Attribute in der Kategorie ServicePerformance

In diesem Abschnitt definierte Attribute zielen auf eine Beschreibung leistungsbezogener Aspekte eines Dienstes ab. Sie stehen in Bezug zu der Informationsanalyse des Service-Level Management Prozesses (Abschnitt 5.3.3) und dienen in diesem Zusammenhang insbesondere zur Unterstützung der Teilaktivität *Monitoring and Reporting*.

Entsprechend der kumulierten Informationsanforderungen in Tabelle 5.3 wird die schon in der Kategorie *ServiceFault* eingeführte Aufteilung beibehalten: Zunächst werden die, die ganzheitliche Leistung des Dienstes betrachtenden Attribute eingeführt, danach folgen Attribute, die Auskunft über die Performanz der dienstrealisierenden Komponenten vermitteln. Abschließend werden wiederum Basisgrößen zur Berichterstellung innerhalb des Service-Level Management Prozesses in geeignete Attributsdefinitionen überführt.

Als generische Bezugsgrößen zur Darstellung des Leistungsverhaltens eines Dienstes finden zunächst wiederum als Teil der *ServiceFunctionality* modellierte Diensttransaktionen Verwendung. Während das an erster Stelle in Tabelle 6.20 stehende Attribut *ServiceTransaction* die Anzahl aller gegenwärtig in Bearbeitung befindlicher Transaktionen repräsentiert, stellt die *AverageServiceTransaction* deren durchschnittlichen Wert dar. Als weitere auf Diensttransaktionen bezogene Größe beschreibt das Attribut *ServiceTransactionSpeed* die aktuell gemessene Ausführungszeit für eine Transaktion, eine durchschnittliche Betrachtung von Ausführungszeiten findet sich wiederum in dem Attribut *AverageServiceTransactionSpeed*.

Auskunft über die Anzahl an den Dienst gestellter Anfragen geben die nächsten beiden Attribute in Tabelle 6.20. Mit Hilfe dieser Werte kann eine genauere Leistungsabschätzung erzielt werden – vor allem für Dienste, bei denen die Anzahl von Anfragen pro Transaktion stark variiert. Neben dem, die gegenwärtige Anzahl an Anfragen wiedergebenden Attribut *ServiceRequests* wurde für diesen Zweck das Attribut *AverageServiceRequests* vorgesehen, das den durchschnittlichen Wert aller Anfragen enthält.

Eine vor allem im Hinblick auf vertragliche Vereinbarungen zwischen Dienstkunde und Provider wichtige Kennzahl stellt die durchschnittli-

che Verfügbarkeit eines Dienstes dar. Sie kann als zeitliches Korrelat des in der Kategorie *ServiceFault* eingeführten Attributs *OperationalStatus* verstanden werden (siehe Tabelle 6.17) und wird durch das Attribut *ServiceAvailability* repräsentiert. Weiterhin von Belang für die Überwachung der Leistung eines Dienstes ist die von und zu dem Dienst übertragene Datenmenge. Hier sind es wiederum der aktuelle (Attribut *ServiceDataTransfer*) bzw. durchschnittliche Wert (Attribut *AverageServiceDataTransfer*) dieser Volumengröße, die in der Service-MIB Berücksichtigung finden.

Attribute für die Kategorie ServicePerformance

Name	Beschreibung	Datentyp	Werte	Optional
Service Transactions	Enthält die aktuell in Bearbeitung befindliche Anzahl an Diensttransaktionen	Integer	≥0	nein 18
Average Service Transactions	Repräsentiert den Durchschnittswert an in Bearbeitung befindlicher Transaktionen	Integer	≥0	nein 19
Service Transaction Speed	Beschreibt die gegenwärtig benötigte Zeitspanne für die Fertigstellung einer Transaktion mit dem Dienst in Millisekunden	ms	≥0	nein 20
Average Service Transaction Speed	Enthält die durchschnittliche Bearbeitungsdauer für eine Transaktion mit dem Dienst in Millisekunden	ms	≥0	nein 21
Service Requests	Beschreibt die aktuell auftretende Anzahl an Dienstanfragen	Integer	≥0	nein 22

Attribute für die Kategorie ServicePerformance (*Fortsetzung)*				
Name	**Beschreibung**	**Datentyp**	**Werte**	**Optional**
Average Service Requests	Enthält die durchschnittliche Anzahl an Dienstanfragen	Integer	≥ 0	nein 23
Service Availability	Beschreibt die Verfügbarkeit des Dienstes über einen bestimmten Zeitraum	Float	$0 \leq INP \leq 1$	nein 24
Service Data Transfer	Stellt das aktuell gemessene Volumen an Ende-zu-Ende übertragenen Daten von und zu dem Dienst in Bytes dar	Bytes	≥ 0	nein 25
Average Service Data Transfer	Repräsentiert das durchschnittliche Volumen Ende-zu-Ende übertragener Daten von und zu dem Dienst in Bytes	Bytes	≥ 0	nein 26

Tabelle 6.20: Attribute zur ganzheitlichen Betrachtung von Leistungsaspekten des Dienstes

In den, aus Übersichtlichkeitsgründen in einer separaten Tabelle präsentierten, zweiten Teil der Darstellung der Attribute zur ganzheitlichen Betrachtung des Dienstes gehen zunächst Leistungsgrößen bezüglich des Verbindungsaufbaus mit dem Dienst ein (siehe Tabelle 6.21). Während das Attribut *ServiceAttachTime* die aktuell gemessene Zeitspanne für einen Verbindungsaufbau zum Service Access Point angibt, findet sich die Durchschnittsbetrachtung dieses Wertes in der *AverageServiceAttachTime* wieder. Im selbigen Kontext geben die Attribute

ServiceAttachCount und *ServiceAttachSuccessRate* Aufschluss über die Gesamtanzahl von Versuchen, eine Dienstverbindung aufzubauen, bzw. über deren Erfolgsquote.

Weiterhin wird mit dem Attribut *EndtoEndServiceEstablishmentTime* die Gesamtzeit für einen Ende-zu-Ende Verbindungsaufbau mit dem Dienst betrachtet. Dem bisherigen Schema folgend, findet sich ferner die Durchschnittsbetrachtung dieses Wertes in dem Attribut *AverageEndtoEndServiceEstablishmentTime* wieder.

Attribute für die Kategorie ServicePerformance

Name	Beschreibung	Datentyp	Werte	Optional
Service Attach Time	Beschreibt die aktuell benötigte Zeitspanne für den Verbindungsaufbau zum SAP in Millisekunden	ms	$\geq$0	nein 27
Average Service Attach Time	Enthält die durchschnittliche Zeitspanne für den Verbindungsaufbau zum SAP in Millisekunden	ms	$\geq$0	ja 28
Service Attach Count	Stellt die Anzahl benötigter Versuche zum Aufbau einer Dienstverbindung dar	Integer	$\geq$0	nein 29
Service Attach Success Rate	Beschreibt die Erfolgsrate für den Aufbau von Dienstverbindungen	Float	$0 \leq ASR \leq$	ja 30
End to End Service Establishment Time	Repräsentiert die vergangene Zeitspanne für den Ende-zu-Ende Verbindungsaufbau zum Dienst in Millisekunden	ms	$\geq$0	nein 31

Attribute für die Kategorie ServicePerformance (*Fortsetzung)*				
Name	**Beschreibung**	**Datentyp**	**Werte**	**Optional**
Average End to End Service Establishment Time	Beschreibt die durchschnittliche Zeitspanne für den Ende-zu-Ende Verbindungsaufbau zum Dienst in Millisekunden	ms	≥ 0	nein 32

Tabelle 6.21: Attribute zur ganzheitlichen Betrachtung von Leistungsaspekten des Dienstes (Fortsetzung)

In dem nächsten Bereich der Kategorie *ServicePerformance* stehen Attribute im Mittelpunkt, die eine Überwachung der Leistung in Bezug auf die dienstrealisierenden Komponenten ermöglichen (Tabelle 6.22). Wenngleich diese Informationen typischerweise nicht unmittelbar in SLAs miteinfließen, ermöglichen sie doch eine Analyse und Verbesserung der Dienstbereitstellung (z.B. im Rahmen des *ITIL Capacity Managements*) und wurden deshalb innerhalb der Service-MIB berücksichtigt.

Beginnend mit der *SEAvailability* und *SELowestAvailability* werden hierbei zunächst die durchschnittliche Verfügbarkeit aller Dienstelemente bzw. das Dienstelement mit dem niedrigsten Verfügbarkeitswert durch Attributsdefinitionen repräsentiert. Insbesondere letzteres Attribut liefert dabei Hinweise auf mögliche Schwachstellen der Diensterbringungskette. An nächster Stelle stehen Attribute, die den Durchsatz der Dienstelemente beschreiben: Während *SEThroughput* Auskunft über den kumulierten Durchsatz aller Dienstelemente gibt, kann über das Attribut *SEAverageThroughput* auf dessen Durchschnittswert zugegriffen werden.

Weiterhin finden Leistungsgrößen bzgl. der Kommunikation zwischen den Dienstelementen Berücksichtigung. Dies umfasst die Geschwindigkeit und das Volumen der Datenübertragung, die in entsprechende Attributsdefinitionen überführt wurden (*InterSEDataTransferSpeed* und

InterSEDataTransfer). Darüber hinaus wurden mit *AverageInterSEDataSpeed* und *AverageInterSEDataTransfer* Attribute geschaffen, die eine Durchschnittsbetrachtung der genannten Größen ermöglichen.

Attribute für die Kategorie ServicePerformance

Name	Beschreibung	Datentyp	Werte	Optional
SE Availability	Stellt die kumulierte Verfügbarkeit aller Dienstelemente dar	Float	$0 \leq ASR \leq$	nein 33
SE Lowest Availability	Verweist auf das Dienstelement mit dem schlechtesten Verfügbarkeitswert	String	{*ref*}	nein 34
SE Throughput	Repräsentiert den kumulierten Durchsatz aller Dienstelemente in Bytes-per-Sekunde	bps	≥ 0	nein 35
SE Average Throughput	Enthält den Durchschnittswert für kumulierten Durchsatz aller Dienstelemente in Bytes-per-Sekunde	bps	≥ 0	nein 36
Inter SE Data Transfer Speed	Enthält die aktuelle Geschwindigkeit der Datenübertragung zwischen Dienstelementen in Bytes-per-Sekunde	bps	≥ 0	nein 37
Average Inter SE Data Speed	Beschreibt die durchschnittliche Geschwindigkeit der Datenübertragung zwischen Dienstelementen in Bytes-per-Sekunde	Byte	≥ 0	nein 38

Attribute für die Kategorie ServicePerformance (*Fortsetzung*)				
Name	**Beschreibung**	**Datentyp**	**Werte**	**Optional**
Inter SE Data Transfer	Beschreibt die Gesamtmenge an zum gegenwärtigen Zeitpunkt übertragenen Daten zwischen Dienstelementen in Bytes	Byte	≥ 0	nein 39
Average Inter SE Data Transfer	Stellt die durchschnittliche Menge an übertragenen Daten zwischen Dienstelementen in Bytes dar	Byte	≥ 0	nein 40

Tabelle 6.22: Attribute zur Beschreibung von Leistungsgrößen dienstrealisierender Komponenten

Der verbleibende Bereich in der Kategorie *ServicePerformance* beschäftigt sich mit Attributen, die Aussagen zu dem Erfüllungsgrad in einem SLA vereinbarter Vorgaben (vgl. Entität *ServiceLevelObjective* in Abschnitt 6.2.6) ermöglichen. Derartige Managementinformationen dienen insbesondere zur Berichterstellung gegenüber dem Dienstkunden und sind in der Teilaktivität *Monitoring and Reporting* des Service-Level Management Prozesses (siehe auch entprechende Informationsanforderungen in Tabelle 5.3) angesiedelt. Analog zu Abschnitt 6.3.3 werden an dieser Stelle nur die wichtigsten Basisgrößen definiert (siehe Tabelle 6.23), Möglichkeiten zu deren Erweiterung werden innerhalb der Nutzungsphase (Abschnitt 7.2) diskutiert.

Um zu entscheiden, ob die im Service-Level Management definierten Zielvorgaben eingehalten wurden, gilt es, neben dem SLA auch die mit externen Partnern (*Underpinning Contracts*) und internen Abteilungen (*Operational Agreements*) getroffenen Vereinbarungen zu berücksichtigen (siehe auch Abschnitt 5.3.3). Entsprechend wurden zusätzlich zu dem Attribut *ObjectivesMet*, das den Anteil der erfüllten

SLA-Zielvorgaben repräsentiert, die Attribute *OLAObjectivesMet* und *UCObjectivesMet* eingeführt, die dem gleichen Zwecke im Kontext von *Underpinning Contracts* bzw. *Operational Agreements* dienen. Weiterhin wurde mit dem Attribut *SeverityOfUnmetObjectives* eine Möglichkeit zur Beschreibung des Schweregrades von SLA-Verletzungen geschaffen. Als mögliche Werteskala dienen hierbei die Zahlen 1 bis 5, wobei der höhere Wert eine relativ schwerwiegendere Verletzung anzeigt. Abschließend wurde in Tabelle 6.23 das Attribut *ResultingPenalties* definiert. Es ist dafür destiniert, den monetären Schaden, der aufgrund einer Nichteinhaltung von SLA-Zielen entstand, kumuliert zu erfassen.

Attribute für die Kategorie ServicePerformance

Name	Beschreibung	Datentyp	Werte	Optional
Objectives Met	Stellt den Anteil der in einem Service Level Agreement enthaltenen Zielvorgaben die erreicht wurden dar	Float	$0 \leq OM \leq 1$	nein 41
OLA Objectives Met	Beschreibt den Anteil der in einem Operational Level Agreement enthaltenen Zielvorgaben die erreicht wurden	Float	$0 \leq OOM \leq 1$	ja 42
UC Objectives Met	Repräsentiert den Anteil der in einem Underpinning Contract enthaltenen Zielvorgaben die erreicht wurden	Float	$0 \leq UOM \leq 1$	ja 43
Severity Of Unmet Objectives	Beschreibt den relativen Schweregrad von SLA Verstößen	Integer	$1 \leq SUO \leq 5$	nein 44

Attribute für die Kategorie ServicePerformance (*Fortsetzung*)				
Name	**Beschreibung**	**Datentyp**	**Werte**	**Optional**
Resulting Penalties	Stellt die aus der Nicht-einhaltung des SLAs resultierenden Pönalen während eines bestimmten Zeitraumes dar	Integer	≥ 0	ja 45

Tabelle 6.23: Attribute zur Berichterstellung gegenüber Dienstkunden im Bereich Service-Level Management

Mit der Definition von Dienstattributen in den Kategorien *ServiceFault* und *ServicePerformance* wurde konkrete Managementinformation geschaffen und gleichzeitig eine Erweiterung des Service-MIB Basismodells vorgenommen. Der zweite Teil der Spezifikationsphase widmet sich nun dem Aufzeigen des Zusammenhangs zwischen Dienstattributen und Parametern dienstrealisierender Komponenten. Insbesondere werden Konzepte vorgestellt, die eine Errechnung der Attributswerte durch Aggregation von Komponentenparametern ermöglichen. Diese sind für die Abbildung von QoD auf QoS von Bedeutung.

6.4 Entwicklung einer Spezifikationssprache

Sprachentwurf zur Realisierung der Synthesemethodik

Wie insbesondere anhand des GRID Szenarios in Abschnitt 2.2.2 verdeutlicht wurde, besteht eine elementare Herausforderung im Service Management darin, komponentenorientierte Managementinformationen zu aggregieren bzw. zu verdichten, um somit Aussagen über den Zustand des von diesen Komponenten realisierten Dienstes treffen zu können. In diesem Abschnitt werden nun Sprachkonzepte zur Beschreibung der genannten Zusammenhänge entwickelt und somit die Grundlagen zur Realisierung der in Kapitel 4 vorgestellten Synthesemethodik erarbeitet.

Betrachtet man die Vielzahl an Möglichkeiten einen gegebenen Dienst zu realisieren, wird deutlich, dass die Art und Weise, in der Komponenteninformationen verdichtet werden müssen, von der jeweiligen Implementierung abhängig und so zu einem gewissen Maß *dienstspezifisch* ist. Erschwerend kommt weiterhin hinzu, dass Komponenteninformationen oftmals in unterschiedlichen Formaten vorliegen, in verschiedenen Datentypen gehalten und zudem mit verschiedenen Abtastraten gemessen werden.

Erforderlich wird also an dieser Stelle eine flexible Spezifikationssprache, die es erlaubt, Aggregationsbeziehungen zwischen Komponenten- und Dienstinformation innerhalb der Service-MIB zu spezifizieren. Wie sich in Kapitel 3 zeigte, wurden allerdings bisher nur domänen- und technologiespezifische Ansätze vorgestellt, die nicht zur Erfüllung dieser Zielsetzung herangezogen werden können: Zwar bietet WSLA (Abschnitt 3.2.3) Möglichkeiten zur Beschreibung von Aggregationsbeziehungen, zielt aber auf die Spezifikation von SLA-Parametern ab. Die Beschreibung von Diensten erfolgt deshalb als Teil des zentralen Sprachelements SLA, was nicht dem Verständnis dieser Arbeit bzw. dem Service-MIB Modell entspricht. Die Expression-MIB (Abschnitt 3.1.3) hingegen erlaubt die Aggregation von Internet-MIB Parametern, kann aber nur unter Benutzung dieser Managementarchitektur verwendet werden.

Aus diesem Grund konnte keine Erweiterung bestehender Arbeiten vorgenommen werden, so dass an dieser Stelle ein neuer Sprachentwurf, die *Service Information Specification Language* (SISL), notwendig erschien. Dabei wurden folgende, mit den in Kapitel 2 ermittelten Anforderungen korrespondierende Ziele verfolgt:

Aggregationen sollen Diensteigenschaften reflektieren

I. *Deklarativität*
Zunächst sollte es ermöglicht werden, Eigenschaften eines Dienstes (AGG 1) als Aggregation von Komponentenparametern einfach und nachvollziehbar zu beschreiben, ohne dabei auf den konkreten Ablauf der Berechnung eingehen zu müssen (AGG 2). Einerseits kann somit die Übersichtlichkeit bzw. Lesbarkeit der Spezifikation erhöht und andererseits eine Unabhängigkeit von bestimmten Implementierungstechnologien zur weiteren Verarbeitung der Aggregationen erzielt werden. Genannte Aspekte führen zur Wahl

eines deklarativen Spezifikationskonzepts, mit dessen Hilfe vorrangig das erwünschte Ergebnis der Aggregation bzw. die resultierende Eigenschaft des Dienstes beschrieben werden kann.

II. *Verweise auf dienstrelevante Komponentenparameter*
Komponentenparameter als Verweise
Wie bereits erwähnt, werden komponentenorientierte Managementinformationen nicht als Teil einer Service-MIB betrachtet. Entsprechend muss die Spezifikationssprache Verweise auf für den Dienst relevante Komponentenparameter bzw. auf die Datenquellen, die diese bereitstellen, erlauben. Somit soll insbesondere der dritte Teilschritt der Synthesemethodik realisiert werden. Um ferner eine Steuerung der nachfolgenden Überwachungsphase zu gewährleisten, müssen zusätzlich Abtastraten für den Messvorgang und zu generierende Datentypen in der Spezifikation berücksichtigt werden (Teilschritt IV der Synthesemethodik).

III. *Aggregationsoperationen*
Bibliothek von Aggregationsoperationen
Das Kernstück einer Aggregation bilden mathematische und logische Operationen, die auf den referenzierten Komponentenparametern durchgeführt werden. Um eine möglichst flexible Spezifikation zu unterstützen, müssen elementare Funktionen zur Verfügung gestellt werden und gleichzeitig muss die Möglichkeit zur benutzerspezifischen Erweiterung gegeben sein. Nicht zuletzt sollen durch diesen Teil des Sprachentwurfs die Voraussetzungen zur Umsetzung des fünften Teilschrittes der Synthesemethodik geschaffen werden.

IV. *Schwellwerte und Benachrichtigungsoptionen*
Unter der Prämisse, dass komponentenorientierte Managementinformationen außerhalb der Service-MIB verwaltet und von externen Datenquellen bezogen werden müssen, besteht ein weiteres Ziel im Entwurf der Spezifikationssprache darin, Konstrukte zur Steuerung dieses Vorgangs bzw. der Überwachungsphase bereitzustellen (AGG 4). Dies umfasst Abtastraten für die Messung von Komponentenparametern sowie Schwellwerte, die anzeigen, wann Benachrichtigungen versandt werden sollen.

Die Umsetzung der genannten Ziele erfolgte durch die Entwicklung einer formalen Sprache, der *Service Information Specification Language*

SISL erlaubt Spezifikation von Abbildungsvorschriften

(SISL). Sie wurde einführend in [Lan06] vorgestellt und in [DSgF07] mit Erweiterungen versehen. SISL zielt auf eine universelle Einsetzbarkeit ab und beschränkt sich dabei insbesondere nicht auf bestimmte Arten von Diensten, Komponenten oder Implementierungstechnologien. Um eine einfache Integration in bestehende Managementanwendungen zu gewährleisten, wurde SISL zudem als XML-basierte Sprache entwickelt.

Vereinfachte Notation wird gewählt

Im Folgenden werden nun die wichtigsten syntaktischen Elemente von SISL vorgestellt und deren Verwendungszweck erläutert. Um schwer lesbare XML-Ausschnitte zu vermeiden, wird die Syntax der Sprache in einer vereinfachten, aber dennoch zur XML-Grammatik äquivalenten Notation dargestellt: Ausdrücke in Klammern repräsentieren dabei die Inhalte eines XML-Elements, Zeichen vor der öffnenden Klammer den Namen dieses Elements. Attribute werden ihrerseits in einer "Name=Wert" Notation am Anfang des Inhalts eines Elements ausgedrückt. Ferner erfolgt die Präsentation der Produktionen der Sprache in der *Erweiterten Backus Naur Form (EBNF)*. In dieser Weise kann eine kompakte Darstellung von SISL erzielt werden – das vollständige XML-Schmea findet sich in Anhang B.

6.4.1 Grundlegende Sprachkonzepte

Aggregation als Teil eines Dienstattributs

Das zentrale Element in SISL bildet das `serviceAtribute`. Es enthält, neben einem Bezeichner und einer Beschreibung, vor allem die zur Abbildung von Komponentenparametern essentiellen Aggregationen. Letztere werden durch das Element `aggregation` repräsentiert und umfassen – entsprechend der in Abschnitt 6.4 dargelegten Zielsetzung – relevante Komponentenparameter, Abbildungsvorschriften für diese Parameter und Benachrichtigungsoperationen. Abbildung 6.11 veranschaulicht die Grundstruktur einer SISL-Aggregation: Sie besteht aus `resource`-Deklarationen (siehe Abschnitt 6.4.2), `function(s)` (siehe 6.4.3), einer `notification` (siehe 6.4.4) und einer `description`.

Typen und Wertemengen

SISL wurde als typisierte Sprache konzipiert und unterstützt ganzzahlige Werte (*integer*) und Fließkommazahlen (*float*), Zeichenketten

⟨serviceAttribute⟩::= "*serviceAttribute{*" ⟨aggregation⟩ ⟨identifier⟩ ⟨description⟩ "*}*"

⟨aggregation⟩::= "*aggregation{*" ⟨identifier⟩ ⟨description⟩ ⟨resource⟩ ⟨function⟩ ⟨notification⟩ "*}*"

⟨description⟩::= "*description{*" [⟨author⟩] [⟨date⟩] [⟨text⟩] "*}*"

⟨identifier⟩::= "*id* = " ⟨id⟩

⟨id⟩::= ⟨string⟩

⟨author⟩::= "*author{*" ⟨string⟩ "*}*"

⟨date⟩::= "*date{*" ⟨dateString⟩ "*}*"

⟨text⟩::= "*text{*" ⟨string⟩ "*}*"

Abbildung 6.11: Grundlegende Sprachelemente der SISL

(*strings*), temporale (*date*, *time*) und boolesche Ausdrücke. Um eine möglichst deklarative Spezifikation zu gewährleisten, werden die Genauigkeit dieser Typen bzw. deren Wertemenge absichtlich nicht weiter konkretisiert. Dies würde ansonsten zu einer großen Anzahl verschiedener Typen führen (z.B. *long* und *short integers*) und zugleich Annahmen bezüglich der Wertemengen aggregierter Komponentenparameter (siehe Abschnitt 6.4.2) erzwingen.

6.4.2 Ausdrücke zur Referenzierung von Komponentenparametern

Wie aus Abschnitt 6.4 hervorgeht, wurde SISL mit der Vorgabe entwickelt, dass für Abbildungsvorschriften relevante Komponentenparameter durch Abfrage bestimmter Datenquellen in periodischen Abständen ermittelt werden. Entsprechend wurden Sprachelemente eingeführt, die festlegen, welche Komponenten bzw. Komponentenparameter einbezogen werden sollen: Eine `aggregation` kann dementsprechend mehrere Komponentenverweise (*resources*) umfassen, die wiederum ein oder mehrere Parameter zur Verfügung stellen. Abbildung 6.12 stellt die Grammatik eines `resource`-Blocks dar. Zunächst kann mit Hilfe des

⟨resource⟩::=	[⟨resource⟩] "*resource{*" ⟨identifier⟩ [⟨description⟩] ⟨source⟩ ⟨sourceAttrib⟩ "*}*"
⟨source⟩::=	"*source{*" ⟨string⟩ "*}*"
⟨sourceAttrib⟩::=	[⟨sourceAttrib⟩] "*sourceAttrib{*" ⟨id⟩ ⟨interval⟩ ⟨return⟩ "*}*"
⟨interval⟩::=	"*interval{*" ⟨float⟩ "*}*"
⟨return⟩::=	"*return{*" ⟨integer⟩ | ⟨float⟩ | ⟨boolean⟩ | ⟨date⟩ | ⟨string⟩ "*}*"

Abbildung 6.12: Das Sprachelement SISL Resource

source-Ausdrucks spezifiziert werden, welche Komponenten als Datenquelle herangezogen werden sollen. Für die jeweilige Abbildungsvorschrift relevante Komponentenparameter werden dann entsprechend als sourceAttrib definiert. Zusätzlich kann festgelegt werden, in welchen Zeitabständen (interval) die betreffende Komponente abgefragt werden soll.

Aus dem Aufbau von resource-Beschreibungen wird weiterhin ersichtlich, dass SISL inhärent auf eine Unterstützung der dieser Arbeit zugrundeliegenden Synthesemethodik ausgerichtet ist. Dies zeigt sich auch in der Festlegung von Abfrageintervallen, was effektiv der Steuerung der nachfolgenden Überwachungsphase dient.

6.4.3 Abbildungsvorschriften

Um zu spezifizieren, in welcher Weise Komponentenparameter verknüpft werden müssen, um ein Dienstattribut zu reflektieren, wird die Einführung von Abbildungsvorschriften notwendig. Letztere repräsentieren vorwiegend Operationen auf komponentenorientierter Managementinformation. Dabei muss neben dem aktuellen Wert eines Komponentenparameters oftmals auch dessen Varianz berücksichtigt werden: In vielen Fällen wird die Berechnung von Mittelwerten oder eine Summenbildung über eine Reihe von Komponentenparametern erforderlich. Diese Aspekte adressiert das Sprachelement function (Abbildung 6.13)

⟨function⟩::=	"*function{*" ⟨identifier⟩ ⟨description⟩ ⟨method⟩ ⟨parameters⟩ ⟨return⟩ "*}*"
⟨method⟩::=	"*method{*" "*sum*" | "*diff*" | "*mult*" | "*div*" | ... "*}*"
⟨parameters⟩::=	"*parameters{*" ⟨valueset⟩ "*}*"
⟨valueset⟩::=	[⟨valueset⟩] "*valueset{*" ⟨resourceRef⟩","⟨integer⟩ | ⟨functionRef⟩ "*}*"
⟨resourceRef⟩::=	"*resourceRef{*"" ⟨id⟩ "*@*" ⟨id⟩ "*',*" ⟨integer⟩ "*}*"
⟨functionRef⟩::=	"*functionRef{*"" ⟨id⟩ "*'}*"

Abbildung 6.13: Abbildungsvorschriften in SISL

in SISL. Es setzt sich aus einer Reihe von `parameters` zusammen, die gemäß der in der `method` spezifizierten Weise abgebildet werden.

Als `parameters` können Literale, Verweise auf Komponentenparameter, aber auch Rückgabewerte anderer Funktionen auftreten. Zur Spezifikation von Abbildungsvorschriften wird weiterhin eine Bibliothek grundlegender mathematischer Operationen von SISL zur Verfügung gestellt. Um Nutzern der Sprache eine Möglichkeit zur Erweiterung und somit Definition eigener Operationen zu geben, werden Operationen dynamisch, auf ihren Namen basierend gebunden. Fügt ein Nutzer also Operationen der Biblitothek hinzu, können diese unmittelbar als `method` referenziert werden. Auf diese Weise kann ein schlanker Sprachentwurf erzielt – die Anzahl denkbarer mathematischer und statistischer Operationen würde die Grammatik unübersichtlich erscheinen lassen – und gleichzeitig ein einfacher Erweiterungsmechanismus zur Verfügung gestellt werden.

6.4.4 Benachrichtigungsoptionen

Vorwiegend der Steuerung der Überwachungsphase widmen sich die im Folgenden vorgestellten Sprachelemente. Eine `notification`-Anweisung dient dabei der Festlegung, unter welcher Bedingung (`condition`-Anweisung) Benachrichtigungen verschickt werden sollen. Mit der

⟨notification⟩::=	[notification] "*notification{*" ⟨condition⟩ ⟨declaration⟩ "*}*"
⟨condition⟩::=	"*condition{*" ⟨identifier⟩ ⟨description⟩ ⟨boolExpression⟩ "*}*"
⟨declaration⟩::=	"*declaration{*" ⟨valueset⟩ "*}*"

Abbildung 6.14: SISL Conditions

`declaration` wird angezeigt, ob einzelne oder mehrere Werte übermittelt werden sollen. Effektiv können so mit den genannten Sprachelementen Vorgaben hinsichtlich der Aktualität eines Dienstattributs gemacht werden.

`Conditions` werden durch logische Ausdrücke in Normalform beschrieben. Dabei kann sowohl auf die konjunktive als auch auf die disjunktive Normalform zurückgegriffen werden. Sie können weiterhin als binäre (z.B. Vergleich zweier Werte) oder unäre Verknüpfungen (z.B. für boolesche Ausdrücke) ausgedrückt werden. SISL unterstützt geläufige relationale und arithmetische Operatoren, wie z.B. Gleich (*equality*), größer/kleiner (*greater/less*), *AND*, *OR* oder *XOR*.

Abschließend zur Vorstellung der *Service Information Specification Language* wird in Abbildung 6.15 noch einmal dessen Schema im Überblick dargestellt. Aus Übersichtlichkeitsgründen wurden dabei vorwiegend Beziehungen zwischen grundlegenden Sprachelementen illustriert, die ausführliche und formale Darstellung des Gesamtschemas findet sich in Anhang B.

6.5 Exemplarische Anwendung

Nachdem mit dem Basismodell der Service-MIB, der Verfeinerung dieses Modells mit Dienstattributen und der Spezifikationssprache SISL die Grundlagen für eine Umsetzung der fünf Teilschritte der Spezifikationsphase (Abschnitt 4.2.2) geschaffen wurden, wird dessen Anwendung nun exemplarische demonstriert.

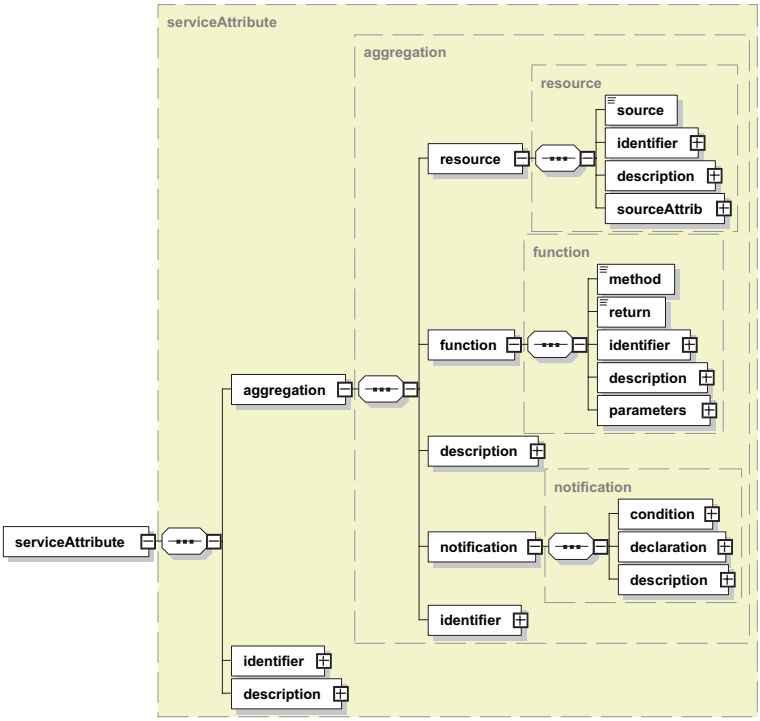

Abbildung 6.15: Das SISL-Schema im Überblick

Dazu wird das in Kapitel 4 bereits verwendete Beispiel – ein Ausschnitt des GRID-Computing Szenarios aus Abschnitt 2.2.2 – erneut aufgegriffen und entsprechend konkretisiert. Der relevante Teil des Szenarios wird in Abbildung 2.4 grau unterlegt dargestellt.

Deklaration statischer Felder

Statische Eigenschaften von Dienstattributen wurden in Abschnitt 6.3 definiert und können in dieser Form unverändert dem Teilschritt I der Methodik zugeführt werden. Als Beispiel wurde das ebenfalls in Kapitel 4 genannte Attribut *OperationalStatus* ausgewählt. Tabelle 6.24 stellt erneut die Definition dieses Attributs innerhalb der Service-MIB dar.

Name	Beschreibung	Datentyp	Werte	Optional
Operational Status	Zeigt an, ob der Dienst momentan ordnungsgemäß funktioniert. Werte kleiner als 1 weisen dabei auf eine Degradierung der Dienstqualität hin.	float	$0 \leq OS \leq 1$	nein 46

Tabelle 6.24: Beispielattribut OperationalStatus

Emittlung relevanter Komponenten und Komponentenparameter

Zur Ermittlung relevanter Komponenten bzw. Komponentenparameter (Teilschritte II und III der Spezifikationsphase) dient das in Abschnitt 6.2.10 vorgestellte Basismodell der Service-MIB. Den Ausgangspunkt bilden *ServiceResourceDependencies* in der Instanzsicht (Abbildung 6.16): Durch Nachverfolgung der entsprechenden - vom Service ausgehenden - funktionalen Abhängigkeiten, können Router, Computing Node und DNS-Server als mögliche Kandidaten ermittelt werden.

Für die Identifikation relevanter Parameter dieser Komponenten ist nun eine weitere Begutachtung der betreffenden *resource* Objekte erforderlich. Für das als Beispiel gewählte Dienstattribut werden in diesem Schritt die jeweiligen *status*-Attribute ermittelt.

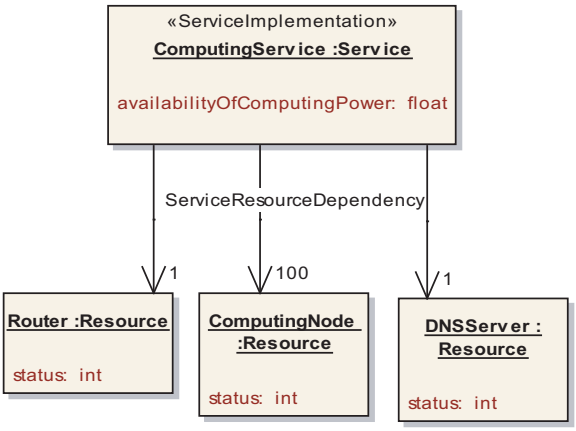

Abbildung 6.16: Objektdiagramm von ServiceResourceDependencies

Festlegung von Messparametern und Abbildungsvorschriften

Zur Umsetzung der Teilschritte IV und V der Spezifikationsphase wurde SISL konzipiert (Abschnitt 6.4). Die grundlegende Idee besteht hierbei darin, Dienstattribute mit SISL-Ausdrücken anzureichern und so die Abbildung von komponentenorientierter auf dienstorientierte Managementinformation als Teil der Service-MIB darzustellen. Bevor jedoch konkrete Abbildungsvorschriften bzw. Messparameter spezifiziert werden können, gilt es zunächst, den Zusammenhang zwischen den vorab identifizierten Komponentenparametern und dem betreffenden Dienstattribut formal zu erfassen. Folgende Formel stellt den Wert des Dienstattributs *SEAvailability* $c \in \mathbb{R}, 0 \leq c \leq 1$ in Abhängigkeit von den in Teilschritt II identifizierten Komponentenparametern dar:

$$c = D \cdot R \cdot \frac{\sum_{i=1}^{N}(p_i a_i)}{\sum_{i=1}^{N} p_i}, \tag{6.1}$$

Dabei repräsentiert N die Anzahl der Rechnerknoten, $D \in \{0, 1\}$ und

Name	totalCurrentlyAvailableComputingPower
Description	The computing power available to service users.
Synonym	N/A
Type	float
Unit	*none*
Constraint	$0 \leq$ computingPowerAvailable ≤ 1
Related Component Parameters	Router (Availability $\rightarrow R$) DNS (Availability $\rightarrow D$) Node* (Availability, Strength $\rightarrow a_i, p_i$)
Aggregation	$D \cdot R \cdot \frac{\sum_{i=1}^{N}(p_i a_i)}{\sum_{i=1}^{N} p_i}$

Tabelle 6.25: Beispielattribut OperationalStatus mit Abbildungsvorschriften

$R \in \{0, 1\}$ die Verfügbarkeit des DNS-Servers bzw. des Routers. Um den Einfluss der verschiedenen Rechnerknoten unterschiedlich zu gewichten, wurde zusätzlich der Faktor p_i eingeführt. Er zeigt die relative Stärke des Rechnerknotens $i \in \{1, .., N\}$ an und wird entsprechend mit deren Verfügbarkeit $a_i \in \{0, 1\}$ verknüpft. In welcher Weise die Werte p_i ermittelt werden, wird an dieser Stelle nicht weiter betrachtet, mögliche Ansätze finden sich beispielsweise in [BB03]. Eine denkbare Möglichkeit bestünde darin, diese Werte als Attribute der jeweiligen *resource*-Objekte zu modellieren und sie so der Service-MIB zuzuführen. Tabelle 6.25 fasst die unter Einbeziehung der genannten Aspekte verfeinerte Spezifikation des Attributs *OperationalStatus* zusammen.

Als letzter Schritt der Spezifikationsphase erfolgt nun die Überführung der genannten Aspekte in SISL, was im Folgenden anhand von Codebeispielen schrittweise erläutert wird. Zunächst werden dabei Meta-Informationen zu der `aggregation` festgelegt, wie in den Zeilen 1-6 des Codebeispiels 6.1 ersichtlich. Dies umfasst Elemente zur Beschreibung des Autors und Erstellungsdatums sowie die Vergabe einer Identifikation (*id*). In den Zeilen 7 bis 30 finden sich die Spezifikationen der

Komponenten und -parameter, die für die Abbildung relevant sind (vgl. Teilschritt II und III): Neben den bei allen `resources` vorhandenem `sourceAttrib` up_down, wird für Rechnerknoten zusätzlich noch der Parameter *strength* berücksichtigt. Die in der Spezifikation der `nodes` vorhandenen *Wildcards* signalisieren dabei, dass sich der Ausdruck auf alle 100 Rechnerknoten bezieht.

```
aggregation{
 id=compServAvailability
 description{
  author{Sailer}
  date{20070501134242}
 }
/* parameters needed by adapters */
 resource{ id=siteA.DNS
  source{siteA.DNS}
  sourceAttrib{up_down}
  interval{1}
  return{boolean}
 }
 resource{ id=siteA.router
  source{siteA.router}
  sourceAttrib{up_down}
  interval{1}
  return{boolean}
 }
 resource{ id=siteA.node*
  source{siteA.node*}
  sourceAttrib{ up_down
   interval{1}
   return{integer}
  }
  sourceAttrib{ strength
   interval{0}
   return{float}
  }
 }
```

Codebeispiel 6.1: Festlegung relevanter Komponenten und Parameter in SISL

Im Codebeispiel 6.2 werden nun eine Reihe von Hilfsfunktionen zur Berechnung des in Formel 6.1 abgebildeten Bruchs definiert. Durch die Funktion *strength.node*.siteA* (Zeilen 32-44) wird für einen Rechnerknoten das Produkt aus den Werten des *strength* und des *up_down*

Attributs berechnet. Mit Hilfe der Funktion *sumAvailability* (Zeilen 46-56) werden die Ergebnisse dieser Berechnung über alle Rechnerknoten summiert und somit der Dividend in Formel 6.1 bestimmt. Die Funktion *sumStrength* (Zeilen 58-68) bildet die Summe der *strength* Attribute aller Rechnerknoten und berechnet damit den Divisor. Mit Hilfe der Funktion *sumAvailability* wird das Produkt der *up_down* Attribute aller Rechnerknoten gebildet. Schließlich werden mit Hilfe der Funktion *availability* (Zeilen 70-82) alle bisher gewonnenen Ergebnisse gemäß des in Formel 1.1 dargestellten Bruchs dividiert.

```
/* Multiplikation von strength und up_down */
function{ id=strength.node*.siteA
  description{
   text{strength of node}
  }
  method{mult}
  parameters{
   valueset{resourceRef{'strength
                @siteA.node*', 1}}
   valueset{resourceRef{'up_down
                @siteA.node*', 1}}
  }
  return{float}
 }
/* Summe der Ergebnisse von  strength.node*.siteA */
 function{ id=sumAvailability
  description{
   text{calculates sum of nodes}
  }
  method{sum}
  parameters{
   valueset{functionRef{
           'strength.node*.siteA'}}
  }
  return{float}
 }
/* Summe aller strength Attribute */
 function{ id=sumStrength
  description{
   text{calculate sum of strengths}
  }
  method{sum}
  parameters{
   valueset{resourceRef{'strength
               @siteA.node*', 1}}
  }
  return{float}
 }
```

```
/* Quotient aus sumAvailability und sumStrength */
function{ id=availability
 description{
  text{calculate ratio of values}
 }
 method{div}
 parameters{
  valueset{functionRef{
            'sumAvailability'}}
  valueset{functionRef{
            'sumStrength'}}
 }
 return{float}
}
```

Codebeispiel 6.2: Hilfsfunktionen zur Berechnung des Beispielattributs

Zur eigentlichen Berechnung des Dienstattributs *OperationalStatus* wird nun in Codebeispiel 6.2 die Funktion *serviceAvailability* definiert. Sie benutzt als Eingabegrößen die Resultate der vorgestellten Hilfsfunktionen bzw. die Werte der *up_down* Attribute von DNS-Server und Router und führt somit die vollständige Berechnung nach Formel 1.1 durch.

```
function{ id=serviceAvailability
 description{
  text{currently available
       computing power  }
 }
 method{mult}
 parameters{
  valueset{functionRef{
            'availability'}}
  valueset{resourceRef{'up_down
            @siteA.DNS', 1}}
  valueset{resourceRef{'up_down
            @siteA.router', 1}}
 }
 return{float}
}
```

Codebeispiel 6.3: Berechnung des Beispielattributs

Als letzter Schritt werden nun Benachrichtigungsoptionen für das Dienstattribut *operationalStatus* spezifiziert (Codebeispiel 6.4). Der Ausdruck `timeout(60)` zeigt dabei an, dass eine Aktualisierung des Attributwerts bzw. des Ergebnisses der Funktion *serviceAvailability* alle 60 Sekunden vorgenommen werden soll.

```
100 notifications{
     condition{ id=service-
102            AvailabilityNotification
      description{
104    text{Availability of service}
      }
106   timeout{60}
     }
108  declaration{
      valueset{functionRef{
110             'serviceAvailability'}}
     }
112 }
}
```

Codebeispiel 6.4: Benachrichtigungsoperationen für das Beispielattribut

Nach der exemplarischen Anwendung der Methodik erfolgt nun eine Zusammenfassung der Ergebnisse dieses Kapitels und ein Zwischenabgleich mit dem in Abschnitt 2.3 vorgestellten Anforderungskatalog.

6.6 Zusammenfassung

Dieses Kapitel diente der Vertiefung der zweiten und gleichzeitig umfangreichsten Teilphase der Synthesemethodik. Dafür wurden zunächst bestehende Arbeiten erweitert bzw. neue Ansätze entwickelt, um notwendige Voraussetzungen zur Anwendung der Methodik zu schaffen. Den ersten Schritt stellte hierbei die Spezifikation grundlegender Entitäten, basierend auf den in Kapitel 5 ermittelten Informationsanforderungen, dar. Verschiedene Teilmodelle in den Bereichen Dienst- und Dienstspezifikation, SLA, QoS, Rollen, Dienstzugangspunkt und Abhängigkeitsbeziehungen wurden entwickelt und in einem Basismodell zusammengeführt.

Als nächster Schritt erfolgte die Erweiterung dieses Basismodells mit Dienstattributen. Hier waren es vor allem anhand von Managementprozessen abgeleitete Informationsanforderungen, die in geeignete Attributsdefinitionen überführt wurden. Zur weiteren Strukturierung wurde die dabei entstehende konkrete Managementinformation in verschiedene Kategorien (*ServiceFault*, *ServicePerformance*,..) eingeordnet und in die Service-MIB integriert. Im Hinblick auf die Synthesemethodik konnte somit der Teilschritt I der Spezifikationsphase abgeschlossen werden.

Die Abbildung von komponentenorientierter auf dienstorientierte Managementinformation stellte Gegenstand des nächsten Abschnitts dieses Kapitels dar. Hierbei fand die Vorstellung der deklarativen Spezifikationssprache SISL und die Erläuterung deren Sprachkonzepte statt.

Am Ende des Kapitels wurde die Anwendung der Synthesemethodik anhand eines aus dem GRID-Szenario (Abschnitt 2.2.2) entlehnten Beispiels demonstriert. Insbesondere wurde die Einbettung der in diesem Kapitel erarbeiteten Teillösungen in die fünf Teilschritte der Synthesemethodik bzw. Spezifikationsphase veranschaulicht: Während mit dem Basismodell die Teilschritte I bis III umgesetzt werden konnten, diente SISL vor allem zur Spezifikation von Abbildungsvorschriften in den Schritten III und IV.

Zwischenabgleich mit dem Anforderungskatalog

Analog zu Abschnitt 5.4 werden nun die entwickelten Teillösungen dem Anforderungskatalog gegenübergestellt und somit deren Tauglichkeit zur Erfüllung der in Kapitel 3 identifizierten Anforderungen ermittelt. Zunächst werden dabei die Kategorien *Anforderungen an die Modelleigenschaften*, *Definition konkreter Managementinformation* und *Beschreibung von Abhängigkeitsbeziehungen* betrachtet (Tabelle 6.26):

- Vergleicht man das Basismodell der Service-MIB mit bestehenden Managementmodellen (SID, CIM) wird deutlich, dass es wesentlich kompakter und übersichtlicher konzipiert wurde und dadurch auch eine einfachere Anwendung gestattet ist (MOD 1). Gleichzeitig konnte mit der Verwendung von Design-Patterns ein möglichst

modularer (MOD 5) und erweiterbarer (MOD 2) Modellentwurf geschaffen werden. Dies zeigt sich beispielsweise in der Einführung von Oberklassen für Dienstattribute (*z.B. ServiceFault*) oder der Beschreibung von Templatehierarchien (z.B. *BasicServiceTemplate*, *SLATemplate*). Zudem beinhaltet das Basismodell keine Vorgaben bzgl. einer bestimmten Art von Diensten, so dass eine breite Anwendbarkeit ermöglicht wird (MOD 4). Basierend auf den Vorgaben der Informationsanalyse wurden weiterhin die wichtigsten Konzepte im Dienstmanagementumfeld in der Modellierung berücksichtigt (z.B. SLA, QoS) und damit die Anforderung Mächtigkeit (MOD 3) erfüllt.

- Ebenfalls an den Vorgaben der Informationsanalyse orientierte sich die Definition konkreter Managementinformationen in diesem Kapitel. Da die gestellten Informationsanforderungen lückenlos in Entitäten und Attribute überführt wurden, wird die Bewertung dieser Kategorie unverändert aus Abschnitt 5.4 übernommen.

- Die Beschreibung von Abhängigkeitsbeziehungen wurde als Teil des Basismodells in Abschnitt 6.2.9 adressiert. Aufgrund der Einführung verschiedener Typen von Abhängigkeitsbeziehungen (funktional oder organisatorisch) kann die geforderte Differenziertheit (DEP 1) als erfüllt gewertet werden. In der weiteren Verfeinerung funktionaler Abhängigkeiten wurden zudem Bezüge zu Dienstfunktionalitäten hergestellt (z.B. Attribut *dependentFunc*) und somit die Möglichkeit zur Auswertung dynamischer Aspekte der Abhängigkeitsbeziehung (DEP 3) geschaffen. Weiterhin wurden die geforderten Beziehungen zwischen Diensten, SLAs und Kunden (DEP 2) in das Modell integriert.

Auf die im zweiten Teil der Spezifikationsphase vorgestellte Spezifikationssprache beziehen sich die der Kategorie *Aggregation komponentenorientierter Managementinformation* zugeordneten Anforderungen (Tabelle 6.27). Der Großteil dieser Anforderungen wurde bereits im Sprachentwurf berücksichtigt (siehe Abschnitt 6.4): SISL dient insbesondere zur Formalisierung von Abbildungsvorschriften (AGG 3) und wurde zudem als deklarative Sprache (AGG 2) konzipiert.

Anforderungen an die Modelleigenschaften	
Anforderung	Bewertung
MOD 1 Einfache Anwendbarkeit	✔
MOD 2 Erweiterbarkeit	✔
MOD 3 Ausdrucksmächtigkeit	✔
MOD 4 Flexibilität	✔
MOD 5 Modularität	✔

Definition konkreter Managementinformation	
Anforderung	Bewertung
KMI 1 Allgemeingültigkeit	✔
KMI 2 Ausrichtung an Geschäftszielen und -prozessen	✔
KMI 3 Vollständigkeit	(✔)
KMI 4 Einheitlichkeit	✔
KMI 5 Nachvollziehbarkeit	✔
KMI 6 Kundenspezifische Aufbereitung	✔
KMI 7 Berücksichtigung interorganisationaler Aspekte	✔

Beschreibung von Abhängigkeitsbeziehungen	
Anforderung	Bewertung
DEP 1 Differenziertheit	✔
DEP 2 Abhängigkeiten zwischen Kunden und SLAs	✔
DEP 3 Berücksichtigung dynamischer Aspekte	✔
✔: erfüllt (✔):teilweise erfüllt **?**: keine Aussage möglich	

Tabelle 6.26: Erfüllung der Anforderungen durch die Spezifikationsphase

SISL ist ebenfalls inhärent auf die Beschreibung von Dienstattributen ausgerichtet, so dass die korrespondierende Anforderung (AGG 1) als erfüllt betrachtet werden kann. Die Unterstützung der Dienstüberwachung (AGG 4) wurde bereits in die Methodik (Übergang zwischen

Aggregation komponentenorientierter Managementinformation	
Anforderung	Bewertung
AGG 1 Berechnung von Dienstattributen	✔
AGG 2 Deklarativität	✔
AGG 3 Formalisierungsgrad	✔
AGG 4 Unterstützung der Dienstüberwachung	✔
✔: erfüllt **?**: keine Aussage möglich	

Tabelle 6.27: Erfüllung der Anforderungen durch die Spezifikationsphase (Fortsetzung)

Spezifikations- und Überwachungsphase) verankert und wird durch entsprechende Sprachkonstrukte in SISL (notifications) umgesetzt.

Kapitel 7

Überwachung und Benutzung von Dienstmanagementinformationen

An die Vorstellung der Analyse- und Spezifikationsphase in den vorangegangenen Kapiteln, schließt sich nun die Betrachtung der letzten beiden Phasen der Synthesemethodik an. Zunächst gilt es dabei, Konzepte zu schaffen, die eine Überwachung der neu spezifizierten Dienstmanagementinformationen unterstützen und insbesondere die anvisierte Aggregation von komponentenorientierter Managementinformation ermöglichen. Hier besteht die vorrangige Idee darin, die genannten Vorgänge durch die in Abschnitt 6.4 entwickelte Spezifikationssprache zu steuern und so ein Zusammenspiel zwischen Spezifikations- und Überwachungsphase zu vermitteln. Dazu wird in diesem Kapitel eine bestehende Monitoringarchitektur dahingehend erweitert, dass eine Auswertung von SISL-Ausdrücken und somit eine Generierung von Dienstmanagementinformationen unterstützt wird.

Den zweiten Teil dieses Kapitels nimmt die Nutzungsphase ein. Sie setzt sich mit einer Einbindung des Service-MIB Ansatzes in den täglichen Betrieb von IT-Providern auseinander. Dies umfasst den Zugriff bestehender Managementanwendungen auf die neu definierten Dienstmana-

gementinformationen und Möglichkeiten zur Integration mit bestehenden Managementmodellen.

7.1 Werkzeugunterstützung für die Überwachungsphase

In SISL vorliegende Beschreibungen von Dienstattributen durch geeignete Werkzeugunterstützung auszuwerten, stellt Gegenstand dieses Abschnitts dar. Dabei besteht das übergeordnete Ziel darin, eine aktuelle Sicht auf den Dienst bereitzustellen und somit die gewünschte Aktualität der Service-MIB zu gewährleisten. Entsprechend fungieren die in diesem Abschnitt vorgestellten Werkzeuge quasi als Dienstagenten (siehe Anforderung REA 3).

Ziel1: Ausführung von arithmetischen Operationen

Prinzipiell muss ein derartiges Werkzeug – neben der Auswertung von SISL-Ausdrücken – dazu imstande sein, darin enthaltene arithmetische Operationen zur Berechnung des Dienstattributs auszuführen. Es muss weiterhin in der Lage sein, in diese Berechnung eingehende Komponentenparameter bzw. deren aktuelle Werte nach den im SISL-Ausdruck getroffenen Vorgaben zu ermitteln. Gemäß der in Abschnitt 2.3 identifizierten Anforderungen sollte dies ferner unter Wiederbenutzung bestehender komponentenorientierter Managementtools (Anforderung REA 1) geschehen. Letzterer Aspekt stellt insbesondere eine Maßzahl für die Praxistauglichkeit der Lösung dar: Typischerweise betreiben IT-Provider bereits verschiedene komponentenorientierte Werkzeuge (vgl. Szenario in Abschnitt 2.2.1) und sind – nicht zuletzt aus Investitionsschutzgründen – selten gewillt, diese komplett durch Dienstmanagementwerkzeuge zu ersetzen.

Ziel2: Einbeziehung bestehender Werkzeuge

Wiederbenutzung erfordert Harmonisierung von Daten

Allerdings impliziert die Integration mit bestehenden Werkzeugen eine Reihe neuer Problemstellungen: Von diesen Werkzeugen bereitgestellte Daten liegen oftmals in verschiedenen Datenformaten vor, die vorab in eine gemeinsame Syntax übersetzt werden müssen. Weiterhin unterscheiden sich diese Werkzeuge in ihrer Benutzung, d.h. es müssen verschiedene APIs zur Datenanfrage oder Konfiguration von Überwachungsoptionen verwendet werden. Ein zur Unterstützung der Überwachungsphase geeignetes Werkzeug muss also weiterhin in der Lage sein, in einer SISL-Anweisung enthaltene Anweisungen auf die Gegebenhei-

ten des jeweiligen komponentenorientierten Managementwerkzeugs abzubilden.

Zur Umsetzung dieses Vorhabens wurden deshalb bestehende Überwachungswerkzeuge (Kapitel 3) dahingehend untersucht, ob durch entsprechende Erweiterungsmassnahmen eine Steuerung des Messvorgangs durch SISL realisiert werden kann. Es zeigte sich, dass mehrere Umstände die Auswahl erheblich einschränken: Bei kommerziellen Produkten erweist es sich grundsätzlich als schwierig, eine Anpassung der verfügbaren Schnittstellen vorzunehmen. Beispielswiese erlaubt der in Abschnitt 3.3.1 vorgestellte *HP OpenView Service Quality Manager* nur die Auswertung eigener Konfigurationsskripten, die allerdings nicht dem von SISL gebotenen Funktionsumfang entsprechen. Ferner wurden zwar Möglichkeiten zum Datenaustausch innerhalb der eigenen Produktreihe geschaffen, die dabei verwendeten proprietären Lösungen können allerdings nicht auf ein Zusammenspiel mit der Service-MIB hin angepasst werden.

Ebenfalls erwiesen sich die als *Open Source Software* verfügbaren Überwachungswerkzeuge (z.B. *Nagios*, *Cacti*) als nicht geeignet. Diese können zwar prinzipiell einfach erweitert werden, sind aber inhärent auf die Überwachung von Netz- und Systemressourcen ausgelegt – der gewünschte Fokus auf höherwertige Dienste fehlt somit. Aus der Begutachtung von Ansätzen, die sich mit der Aggregation von Managementinformationen beschäftigen, ging schließlich hevor, dass diese zwar prinzipiell in der Lage sind, die erforderlichen Berechnungen auszuführen, sich aber inflexibel in Bezug auf die verwendete Spezifikationssprache zeigen (vgl. Abschnitt 3.4). Desweiteren würde eine Anpassung auf SISL aus den in Abschnitt 6.4 dargelegten Gründen die eigentliche Zielsetzung dieser Ansätze kompromittieren.

Die Auswahl fiel schließlich auf die in[DHHS06, DS05] vorgestellte generische Monitoringarchitektur. Dieser Ansatz bietet bereits Möglichkeiten zur Aggregation von Komponentenparametern, ist auf die Überwachung höherwertiger Dienste ausgerichtet und kann gleichzeitig einfach in der gewünschten Weise erweitert werden. Die aus dieser Erweiterung hervorgehende *Service Monitoring Architecture (SMONA)* wird nun im folgenden Abschnitt eingehenden besprochen.

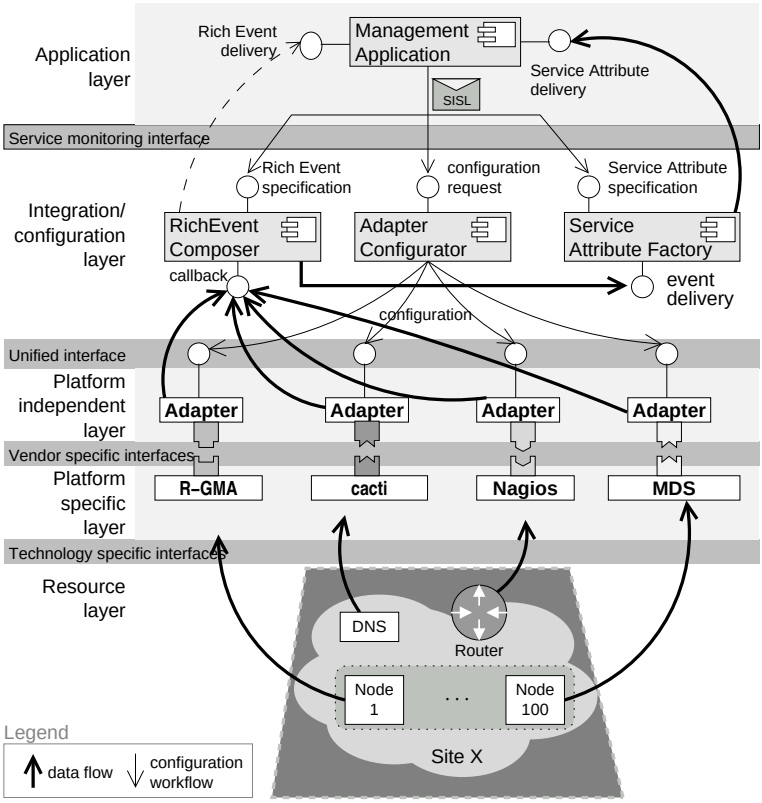

Abbildung 7.1: Architektur zur Synthese von Dienstmanagementinformation aus [DSgF07]

7.1.1 Die Service Monitoring Architecture (SMONA)

Technische Realisierung der Abbildung

Mit der SMONA wurde ein Werkzeug zur technischen Realisierung der Abbildung zwischen komponenten- und dienstorientierter Managementinformation geschaffen. Sie ermittelt, gemäß in SISL-Dokumenten spezifizierter Vorgaben, aktuelle Werte für Dienstattribute und dient somit

einer Akutalisierung der Service-MIB. Gegenüber dem ursprünglichen Architekturentwurf [DHHS06, DS05] wurde zu diesem Zweck eine *Service Attribute Factory* Komponente eingeführt [DSgF07].

Wie aus Abbildung 7.1 ersichtlich, folgt SMONA einem modularen, mehrere Schichten umfassenden Aufbau. Im Folgenden werden die Aufgaben der einzelnen Schichten sowie die vorgenommen Anpassungen vorgestellt.

7.1.1.1 Ressource Schicht

In der untersten Schicht (*resource layer*) sind die zu überwachenden Komponenten bzw. für die Aggregation relevanten Datenquellen angesiedelt. Wie in Abbildung 7.1 durch die wolkenartige Struktur angedeutet, umfasst dies eine Vielzahl unterschiedlichster Datenquellen, die wiederum über verschiedene, oftmals auch proprietäre, Managementschnittstellen verfügen. Um das Zusammenspiel zwischen SISL und der vorliegenden Monitoringarchitektur zu veranschaulichen wurde das in der exemplarischen Anwendung der Synthesemethodik (Abschnitt 6.5) besprochene Beispielszenario herangezogen und die entsprechenden Komponenten in dem *resource layer* in Abbildung 7.1 dargestellt. Sie repräsentieren die in Codebeispiel 6.1 definierten `resources`.

7.1.1.2 Plattformspezifische Schicht

Bestehende komponentenorientierte Managementwerkzeuge treten aus Sicht der SMONA innerhalb der plattformspezifischen Schicht auf. Typischerweise setzen Provider mehrere Werkzeuge parallel und auf ihre Bedürfnisse angepasst ein, so dass nicht selten eine Mischung aus kommerziellen Produkten und skriptbasierten Werkzeugen entsteht. Entsprechend ist eine Uneinheitlichkeit in Bezug auf von diesen Werkzeugen verarbeiteten und zur Verfügung gestellten Daten anzutreffen, die zur weiteren Verarbeitung in der darüberliegenden Schicht normalisiert werden müssen.

7.1.1.3 Plattformunabhängige Schicht

Von den verschiedenen komponentenorientierten Managementwerkzeugen gewonnene Daten zu harmonisieren, stellt die vorrangige Aufgabe der plattformunabhängigen Schicht dar. Sie greift zur Erfüllung dieser Aufgabe auf mehrere Adaptoren zurück, die an den *vendor specific interfaces* verfügbare Daten in ein gemeinsames Format überführen und mit Hilfe eines *unified interface* den oberen Schichten der SMONA zur Verfügung stellen. Im Hinblick auf die Diversität der in der plattformspezifischen Schicht auftretenden Datenquellen muss ein Adaptor in der Lage sein, das betreffende Werkzeug zu konfigurieren und dessen (proprietäres) Datenformat zu verarbeiten. Entsprechend wurden bisher entwickelte Adaptoren meist auf ein konkretes Überwachungswerkzeug hin ausgerichtet, auch wenn sogenannte *general-purpose* Adaptoren durchaus denkbar erscheinen [Dür06].

Adaptoren unterliegen selbst wiederum der Konfiguration durch den *AdaptorConfigurator* (Integrations-/Konfigurationsschicht). Auf diese Weise werden Einstellungen dahingehend vorgenommen, wie von Ressourcen gewonnene Daten verarbeitet und an den *RichEventcomposer* übermittelt werden sollen. An dieser Stelle zeigt sich insbesondere die Verflechtung mit SISL: Instruktionen, die Vorverarbeitung von Daten betreffend, werden auf Basis von `function` Ausdrücken (Abschnitt 6.4.3) bestimmt, Weiterleitungsoptionen durch `conditions` gesteuert (Abschnitt 6.4.4). Letztere legen zeitliche Aspekte der Weiterleitung (z.B. 10 Sekunden Intervall), den Kommunkationstyp des Adaptors (push/pull) sowie zusätzliche, an Ereignisse geknüpfte Bedingungen für die Weiterleitung (z.B. Überschreiten eines Schwellwertes) fest.

Mit der Verwendung spezifischer Adaptoren wurde gleichzeitig ein flexibler Erweiterungsmechanismus für die SMONA geschaffen: Neue Datenquellen können einfach durch Erstellung eines auf die entsprechende Ressource ausgerichteten Adaptors integriert werden. Die Kompatibilität zu den oberen Architekturschichten wird dabei durch die Verwendung einer normierten Schnittstelle *(unified interace)* sichergestellt und muss von neuen Adaptoren entsprechend implementiert werden [Dür06].

7.1.1.4 Integrations- und Konfigurationsschicht

Den wichtigsten Bestandteil der Architektur hinsichtlich der Zusammenstellung von Dienstattributen stellt die Integrations- und Konfigurationsschicht dar. Ihr liegt eine funktionale Aufteilung in drei Komponenten zu Grunde, die jeweils für sie relevante Teile einer SISL Anweisung auswerten und so die gewünschte Aggregation realisieren:

- *RichEvent Composer*
  Der *RichEvent Composer* beschäftigt sich mit der Komposition von Daten gemäß einer SISL-Spezifikation. Er sammelt die (vorverarbeiteten) Daten aller Adaptoren, die für die Zusammenstellung des Dienstattributs von Interesse sind, und erzeugt somit innerhalb von SISL `aggregations` (Abschnitt 6.4.1) spezifizierte Datensätze. Weiterhin wertet der *RichEvent Composer* die `notification` Teile in einer SISL-Anweisung (Abschnitt 6.4.4) aus und bestimmt so, wann und unter welchen Bedingungen Datensätze (sogenannte *Rich Events*) zusammengestellt und weitergeleitet werden sollen. *Rich Events* können dann einerseits Managementanwendungen direkt zur Verfügung gestellt werden, wie mit der gestrichelten Linie in Abbildung 7.1 symbolisiert, oder zur weiteren Verarbeitung den Dienstattributen der *Service Attribute Factory* zugeführt werden.

- *Adapter Configurator*
  Abhängig von der Dienstattributsspezifikation in SISL müssen unterschiedliche Adaptoren ausgewählt und entsprechend konfiguriert werden. Dieser Aufgabe widmet sich die *Adapter Configurator* Komponente; sie ist zuständig für das Management von Adaptoren einschließlich deren Installation, Aktivierung und Konfiguration. Dafür bestimmt der *Configurator* zunächst geeignete Adaptoren durch Auswertung von SISL `resource` (Abschnitt 6.4.2) Deklarationen. Wurde dieser Vorgang abgeschlossen, nimmt er eine Instanziierung und Konfiguration dieser Adaptoren basierend auf `interval`, `function` und `source` Anweisungen vor (siehe Codebeispiele 6.1 und 6.2).

Beide Komponenten der Integrations- und Konfigurationsschicht stellen Bestandteile des ursprünglichen, in [DS05] vorgestellten Entwurfs dar, der ursprünglich nicht auf die Überwachung von Dienstattributen ausgelegt war. Im Hinblick auf die Unterstützung der Überwachungsphase wurde deshalb eine Erweiterung der Architektur notwendig, die zu der Einführung der folgenden Komponente führte:

- *Service Attribute Factory*
  Basierend auf einer Reihe verschiedener *Rich Events* lässt sich schließlich der Wert des Dienstattributs (`serviceAttribute`) ermitteln und die Aggregation vervollständigen. Dafür sammelt die *Service Attribute Factory* Komponente entsprechende Datensätze und appliziert mit Hilfe von SISL `function` Ausdrücken deklarierte Verarbeitungsregeln (siehe Codebeispiel 6.4.3). Eine Bibliothek mathematischer Operationen und Funktionen steht dabei als Teil der *Service Attribute Factory* zur Verfügung und erlaubt dem Anwender die Ausführung komplexer Verarbeitungsregeln. Gleichzeitig wurde ein flexibler Erweiterungsmechanismus eingeführt, der es einem Anwender erlaubt, eigene Funktionen zu verwenden: Über dynamisches Binden von Funktionen wird sichergestellt, dass neue Funktionen einfach und ohne Veränderung des SISL-Sprachentwurfs der Bibliothek hinzugefügt werden können.

7.1.1.5 Applikationsschicht

Dieser Architekturschicht werden Managementapplikationen zugeordnet, die Abnehmer für die in der Integrations- und Konfigurationsschicht erzeugte Managementinformation darstellen. Neben der Weiterleitung von aktuellen Werten für Dienstattribute (Service Attribute delivery) besteht dabei die Möglichkeit, von dem RichEvent Composer generierte Datensätze zu beziehen (Rich Event delivery). In jedem Fall werden die Konventionen für den Datenzugriff dabei von dem *Service monitoring interface* festgelegt.

In welcher Form Managementanwendungen durch SMONA erzeugte Dienstmanagementinformationen nutzen können bzw. wie die vorliegen-

de Architektur zur Realisierung einer Service-MIB verwendet werden kann, wird detaillierter in der Nutzungsphase (Abschnitt 7.2) besprochen. An dieser Stelle bleibt noch zu erwähnen, dass die vorgestellte Monitoringarchitektur zwar vorwiegend auf das Dienstmanagement ausgerichtet wurde, die verwendeten Konzepte aber ein hohes Maß an Generizität aufweisen und deshalb für andere Anwendungsbereiche ebenfalls geeignet erscheinen. Insbesondere können Policy-basierte Ansätze von der Rich Event Weiterleitungsoption profitieren, wie in [Dan07] demonstriert wird.

Im folgenden Abschnitt wird anhand eines einfachen Beispiels die Verarbeitung bzw. Aggregation von Managementinformationen entlang der vorgestellten Architekturschichten erläutert.

7.1.2 Anwendungsbeispiel für die Informationsverfeinerung

Um den Ablauf einer typischen Aggregation von Managementinformationen innerhalb der SMONA zu verdeutlichen, wird im Folgenden das in Abschnitt 6.5 vorgestellte Beispiel erneut aufgegriffen. Auf diese Weise soll dargestellt werden, wie die in den Codebeispielen 6.1 bis 6.4 vorgestellte SISL-Anweisung verarbeitet und letztendlich der Wert des Dienstattributs *operationalPerformance* ermittelt wird.

Wie auf der rechten Seite von Abbildung 7.2 ersichtlich, treten in der Ressource bzw. plattformunabhängigen Schicht gesammelte Informationen zunächst in unterschiedlichen Formaten (verschiedene Syntax, Kodierung, Genauigkeit usw.) auf. Sie repräsentieren die gemäß der SISL `resource` Deklarationen in Codebeispiel 6.1 gewonnenen Rohdaten bzgl. des Status von Router, DNS und der Rechnerknoten.

Mit Hilfe entsprechender Adaptoren werden diese Daten nun in ein konsistentes Format überführt (plattformabhängige Schicht) und an die Integrations- und Konfigurationsschicht weitergeleitet. Im nächsten Schritt fasst der *RichEvent Composer,* basierend auf `aggregation` und `notification` Ausdrücke in Codebeipiel 6.2 und 6.3, diese Daten zu einem *Rich Event* zusammen.

Abschließend für den Informationsverfeinerungsprozess verknüpft die Service Attribute Factory nun die im RichEvent gekapselten Daten gemäß der in Abschnitt 6.5 beschriebenen und in SISL transkribierten

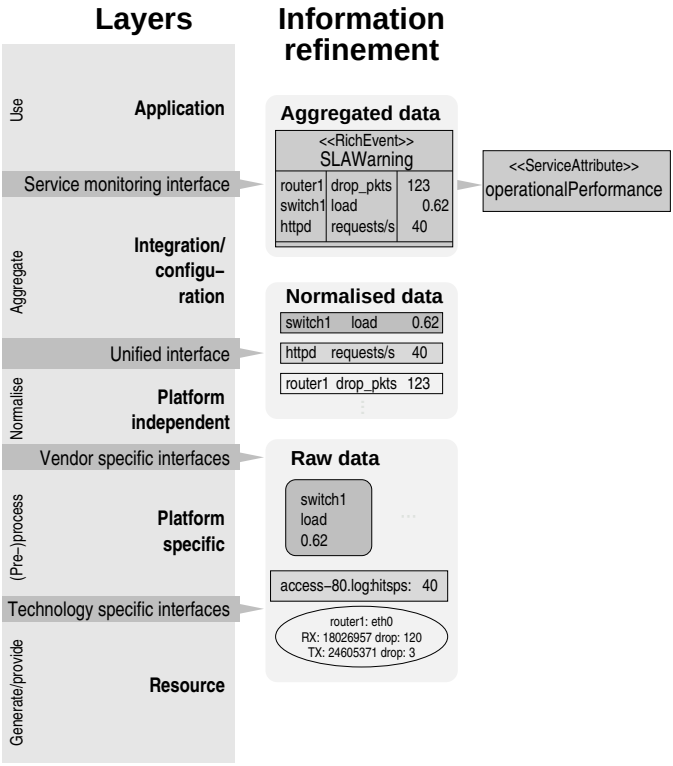

Abbildung 7.2: Datenaggregation entlang der verschiedenen SMONA Schichten in Anlehnung an [DS05]

Formel (Codebeispiel 6.4). Somit wird der Wert des Dienstattributs operationalPerformance bestimmt und kann nun zur Aktualisierung der Service MIB benutzt werden.

7.2 Benutzungsphase

In der anschließenden Phase der Methodik werden nun Konzepte zur operationalen Einbindung des Service-MIB Ansatzes in den täglichen Betrieb eines Providers vorgestellt. Das übergeordnete Ziel besteht hierbei darin, Managementanwendungen die neu definierte Dienstmanagementinformation – möglichst unter Beibehaltung der bestehenden Managementinfrastruktur – zur Verfügung zu stellen. Dabei müssen die folgenden Aspekte Beachtung finden:

- *Integration mit bestehenden Managementanwendungen*
  In der Regel bringen Provider bereits eine Sammlung von Managementanwendungen zum Einsatz und sind aus Investitionsschutzgründen verständerlicherweise nicht gewillt, diese vollständig durch neue Lösungen zu ersetzen. Um eine Verbreitung des Service-MIB Ansatzes zu unterstützen, muss deshalb eine Integration in bestehende Systeme vollzogen bzw.ein Zugriff über die gleichen Schnittstellen ermöglicht werden.
  Offensichtlich stellt eine allgemeine Lösung dieses Problems eine sehr komplexe Aufgabe dar, die auch zum gegenwärtigen Stand der Forschung noch nicht umfassend gelöst werden konnte [HAN99]. Für die vorliegende Arbeit wird deshalb die Integration der Service-MIB in bestehende Managementarchitekturen anhand eines konkreten Beispiels demonstriert (siehe Abschnitt 7.2.1). Die Auswahl fiel hier, vorwiegend aus Verbreitungsgesichtspunkten, auf das *Common information Model*.

- *Szenariospezifische Anpassung der Service-MIB*
  Ein weiterer Aspekt, der in der Nutzungsphase Berücksichtigung finden muss, stellt die Anpassung einer Service-MIB auf die Gegebenheiten eines konkreten Szenarios dar. Dies umfasst vorwiegend die Instanziierung des Modells, die Erstellung geeigneter SISL-Ausdrücke und eventuell die Einbindung neuer SMONA-Adaptoren. In Abschnitt 7.2.2 wird hierfür eine geeignete Vorgehensweise vorgestellt, die sich am Dienstlebenszyklus orientiert.
  Weiterhin wird in diesem Zusammenhang die Anpassung der Service-MIB während des laufenden Betriebs eines Dienstes behandelt und dafür werden ebenfalls Vorgehensrichtlinien erstellt.

Entsprechend der dargestellten Reihenfolge wird nun zunächst eine Möglichkeit zur Integration der Service-MIB in CIM/WBEM skizziert.

7.2.1 Integration der Service-MIB in CIM/WBEM

Die Integration der Service-MIB in bestehende Managementarchitekturen dient dem Ziel, eine Migration für Provider zu erleichtern und somit eine Verbreitung des Ansatzes zu begünstigen. Insbesondere soll Managementanwendungen der Zugriff auf die Inhalte einer Service-MIB unter Benutzung bekannter Mechanismen und Schnittstellen ermöglicht werden.

Dies wird, aufgrund der bereits im vorhergehenden Abschnitt argumentierten Komplexität einer generischen Lösung, im Folgenden anhand des *Common information Model*s (Abschnitt 3.1.1) demonstriert. CIM wurde unter den in Kapitel 3 vorgestellten Managementmodellen ausgewählt, da es gegenüber SID den größeren Verbreitungsgrad besitzt und zudem über eine umfassende Implementierungsbasis verfügt. Ebenfalls erleichtert der objektorientierte Modellierungsansatz von CIM die Integration des Service-MIB Modells, was im Falle des Internet Modells (Abschnitt 3.1.3) nicht gegeben wäre.

Prinzipiell werden dabei Anpassungen sowohl auf Modell- als auch Implementierungsebene notwendig [DK03]: Neben der Einbindung des Service-MIB Modells in die CIM-Klassenhierachie, müssen Komponenten der WBEM-Architektur dahingehend erweitert werden, dass der Datenaustausch mit SMONA bzw. die Benutzung von SISL-Ausdrücken ermöglicht wird:

Einbindung in das CIM Core Model Um die Entitäten des Service-MIB Modells möglichst tief in CIM zu verwurzeln, wurde eine Erweiterung dessen *Core Models* vorgenommen (Abbildung 7.3). Den Ausgangspunkt stellte hierbei die (Wurzel-)Klasse *ManagedElement* dar, die als gemeinsamer Vorfahre aller neu geschaffenen Elemente (durch den Präfix SMIB gekennzeichnet) dient. Mit der Klasse *SMIBService* assoziierte bzw. in Kompositionsbeziehung stehende Elemente entsprechen einem Ausschnitt der in Abbildung 6.9 dargestellten Klassen

des Service-MIB Basismodells und wurden unter Benutzung von CIM-Assoziationen und -Aggregationen integriert (siehe Legende oben rechts im Bild).

Wie aus der Abbildung weiterhin ersichtlich, wurden dabei ebenfalls Anpassungen der Rückgabewerte von Klassenattributen an die Datentypen des CIM-Modells vorgenommen. Dies umfasst beispielsweise die Verwendung von `unint16` bei dem Attribut *type* der Klasse *SMIBQoSParameterSpecification* oder `datetime` für das Attribut *applicableDuring* der Klasse *SMIBSLA*.

Gleichzeitig wurde auf eine Wiederverwendung bestehender CIM-Klassen Wert gelegt. Da bereits in der Definition von Service-MIB Entitäten (Abschnitt 6.2) Bezüge zu CIM hergestellt wurden (Tabellenzeile *Synonym*), konnte diese Information benutzt werden, um geeignete CIM-Klassen zu identifizeren. Entsprechend wurden z.B. Verweise auf Komponenten geschaffen, wie sich in der Assoziation zwischen den Klassen SMIBService und *PysicalResource* zeigt. Hier wurde die im Service-MIB Basismodell als Platzhalter eingeführte Entität *Resource* (Abschnitt 6.2.8) durch das entsprechende Pendant in CIM konkretisiert und somit eine Verbindung zu dem sehr umfangreichen *CIM_Resource*-Modell bzw. darin enthaltener komponentenorientierter Managementinformation geschaffen.

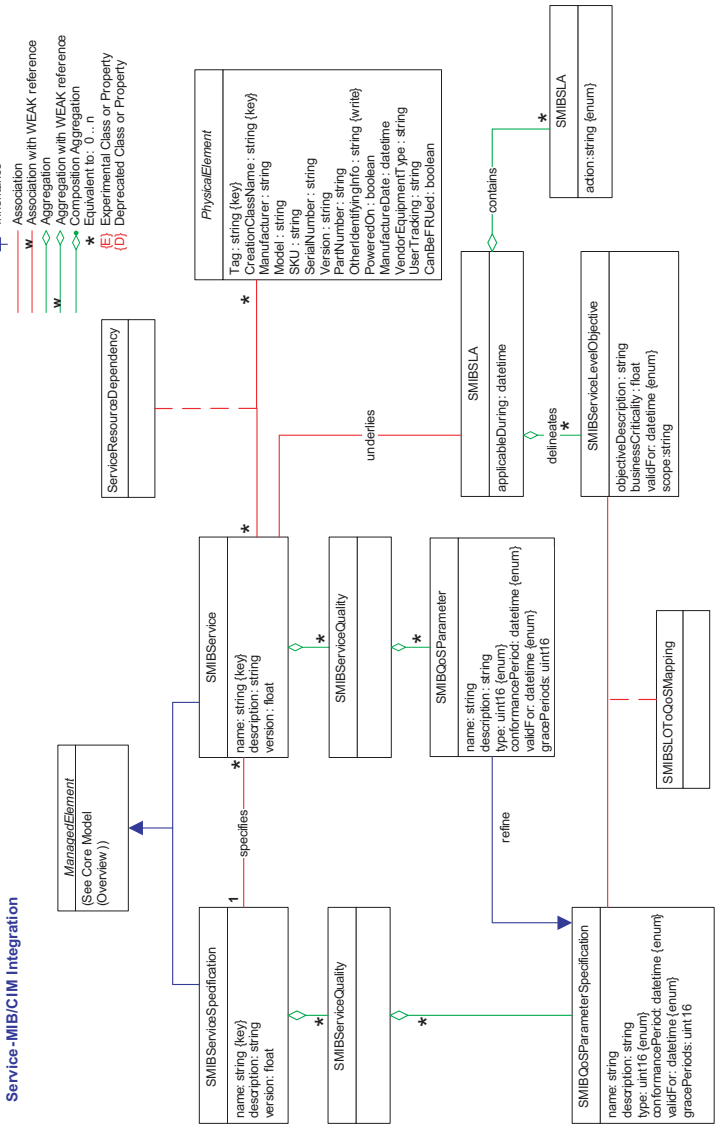

Abbildung 7.3: Integration der Service-MIB in das CIM Core Model

Erweiterung des CIM Object Managers Um die Bestandteile des Service-MIB Ansatzes in eine CIM-Implementierung zu integrieren, ist weiterhin eine Einbindung in die WBEM-Architektur erforderlich. Dies umfasst vorwiegend Erweiterungen des *CIM Object Managers (CIMOM)*, der als zentrale Komponente dieser Architektur fungiert und als eine Art Managementagent angesehen werden kann. Im Folgenden werden deshalb der grundlegende Aufbau eines CIMOMs bzw. die im Rahmen dieser Arbeit vorgenommenen Anpassungen erläutert.

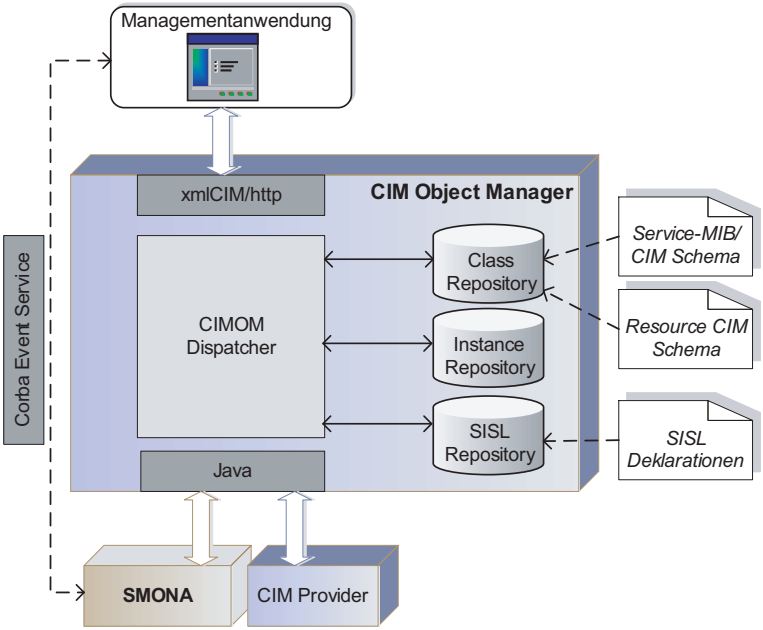

Abbildung 7.4: Integration der Service-MIB in WBEM

Wie sich aus der Abbildung 7.4 entnehmen lässt, bildet der CIMOM die Schnittstelle zu Managementanwendungen und nimmt in *xmlCIM/http* transportierte *WBEM-Requests* entgegen. Diese werden vom *CIMOM*

Dispatcher bearbeitet, der eine statische und aktuelle Sicht des CIM-Modells in entsprechenden internen Datenbanken hält (*Class* und *Instance Respository*). Um dem CIMOM neben dem *Resource CIM Schema* das neu geschaffene *Service-MIB/CIM Schema* zur Verfügung zu stellen, müssen also zunächst die entsprechenden MOF-Dateien eingelesen werden.

Zur Integration mit dem Service-MIB Ansatz wurde hierbei weiterhin ein Repository zur Persistierung von SISL-Deklarationen geschaffen (*SISL Repository*). Dies gibt dem CIMOM die Möglichkeit, die SMONA gewissermaßen als CIM-Provider anzusehen und sie zur Bestimmung von Dienstmanagementinformationen zu verwenden. Dazu leitet er SISL-Ausdrücke weiter und legt von der SMONA erhaltene Werte in seinem *Instance Repository* ab. Um zu entscheiden, ob ein klassischer *CIM Provider* oder die SMONA verwendet werden soll, verwaltet der Dispatcher eine Zuordnung zwischen Klassen und Datenquellen.

Da WBEM bisher nur passive CIM-Provider vorsieht [DK03], wurde weiterhin ein direkter Kommunikationsweg zwischen der SMONA und Managementanwendungen geschaffen. Auf diese Weise können Benachrichtigungen (vgl. SISL-`notifications`) unter Zuhilfenahme des *CORBA Event Buses* [OMG97] empfangen werden.

7.2.2 Szenariospezifische Anpassung der Service-MIB

Entschließt sich ein Provider für den Service-MIB Ansatz, dann stellt sich für ihn vorrangig die Frage, wie er eine Anpassung an die Gegebenheiten seiner spezifischen Umgebung erreichen kann bzw. welche Schritte dafür notwendig sind. Um diesen Prozess zu erleichtern, werden in diesem Abschnitt Vorgehensrichtlinien zur Anwendung der Service-MIB vorgestellt. Die Abfolge der Schritte orientiert sich dabei an dem in Abschnitt 5.2.2 erläuterten Lebenszyklus eines Dienstes:

I. Modellierung der ServiceSpecification
Basierend auf den Ergebnissen der Planungsphase müssen zunächst der *ServiceSpecification* zugeordnete Klassen (Abschnitt 6.2.3) instanziiert und insbesondere deren Einordnung in den Dienstekatalog vorgenommen werden. Dies kann entweder durch

Verfeinerung entsprechender Vorlagen (*BasicService-* und *ProviderCentricTemplates)* oder durch Spezialisierung des Service-MIB Basismodells geschehen. In jedem Fall muss ein eindeutiger Name und Typ vergeben, eine Beschreibung der Dienstfunktionalität erstellt und die aktuelle Lebenszyklusphase (*LifecycleStatus*) des Dienstes dokumentiert werden.
Ebenfalls sollten in diesem Schritt die in Abschnitt 6.3 beschriebenen Dienstattribute den szenariospezifischen Gegebenheiten angepasst werden. Dies kann beispielsweise für das Attribut *ServiceTransactions* (siehe Tabelle 6.20) in folgender Weise vorgenommen werden: Für alle in dem Instanzmodell der *Specification* vorhandenen *Transactions* wird ein Attribut geschaffen, das die Anzahl der in Bearbeitung befindlicher Vorgänge anzeigt (z.B. *ServicePageDeliveries*). Da auf diese Weise eine große Anzahl von Attributen entsteht, muss der Provider ferner eine Einschränkung hinsichtlich für ihn relevanter Aspekte des Dienstes treffen.

II. Instanziierung des SLA-Modells
Wurden die Verhandlungen zwischen Kunde und Provider abgeschlossen, können der daraus resultierende SLA bzw. OLAs und UCs in das Modell überführt werden. Dazu müssen vor allem entsprechende Instanzen für *Service Level Objectives* und *Penalties* (Abschnitt 6.2.6) gebildet und zueinander mit Hilfe der Assoziation *UnmetResultsIn* in Beziehung gesetzt werden. Ebenfalls sollten in diesem Schritt Instanzen für die Rollen *ServiceCustomer*, *ServiceUser* und *ServiceProvider* (Abschnitt 6.2.7) geschaffen werden.

III. Modellierung der Qualitätseigenschaften des Dienstes
Die im SLA enthaltenen Zielvorgaben (*ServiceLevelObjectives)* müssen nun in entsprechende QoS-Eigenschaften (Abschnitt 6.2.5) überführt werden. Dafür bietet es sich zunächst an, *QoSParameterSpecification* bzw. deren *GuarenteeType*, *SemanticDescription* und *ValueRange* festzulegen und mit Hilfe des *SLOToQoSMapping* mit *Objectives* in Beziehung zu setzen. Anschließend sollte eine Verfeinerung des *QoSService* vorgenommen und entsprechende Instanzen erstellt werden.

IV. Modellierung der ServiceImplementation
Mit dem Beginn der Bereitstellungsphase des Dienstes kann eine Instanziierung dem Bereich *ServiceImplementation* (Abschnitt 6.2.3) zugeordneter Klassen bzw. Verfeinerung der Dienstattribute durchgeführt werden. Hierbei muss vor allem die vorher geschaffene Spezifikation mit Implementierungsdetails angereichert werden. Dies umfasst beispielsweise das Binden eines Dienstes an konkrete *Service Access Points* (Abschnitt 6.2.4) oder die Modellierung in dem Szenario auftretender *Dependencies* (Abschnitt 6.2.9) zwischen dem Dienst und dienstrealisierenden Komponenten. Ein konkretes Beispiel für die Abhängigkeitsmodellierung eines Web-Hosting Dienstes findet sich in [Han07].

V. Erstellung von SISL-Ausdrücken und Einbindung in SMONA
Nachdem das instanziierte und szenariospezische Service-MIB Modell erstellt wurde, sollte die Abbildung von Komponentenparametern erfolgen. Zunächst ist hierfür eine Anwendung der Teilschritte II bis V der Spezifikationsphase (Abschnitt 4.2.2) erforderlich, was eine Sammlung von SISL-Spezifikationen für die in Schritt 2 modellierten Dienstattribute zur Folge hat. Möchte der Provider dafür SISL-Funktionen benutzen, die über die Inhalte der Standardbibliothek hinausgehen, kann er dafür den in Abschnitt 6.4.3 beschriebenen Erweiterungsmechanismus verwenden.
Für die Überwachung der neu definierten Dienstmanagementinformation mit SMONA muss weiterhin sichergestellt werden, dass entsprechende Ressourcenadaptoren existieren. Dafür kann es notwendig erscheinen, bestehende Adaptoren zu modifizieren oder neue einzubinden. Die Anleitung hierfür findet sich in in [Dür06].

Die Durchführung der genannten Schritte mündet in einer szenariospezifischen Instanz des Service-MIB Modells sowie entsprechenden SISL-Definitionen und SMONA-Adaptoren, die eine Grundlage für die Betriebsphase des Dienstes darstellen. Welceh Anpassungen im laufenden Betrieb notwendig werden bzw. wie diese umgesetzt werden können, wird im Folgenden weiter ausgeführt.

Anpassung während des laufenden Betriebs Die wohl vordringlichste Aufgabe während des laufenden Betriebs stellt eine Konsistenzsicherung der Service-MIB dar. Dabei sollte entweder durch manuelle Überprüfungsmaßnahmen oder durch entsprechende Werkzeuge sichergestellt werden, dass die Modellinhalte mit der tatsächlichen Situation übereinstimmen. Prinzipiell sollte die Service-MIB dafür in die Configuration, Change und Release Management Prozesse eingebunden werden, so dass Änderungen in der Dienstbereitstellung verlässlich propagiert werden.

Konsistenzsicherung

Weiterhin werden in der Betriebsphase eines Dienstes oftmals Anpassungen hinsichtlich des Detailgrades des Modells oder den Einstellungen für die Dienstüberwachung erforderlich. Dies kann der Fall sein, wenn beispielsweise Abhängigkeiten nicht mit der notwendigen Granularität erfasst oder Messparameter falsch gewählt wurden. Aufgrund dieser Erfahrungen vorgenommene Optimierungen an den Modellinhalten oder SISL-Definitionen sollten dabei sowohl innerhalb des Modells selbst (durch das Attribut *version*) als auch in externen Wissensbasen (z.B. Versionsverwaltungssysteme) dokumentiert werden.

Optimierung der Service-MIB

7.3 Zusammenfassung und Gesamtbewertung des Ansatzes

In diesem Kapitel wurden Konzepte für die Überwachungs- und Nutzungsphase der dieser Arbeit zugrundeliegenden Methodik vorgestellt. Zunächst wurde dabei mit der *Service Monitoring Archittecture (SMONA)* ein Werkzeug präsentiert, das, basierend auf SISL-Ausdrücken, Messungen von Dienstmanagementinformationen vornimmt. SMONA erlaubt insbesondere die Wiederbenutzung bestehender komponentenorientierter Überwachungswerkzeuge und folgt einem aus Schichten bestehenden Architekturentwurf: Während die unteren Schichten einen Zugriff auf Komponentenparameter und notwendige Anpassungen an Datenformaten mit Hilfe von Adaptoren realisieren, beschäftigten sich die oberen Schichten hauptsächlich mit der Erzeugung von Dienstattributen. Wie gezeigt wurde, kann mit SMONA die Abildung komponentenorientierter auf dienstorientierte Managementinformation technisch umgesetzt und zur Aktualisierung der Service-MIB verwendet werden.

Mit der Nutzungsphase wurde vorwiegend die operationale Einbindung einer Service-MIB adressiert. Um den Migrationsprozess für einen Provider zu erleichtern, wurden dabei am Beispiel von CIM/WBEM Integrationsmöglichkeiten in bestehende Managementarchitekturen vorgestellt. Hier bestand das Ziel darin, den Zugriff auf die neu definierte Dienstmanagementinformation unter Benutzung etablierter Schnittstellen und Werkzeuge zu gestatten und damit die Verbreitung des Service-MIB Ansatzes zu unterstützen. Daran anschließend wurden Vorgehensrichtlinien zur szenariospezifischen Anpassung des Service-MIB Ansatzes dargelegt und Erweiterungsmöglichkeiten diskutiert.

Nachdem mit der Überwachungs- und Nutzungsphase alle Teile des Lösungsansatzes vorgestellt wurden, soll nun überprüft werden, inwieweit die in Kapitel identifizierten Anforderungen erfüllt werden konnten.

Umsetzbarkeitsaspekte	
Anforderung	Bewertung
REA 1 Integrationsmöglichkeiten	(✔)
REA 2 Einfache Nutzung durch Anwendungen	✔
REA 3 Pflege der Inhalte	✔
REA 3 Unterstützung durch Managementagenten	✔
REA 5 Reifegrad/Verbreitung	(✔)
✔: erfüllt (✔):teilweise erfüllt **?**: keine Aussage möglich	

Tabelle 7.1: Erfüllung der Anforderungen durch die Überwachungs- und Nutzungsphase

Gesamtbewertung des Service-MIB Ansatzes Bevor der Service-MIB Ansatz dem Anforderungskatalog gegenübergestellt wird, soll zunächst darauf eingegangen werden, welche Anforderungen spezifisch durch die Überwachungs- und Nutzungsphase abgedeckt werden. Dabei wird vorwiegend die Kategorie Umsetzbarkeitsaspekte betrachtet (Tabelle 7.1):

- Zunächst wurde mit SMONA ein Werkzeug geschaffen, das der Aktualisierung der Service-MIB dient und somit der Anforderung

REA 3 gerecht wird. Gleichzeitig bietet diese Architektur wohldefinierte Kommunikationsschnittstellen (*Service monitoring interface*), so dass eine Nutzung durch Managementanwendungen einfach möglich erscheint (Anforderung REA 2).

- Weiterhin wurde in der Nutzungsphase die Einbindung des Lösungsansatzes in das CIM/WBEM vorgenommen und somit eine Integrationsmöglichkeit demonstriert (Anforderung REA 1). Dabei zeigte sich allerdings, dass spezifische Anpassungsvorgänge erforderlich sind und somit eine allgemeine Lösung für dieses Problem gegenwärtig nicht zur Verfügung steht, was mit der Bewertung *teilweise erfüllt* berücksichtigt wurde. Nichtsdestotrotz bietet die Integration mit CIM den Vorteil einer potentiell größeren Verbreitung des Service-MIB Ansatzes (Anforderung REA 5), da Managementanwendungen über bestehende CIM-konforme Schnittstellen und Werkzeuge auf die neu definierte Dienstmanagementinformation zugreifen können. Da allerdings zum gegenwärtigen Zeitpunkt diese Entwicklung nicht abgeschätzt werden kann, wurde diese Anforderung ebenfalls als *teilweise erfüllt* eingestuft.

- Die Anforderung Pflege der Modellinhalte (REA 3) wurde ebenfalls innerhalb der Nutzungsphase durch eine Sammlung diesbezüglicher Vorgehensrichtlinien adressiert. Positiv auf die Erfüllung dieser Anforderung wirkt sich dabei weiterhin der Umstand aus, dass in der Entwicklung der Service-MIB weit verbreitete Standards (UML, XML) Verwendung finden und somit ein weites Feld an Modellierungswerkzeugen zur Verfügung steht.

Nachdem die Gegenüberstellung der verbleibenden Kategorie des Anforderungskatalogs abgeschlossen wurde, kann nun eine Gesamtbewertung des Lösungsansatzes dargestellt werden (Tabelle 7.2).

Gesamtbewertung		
Anforderung		Bewertung
ALL 1	Dienstorientierung	✔
ALL 2	Berücksichtigung des Dienstlebenszyklus	✔
MOD 1	Einfache Anwendbarkeit	✔
MOD 2	Erweiterbarkeit	✔
MOD 3	Ausdrucksmächtigkeit	✔
MOD 4	Flexibilität	✔
MOD 5	Modularität	✔
KMI 1	Allgemeingültigkeit	✔
KMI 2	Ausrichtung an Geschäftszielen und -prozessen	✔
KMI 3	Vollständigkeit	(✔)
KMI 4	Einheitlichkeit	✔
KMI 5	Nachvollziehbarkeit	✔
KMI 6	Kundenspezifische Aufbereitung	✔
KMI 7	Berücksichtigung interorganisationaler Aspekte	✔
DEP 1	Differenziertheit	✔
DEP 2	Abhängigkeiten zwischen Kunden und SLAs	✔
DEP 3	Berücksichtigung dynamischer Aspekte	✔
AGG 1	Berechnung von Dienstattributen	✔
AGG 2	Deklarativität	✔
AGG 3	Formalisierungsgrad	✔
AGG 4	Unterstützung der Dienstüberwachung	✔
REA 1	Integrationsmöglichkeiten	(✔)
REA 2	Einfache Nutzung durch Anwendungen	✔
REA 3	Pflege der Inhalte	✔
REA 3	Unterstützung durch Managementagenten	✔
REA 5	Reifegrad/Verbreitung	(✔)
✔: erfüllt (✔):teilweise erfüllt **?**: keine Aussage möglich		

Tabelle 7.2: Geamtbewertung des Lösungsansatzes dieser Arbeit

Kapitel 8

Zusammenfassung der Ergebnisse und Ausblick

Für eine fortschreitende Integration des Dienstmanagements, die Interoperabilität bestehender und zukünftig entwickelter Dienstmanagementwerkzeuge und die zunehmende Ausrichtung von IT-Betriebsstrukturen an den Geschäftszielen einer Unternehmung ist die Schaffung standardisierter und semantisch festgelegter Dienstmanagementinformationen unerlässlich. In dieser Arbeit wurden mit dem Service-MIB Ansatz methodische und technische Verfahren zur Konzeption einer Managementinformationsbasis für das Dienstmanagement entwickelt.

Ergebnisse dieser Arbeit

Ausgehend von der Beobachtung, dass im Kontext der Modellierung von Managementinformationen auftretende Fachtermini oftmals inkonsistent und mehrdeutig verwendet werden, wurden zunächst relevante Begriffe definiert und im Zuge dessen ein begrifflicher Rahmen für die weiteren Kapitel dieser Arbeit geschaffen. Die anschließende Analyse von

Managementszenarien diente dem Ziel, systematisch Anforderungen an die Konzeption einer Dienstmanagementinformationsbasis abzuleiten. Um verschiedene Problemfelder zu verdeutlichen, wurde hierbei sowohl auf eine weit verbreitete Art von Diensten (*Web-Hosting*) als auch auf ein relatives neuartiges Diensterbringungsszenario (*GRID-Computing*) zurückgegriffen. Dabei konnte die Erkenntnis gewonnen werden, dass sich die Analyse und Spezifikation von Dienstmanagementinformation an einer geschäftsorientierten Perspektive ausrichten sollte und somit Bezüge zwischen Diensten, deren Nutzern und Kunden sowie zu vertraglichen Vereinbarungen geschaffen werden müssen.

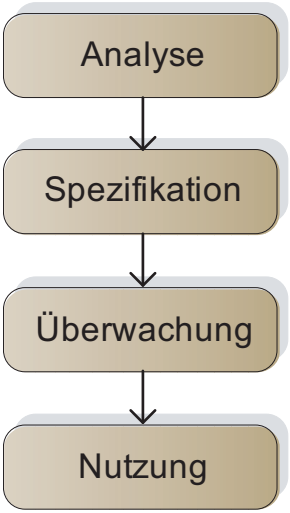

Abbildung 8.1: Phasen der Methodik

Weiterhin wurde deutlich, dass Abhängigkeitsbeziehungen in der Diensterbringung in besonderem Maße Berücksichtigung finden müssen: Eine Dienstmanagementinformationsbasis sollte einerseits verschiedene Arten von Abhängigkeiten (funktionale, organisatorische, topologische, temporale) reflektieren und andererseits eine Abbildung zwischen komponenten- und dienstorientierter Managementinformation schaffen. Letztere sollte dabei insbesondere die Berechnung von Dienstmetriken durch Verdichtung von Komponentenparametern ermöglichen.

Ebenfalls führte die Betrachtung von Managementszenarien zu der Erkenntnis, dass Maßnahmen zur operationalen Einbindung und Aktualisierung einer Dienstmanagementinformationsbasis getroffen werden müssen. Die Ergebnisse der Anforderungsanalyse wurden schließlich in einen strukturierten *Anforderungskatalog* konsolidiert, der im weiteren Verlauf der Arbeit sowohl zur Bewertung bestehender Ansätze als auch für die Entwicklung eines neuen Lösungsansatzes diente.

Daran anschließend wurden bestehende Arbeiten von Standardisierungsgremien, Forschungsansätze und konkrete Produkte evaluiert und

hinsichtlich der vorab identifizierten Anforderungen bewertet. Dabei stellte sich heraus, dass zwar wertvolle Ergebnisse erzielt wurden, sich aber keiner der untersuchten Ansätze zur unmittelbaren Realisierung einer Dienstmanagementinformationsbasis eignet. Dies begründet sich vorwiegend in der nur unzureichenden Ausrichtung an den Geschäftszielen des Providers und dem geringen Umfang an konkreter Dienstmanagementinformation. Ferner sind für die Abbildung von komponenten- und dienstorientierter Managementinformation bisher nur domänen- und technologiespezifische Ansätze vorhanden.

Aus diesem Grund wurde im nächsten Schritt eine aus vier Phasen bestehende Methodik (Abbildung 8.1) zur Realisierung einer Service-MIB und damit der zentrale Ansatz dieser Arbeit vorgestellt. Sukzessive wurden dann im weiteren Verlauf der Arbeit Lösungsmöglichkeiten für die einzelnen Phasen der Methodik entwickelt:

Analysephase In der *Analysephase* wurde der Bedarf an dienstorientierter Managementinformation anhand mehrerer Aspekte ermittelt. Dies umfasste zunächst die Ableitung von Informationsanforderungen anhand des MNM-Dienstmodells sowie unter Berücksichtigung von Dienstlebenszyklus und interorganisationalen Aspekten. Weiterhin wurde ein Schema zur Ableitung von Informationsanforderungen aus standardisierten Beschreibungen von Managementprozessen entwickelt und exemplarisch anhand des ITIL Incident und Service-Level Management Prozesses angewandt. Aus diesen Schritten gewonnene Informationsanforderungen wurden anschließend konsolidiert und in einen Katalog überführt. Letzterer repräsentiert somit grundlegende Informationsanforderungen an eine Service-MIB, zeigt den Bedarf an Entitäten, Attributen und Relationen auf und vermittelte so konkrete Vorgaben hinsichtlich der nachfolgenden Spezifikationsphase.

Spezifikationsphase Die *Spezifikationsphase* widmete sich der Definition konkreter Dienstmanagementinformationen basierend auf den vorab ermittelten Informationsanforderungen. In diesem Zusammenhang wurden zunächst verschiedene Teilmodelle in den Bereichen Dienst- und Dienstspezifikation, SLA, QoS, Rollen, Dienstzugangspunkt und Abhängigkeitsbeziehungen konzipiert und in einem Service-MIB Basismo-

dell zusammengeführt. Dieses wurde anschließend, vorwiegend anhand der aus Managementprozessen abgeleiteten Informationsanforderungen, mit Dienstattributen erweitert.

Als nächster Schritt wurden Konzepte zur Abbildung von komponentenauf dienstorientierte Managementinformation geschaffen. Dazu wurde zunächst die dieser Arbeit zugrundeliegende Methodik mit weiteren Teilschritten (Identifikation relevanter Komponenten bzw. Komponentenparameter, Festlegung von Messparametern und Abbildungsvorschriften) angereichert. Aufgrund der oben genannten Defizite bestehender Arbeiten wurde mit der *Service Information Specification Language (SISL)* eine deklarative Sprache zur Umsetzung dieser Teilschritte konzipiert. SISL ermöglicht die Spezifikation von Vorschriften zur Berechnung von Dienstattributen aus Komponentenparametern und leistet somit einen wichtigen Beitrag zu der anvisierten Informationsabbildung. Zum Abschluss der Spezifikationsphase wurde schließlich die Anwendung der erarbeiteten Lösungen anhand eines aus dem GRID-Szenario entlehnten Beispiels demonstriert.

Überwachungs- und Nutzungsphase Die *Überwachungsphase* beschäftigte sich mit der technischen Umsetzung der Abbildung zwischen komponenten- und dienstorientierter Managementinformation. Hierbei wurde mit der *Service Monitoring Architecture (SMONA)* ein Werkzeug vorgestellt, das SISL-Ausdrücke auswertet und darauf basierend Messungen von Dienstmanagementinformationen vornimmt. Es folgt einem verschiedene Schichten aufweisenden Architekturentwurf und erlaubt die Einbeziehung bestehender komponentenorientierter Überwachungswerkzeuge mit Hilfe von Adaptoren.

In der *Nutzungsphase* wurden schließlich Konzepte zur operationalen Einbindung der Service-MIB in den täglichen Betrieb eines Providers entwickelt. Um den Zugriff auf die neu definierte Dienstmanagementinformation unter Benutzung etablierter Schnittstellen und Werkzeuge zu gestatten, wurde hierbei zunächst eine Möglichkeit zur Integration in die CIM/WBEM-Managementarchitektur vorgestellt. Daran anschließend wurden Vorgehensweisen zur szenariospezifischen Anpassung der Service-MIB erläutert und Möglichkeiten zur Konsistenzsicherung und Optimierung diskutiert. Abschließend konnte durch Gegenüberstellung

mit dem Anforderungskatalog gezeigt werden, dass der Service-MIB Ansatz den gestellten Anforderungen umfassend gerecht wird.

Zukünftige Forschungsfragestellungen

Zum Abschluss dieser Arbeit werden nun Erweiterungsmöglichkeiten für den vorgestellten Lösungsansatz diskutiert und weitere Anwendungsgebiete genannt. Zunächst stehen dabei Fragestellungen im Vordergrund, die sich mit einer szenariospezifischen Anpassung der Service-MIB beschäftigen:

- **Erstellung von Templates für Standard-Dienste**
  Um den Anpassungsprozess einer Service-MIB auf ein konkretes Szenario zu unterstützen, bietet sich die Erstellung von Vorlagen (*Templates*) für Standarddienste an. Dies könnte unter Einbeziehung bestehender Vorarbeiten geschehen: In [KGM03] wurde ein *Template* zur Konfiguration von Web-Hosting Diensten entwickelt und dabei die Annahme getroffen, dass diese Art von Dienst aus Web-, Middleware und Datenbankserver besteht. In ähnlicher Weise könnte eine Vorlage für die Service-MIB von diesem Aufbau ausgehen und entsprechende Entitäten, Dienstattribute und Relationen bereitstellen, die dann als Grundlage zur szenariospezifischen Verfeinerung dienen könnten. Da bereits mit den aus [DR02] adaptierten Konzepten *BasicServiceTemplate* und *ProviderCentricTemplate* Möglichkeiten zur Einbettung von *Templates* geschaffen wurden, kann dieser Verfeinerungsprozess auf Modellebene durch Spezialisierung von Klassen einfach abgebildet werden. In diesem Zusammenhang erscheinen ebenfalls *Templates* für die Erstellung von SISL-Ausdrücken wünschenswert. Dies könnte für den Web-Hosting Dienst beispielsweise Aggregationen umfassen, die den *operationalStatus* dieses Dienstes in Abhängigkeit der oben genannten Komponenten beschreiben.

- **Automatische Anpassung der Service-MIB**
  Um der hohen Änderungsdynamik in der Dienstbereitstellung zu begegnen, wäre weiterhin eine automatisierte Anpassung der Service-MIB von Vorteil. Hierbei sollte durch entsprechende

Werkzeugunterstützung sichergestellt werden, dass Änderungen in der Infrastruktur entsprechend auf die Modellsicht abgebildet werden – oder zumindest die automatische Erkennung von Inkonsistenzen erfolgt. Damit verbundene Fragestellungen finden sich in ähnlicher Form in Forschungsansätzen zur *auto discovery* von Diensten (z.B. [Ale03]) wieder und könnten deshalb unter Einbeziehung der darin gewonnenen Ergebnisse adressiert werden.

Die Abbildung von Informationen zwischen verschiedenen Managementmodellen stellt eine wichtige Voraussetzung für ein integriertes Dienstmanagement dar. Im Rahmen dieser Arbeit wurde dies exemplarisch anhand einer Einbettung der Service-MIB in CIM/WBEM durchgeführt (Abschnitt 7.2.1), wobei allerdings gleichzeitig die Notwendigkeit für eine allgemeine Lösung dieser Problematik betont wurde:

- **Integration auf Meta-Modellebene**
  In den letzten Jahren wurden eine Reihe von Forschungsarbeiten vorgestellt, die sich mit einer modellbasierten Integration von Managementmodellen beschäftigen (z.B. [VVB+03, LDR03]). Die grundlegende Idee besteht dabei darin, durch die Verwendung von Ontologien eine Abbildung zwischen verschiedenen Managementmodellen auf Ebene des *Meta-Models* zu schaffen. Diese Vorgehensweise könnte ebenfalls auf das Service-MIB Modell angewendet und so eine weitergehende Interoperabilität mit bestehenden Managementarchitekturen erzielt werden.

Einen Gegenstand intensiver Forschungsbemühungen stellt gegenwärtig die Automatisierung von Konfigurationsvorgängen dar (z.B. [CCKJ03, BSSG06]). Hierbei wird das Ziel verfolgt, basierend auf Konfigurationsbeschreibungen des Dienstes, automatisch geeignete Komponenten auszuwählen und deren Allozierung und Parametrisierung durchzuführen. Dies soll insbesondere in großen *Datacenters* den Vorteil einer globalen Optimierungsmöglichkeit der zur Verfügung stehenden Ressourcen bieten.

- **Automatisierung der Dienstkonfiguration**
  Zur Automatisierung der Dienstkonfiguration bietet sich eine Erweiterung der in dieser Arbeit entwickelten Methodik an. Unter

Beibehaltung einer Top Down Vorgehensweise könnte so unterhalb von *Analyse-* und *Spezifikationsphase* eine *Konfigurationsphase* geschaffen werden, die Funktionalitäts- und Qualitätsanforderungen des Dienstes auf eine Reihe von Komponenten abbildet und in entsprechende Konfigurationsparameter übersetzt. Die Voraussetzung dafür stellt eine Erweiterung der SMONA dar: Insbesondere müsste ein "Kontrollfluss" geschaffen und Adaptoren in die Lage versetzt werden, Komponenten zu parametrisieren (z.B. durch Anpassung von Konfigurationsdateien).

Wie aus den genannten Aspekten hervorgeht, wurde mit der in dieser Arbeit entwickelten Methodik ein grundlegend neuer Entwicklungsansatz für Managementinformationen vorgestellt, der aufgrund seiner integrativen Sichtweise das Potential zur Anwendung auf weitere Dienstmanagementfragestellungen bietet.

Anhang A

Entitäten

In diesem Anhang werden Entitäten des Service-MIB Basismodells gesammelt dargestellt. Die einzelnen Teilmodelle, in denen diese Entitäten auftreten, finden sich in Abschnitt 6.2.

A.1 Wurzelentiät

Entität ManagedServiceElement	
Beschreibung	Diese Entität repräsentiert die Wurzelentität der Service-MIB. Sie ist gemeinsamer Vorfahre aller Entitäten und beschreibt Eigenschaften, die über alle Entitäten hinweg gleich sind.
Synonym	Einem ähnliche Zweck dienen die Klassen *ManagedElement* in CIM und *RootEntity* in SID.
Prozessbezug	Indirekter Bezug zu allen Prozessen
Verwandte Entitäten	Service, ServiceSpecification, QoS, SLA, SAP, Role, Party, Resource, Dependency

A.2 Dienst und Dienstspezifikation

Entität BasicServiceTemplate	
Beschreibung	Das *BasicServiceTemplate* repräsentiert die generische Spezifikation einer Klasse von Diensten, zusammen mit typischen Diensteigenschaften.
Synonym	Ein entsprechendes Konzept ist gegenwärtig weder in SID noch CIM verfügbar.
Prozessbezug	Indirekter Bezug zu allen Prozessen
Verwandte Entitäten	ManagedServiceElement (Superklasse), ProviderCentricTemplate, Service, ServiceSpecification

Entität ProviderCentricTemplate	
Beschreibung	Das *ProviderCentricTemplate* stellt das Ergebnis der Planungsphase dar und konkretisiert die vom *BasicServiceTemplate* vererbten Diensteigenschaften dahingehend, dass für den jeweiligen Provider typische Ausprägungen bzw. Wertebereiche festgelegt werden
Synonym	Ein entsprechendes Konzept ist gegenwärtig weder in SID noch CIM verfügbar.
Prozessbezug	Indirekter Bezug zu allen Prozessen
Verwandte Entitäten	ManagedServiceElement (Superklasse), BasicServiceTemplate, Service, ServiceSpecification

Entität ServiceCategory/-Spec	
Beschreibung	Die Entitäten *ServiceCategory* und *ServiceCategorySpec* ermöglichen die Einführung eines *ServiceTypes*. Somit kann dem *Service* bzw. der *ServiceSpecification* ein bestimmter Diensttyp zugewiesen und eine Einordnung in Dienstkategorien vorgenommen werden
Synonym	In SID lassen sich sogenannte *ServiceBundles* modellieren, die einem ähnlichen Zweck dienen. CIM verfügt über *Collections*, die Einordnungen erlauben.
Prozessbezug	Insbesondere Incident,Problem, Change und Configuration Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), BasicServiceTemplate, ProviderCentricTemplate, Service, ServiceSpecification

Entität LifecycleStatus	
Beschreibung	Diese Entität verfügt über Attribute zur Beschreibung der aktuellen Lebenszyklusphase des Dienstes (*currentPhase*), des Zeitpunktes, an dem der Service kreiert wurde (*timeOfServiceCreation*), des Zeitpunktes der letzten Änderung der Lebenszyklusphase (*lastPhaseChange*) und der Begründung für diese Änderung (*semanticsForChange*)
Synonym	Synonym
Prozessbezug	Direkter Bezug zu allen Prozessen
Verwandte Entitäten	ManagedServiceElement (Superklasse), Service

Entität ServiceFunctionality/-Spec	
Beschreibung	Zur Beschreibung der Dienstfunktionalität dienen die Entitäten *ServiceFunctionality* und *ServiceSpecificationSpec*. Sie stehen in Aggregationsbeziehung mit dem *Service* bzw. der *ServiceSpecification* und bestehen selbst wiederum aus einer Reihe von *ServiceTransactions*. In dieser Weise können funktionale Bausteine eines Dienstes gegliedert und anhand ihrer Transaktionen weiter detailliert werden
Synonym	In SID finden sich ähnliche Beschreibungsmerkmale innerhalb der Klasse *ServiceCharacteristic*. Zur Beschreibung der Funktionalität in CIM bietet sich die Klasse *Capabilities* an.
Prozessbezug	Wegen der zentralen Bedeutung dieser Entität besteht ein Bezug zu allen Prozessen.
Verwandte Entitäten	ManagedServiceElement (Superklasse), BasicServiceTemplate, ProviderCentricTemplate, Service, ServiceSpecification, ServiceCategory, ServiceCategorySpec

Entität ServiceTransactions/-Spec	
Beschreibung	*ServiceTransactions* bzw. *ServiceTransactionsSpec* charakterisieren generische Dienstprimitive (z.B. Aufruf einer Webseite) und abstrahieren somit effektiv Details der Implementierungstechnologie (z.B. http-Aufrufe). Mit Hilfe dieser Abstraktion wird es möglich, generische Dienstattribute festzulegen.
Synonym	Weder CIM noch SID untergliedern die Dienstfunktionalität weiter. Es müssen die, bereits bei der Entität *ServiceFunctionality* vorgestellten Klassen *Capabilities* und *ServiceCharacteristic* verwendet werden.
Prozessbezug	Wegen der zentralen Bedeutung dieser Entität besteht ein Beuzg zu allen Prozessen.
Verwandte Entitäten	ManagedServiceElement (Superklasse), BasicServiceTemplate, ProviderCentricTemplate, Service, ServiceSpecification, ServiceCategory, ServiceCategorySpec

	Entität Service
Beschreibung	Repräsentiert die Implementierung eines Dienstes gemäß einer *ServiceSpecification* und umfasst aktuell gemessene Werte für Diensteigenschaften. Weiterhin ist der Dienst bestimmten Kategorien zugeordnet und befindet sich in einer definierten Lebenszyklusphase.
Synonym	CIM definiert eine Klasse Dienst, diese trägt aber eine unterschiedliche Semantik. In SID ist eine Klasse Dienst ebenfalls vorhanden, es wird allerdings eine Unterscheidung zwischen *ResourceFacingService* und *CustomerFacingService* eingeführt.
Prozessbezug	Wegen der zentralen Bedeutung dieser Entität besteht ein Bezug zu allen Prozessen.
Verwandte Entitäten	ManagedServiceElement (Superklasse), ServiceSpecification, QoS, Category, LifecycleStatus

	Entität ServiceSpecification
Beschreibung	Diese Entität entsteht durch eine Verfeinerung des *BasicServiceTemplates* bzw. des *ProviderCentricTemplates* und stellt die Basis für die Implementierung des Dienstes für einen bestimmten Kunden dar. Sie beschreibt gewünschte Diensteigenschaften und ermöglicht so einen Soll-Ist Vergleich.
Synonym	Dieses Konzept ist in CIM nicht vorhanden; in SID existiert eine Klasse ServiceSpecification, allerdings wird nicht zwischen den verschiedenen Arten einer Spezifikation unterschieden
Prozessbezug	Wegen der zentralen Bedeutung dieser Entität besteht ein Bezug zu allen Prozessen.
Verwandte Entitäten	ManagedServiceElement (Superklasse), BasicServiceTemplate, ProviderCentricTemplate, Service, Category

A.3 Service Access Point

	Entität ServiceAccessPoint
Beschreibung	Beschreibt die Schnittstelle zwischen Kunde und Provider und wird weiterhin in *ServiceFunctionalityAccessPoint* und *CSMAccessPoint* verfeinert. Mit dem SAP verbundene technische Managementinformationen werden nach *Network-* und *ApplicationAcessProperties* unterschieden; organisatorische Informationen finden sich in der Entität *OrganizationalContact*
Synonym	Eine Klasse *ServiceAccessPoint* ist in CIM vorhanden, die genannten Unterscheidungen werden allerdings nicht berücksichtigt. In SID wird keine Klasse mit ähnlicher Semantik definiert.
Prozessbezug	Vorwiegend Service-Level, Incident, Problem und Change Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), ServiceSpecification, Service, NetworkAccessProperties, ApplicationAccessProperties

Entität ServiceFunctionalityAccessPoint	
Beschreibung	Beschreibt die Schnittstelle zwischen Kunde und Provider hinsichtlich der Nutzung einer Dienstfunktionalität.
Synonym	Weder in SID noch CIM wurde eine derartige Verfeinerung des *ServiceAccessPoints* eingeführt.
Prozessbezug	Vorwiegend Service-Level, Incident, Problem und Change Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), Service Access Point, Service, ServiceSpecification, ServiceUser

Entität CSMAccessPoint	
Beschreibung	Diese Entität dient zur Beschreibung der Customer Service Management Schnittstelle zwischen Kunde und Provider.
Synonym	Weder in SID noch CIM wurde eine derartige Verfeinerung des *ServiceAccessPoints* eingeführt.
Prozessbezug	Vorwiegend Service-Level, Incident, Problem und Change Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), Service Access Point, Service, ServiceSpecification, ServiceCustomer

Entität ApplicationAccessProperties	
Beschreibung	Managementinformationen hinsichtlich der an der Realisierung des SAPs beteiligter Applikationskomponenten werden in der Entität *ApplicationAccessProperties* subsummiert. Beispiele hierfür stellen Verschlüsselungsverfahren oder Protokollports dar.
Synonym	Wird in SID über den *ResourceFacingService* und dessen Assoziation zu einer *Resource* modelliert. In CIM wird die Klasse *ServiceAccessPoint* weiter in *ProtocolEndpoint* und *ServiceAccessURI* verfeinert.
Prozessbezug	Vorwiegend Service-Level, Incident, Problem und Change Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), Service Access Point, Service, ServiceSpecification

Entität OrganizationalContact	
Beschreibung	Diese Entität umfasst für den SAP relevante organisatorische Managementinformationen, wie Ansprechpartner, Emailadresse oder Telefonnummer.
Synonym	Wird in SID über die Klasse *PartyRole* und in CIM durch *AdminDomain* dargestellt
Prozessbezug	Vorwiegend Service-Level, Incident, Problem und Change Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), Service Access Point, Service, ServiceSpecification

Entität NetworkAcessProperties	
Beschreibung	Stützt sich der Zugriff auf den SAP auf Netzkomponenten, so werden in diesem Zusammenhang relevante Managementinformationen mit Hilfe der Entität *NetworkAcessProperties* ausgedrückt. Dies umfasst beispielsweise Parameter des Übertragungsprotokolls, den verwendeten Fehlerkorrekturalgorithmus, die VLAN-ID oder den Standort der entsprechenden Netzkomponente.
Synonym	Wird in SID über den *ResourceFacingService* und dessen Assoziation zu einer *Resource* modelliert. In CIM wird die Klasse *ServiceAccessPoint* weiter in *ProtocolEndpoint* und *ServiceAccessURI* verfeinert.
Prozessbezug	Vorwiegend Service-Level, Incident, Problem und Change Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), Service Access Point, Service, ServiceSpecification

A.4 Quality of Service

Entität ServiceQualitySpecification	
Beschreibung	Diese Entität repräsentiert Zielvorgaben hinsichtlich der Qualitätseigenschaften eines Dienstes. Sie setzt sich aus mehreren QoS-Parametern zusammen und wird von einem bestimmten QoS-Dienst realisiert. Da die Spezification mehrere QoS-Parameter bündelt, dient sie u.a. zur Modellierung von QoS-Kategorien.
Synonym	Eine entsprechende Klasse ist weder in CIM noch in SID existent. In CIM können QoS-Eigenschaften anhand des *CIM_Metrics*-Modells dargestellt werden, in SID bietet sich dafür die Klasse *ServiceCharacteristic* an.
Prozessbezug	Vorwiegend Service-Level, Incident, Problem, Capacity und Continuity Management
Verwandte Entitäten	ServiceSpecification, ServiceQuality, ManagedServiceElement (Superklasse), QoSParameter

Entität ServiceQuality	
Beschreibung	Die Entität repräsentiert die Implementierung einer bestimmten *ServiceQualitySpecification* und beschreibt die aktuell gemessene Dienstqualität. Sie setzt sich ebenfalls aus QoS-Parametern zusammen, die eine Verfeinerung der spezifizierten Parameter hinsichtlich technischer Implementierungsdetails darstellen.
Synonym	In CIM wird eine Klasse *QoS* als Teil des *CIM_Network* Modells definiert, sie bezieht sich aber vorwiegend auf Netz-QoS Dienste. SID enthält momentan keine Entität mit ähnlicher Semantik, es bietet sich jedoch eine Modellierung über die Klasse *ServiceCharacteristic* an
Prozessbezug	Vorwiegend Service-Level, Incident, Problem, Capacity und Continuity Management
Verwandte Entitäten	Service, QoSService, ServiceQualitySpecification, ManagedServiceElement (Superklasse), QoSParameter

Entität QoSService/-Specification	
Beschreibung	Repräsentiert die technische QoS-Implementierung mit Hilfe einer bestimmten QoS-Architektur und stellt eine Verfeinerung der Entität *Service* bzw. *ServiceSpecification* dar. Die Abbildung zwischen verschiedenen QoS-Diensten wird durch die Entität *QoSMappingService* näher umschrieben.
Synonym	Sowohl in CIM als auch SID ist eine gleich lautende Klasse mit ähnlicher Semantik vorhanden. Allerdings fehlt in CIM das Konzept der *QoSServiceSpecification*.
Prozessbezug	Vorwiegend Service-Level, Incident, Problem, Capacity und Continuity Management
Verwandte Entitäten	Service, QoSMappingService, ManagedServiceElement (Superklasse)

Entität QoSMappingService/-Specification	
Beschreibung	Diese Entitäten dienen zur Beschreibung von Koppeldiensten, die den Übergang zwischen verschiedenen QoS-Architekturen bzw. QoS-Diensten realisieren
Synonym	Weder in SID noch CIM ist gegenwärtig ein analoges Konzept verfügbar. Die Modellierung muss deshalb unter Spezialisierung des *QoSService* erfolgen
Prozessbezug	Vorwiegend Service-Level, Incident, Problem, Capacity und Continuity Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), Service, ServiceSpecification, QoSService, ServiceQualitySpecification, QoSParameterSpecification, QoSParameter

Entität QoSParameter/-Specification	
Beschreibung	Diese Entitäten repräsentieren die Spezifikation bzw. die aktuell gemessenen Werte von QoS-Parametern, so dass ein Soll-Ist Vergleich ermöglicht wird.
Synonym	In SID werden QoS-Parameter als Attribute der Klasse *QoSService* beschrieben, in CIM steht dafür das *Metrics-Model* zur Verfügung
Prozessbezug	Vorwiegend Service-Level, Incident, Problem, Capacity und Continuity Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), Service, ServiceSpecification, QoSService, ServiceQualitySpecification,QoSMappingService, QoSMappingServiceSpecification

	Entität SemanticDescription
Beschreibung	Diese Entität enthält eine Beschreibung der Semantik eines *QoSParameters*. Dies wird notwendig, da Parameter mit dem gleichen Bezeichner prinzipiell grundlegend verschiedene Sachverhalte ausdrücken können: Beispielsweise lässt sich die Verfügbarkeit eines Dienstes einerseits über die Anzahl an Dienstausfällen während eines bestimmten Zeitraums definieren, andererseits kann dafür das Verhältnis von beantworteten Anfragen zu allen gestellten Anfragen herangezogen werden. Weiterhin bestimmt die Auswahl der Messpunkte maßgeblich die Semantik eines QoS-Parameters und sollte deshalb in einer formalen Spezifikation berücksichtigt werden
Synonym	In SID werden QoS-Parameter als Attribute der Klasse *QoSService* beaschrieben, in CIM steht dafür das *Metrics-Model* zur Verfügung
Prozessbezug	Vorwiegend Service-Level, Incident, Problem, Configuration, Capacity und Continuity Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), QoSService, ServiceQualitySpecification,QoSMappingService, QoSMappingServiceSpecification, QoSParameter, QoSParameterSpecification

Entität GuarenteeType	
Beschreibung	Mit der Entität *GuarenteeType* steht eine Beschreibungsmöglichkeit hinsichtlich der Garantiegüte für die Einhaltung von *QoS-Parametern* zur Verfügung. Sie erlaubt eine Unterscheidung zwischen dem Garantietyp *Best Effort*, der keine Zusicherungen enthält sowie relativen und absoluten Garantien. Während relative Garantien typischerweise Zusicherungen in der Form von Relationen enthalten, drücken absolute Garantien entweder aus, dass sie zu jedem Zeitpunkt (*deterministische Garantie*) oder über ein statistisches Mittel (*statistische Garantie*) eingehalten werden
Synonym	In SID werden QoS-Parameter als Attribute der Klasse *QoSService* beschrieben, in CIM steht dafür das *Metrics-Model* zur Verfügung
Prozessbezug	Vorwiegend Service-Level, Incident, Problem, Configuration, Capacity und Continuity Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), QoSService, ServiceQualitySpecification,QoSMappingService, QoSMappingServiceSpecification, QoSParameter, QoSParameterSpecification

Entität ValueRange	
Beschreibung	Mit dieser Entität werden mögliche wertemengen für einen *QoS-Parameter* festgelegt. Wertemengen können dabei u.a. als Intervalle oder Liste möglicher Belegungen auftreten und somit z.B. vom Provider eingeführte Qualitätsstufen (z.B. Gold, Silber, Bronze) abbilden. In welcher Weise die einzelnen Werte dieser Menge interpretiert werden müssen, legt dabei wiederum die *SemanticDescription* fest.
Synonym	In SID werden QoS-Parameter als Attribute der Klasse *QoSService* beschrieben, in CIM steht dafür das *Metrics-Model* zur Verfügung
Prozessbezug	Vorwiegend Service-Level, Incident, Problem, Configuration, Capacity und Continuity Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), QoSService, ServiceQualitySpecification,QoSMappingService, QoSMappingServiceSpecification, QoSParameter, QoSParameterSpecification

A.5 Service Level Agreement

	Entität SLA
Beschreibung	Diese Entität beschreibt die vertraglichen Vereinbarungen zwischen Provider und Dienstnutzer hinsichtlich eines konkreten Dienstes. Ein SLA trägt einen bestimmten Typus (*AgreementType*)und besteht aus einer Reihe von Zielvorgaben (*Objectives*) und Pönalen, falls diese nicht eingehalten werden können.
Synonym	In CIM ist ein entsprechendes Konzept nicht vorhanden; die Modellierung in SID weist Ähnlichkeiten zu der vorliegenden auf, eine Klasse *ServiceLevelAgreement* ist vorhanden
Prozessbezug	Service-Level Management, Incident Management, Problem Management, Availability Management, Capacity Management, Financial Management, Continuity Management
Verwandte Entitäten	SLATemplate, AgreementType, ServiceLevelObjective, ServiceLevelPenalties, Service, ServiceCustomer

	Entität SLOToQoSMapping
Beschreibung	Beschreibt das Schema nach dem *ServiceLevelObjectives* auf *QoS-Parameter* abgebildet werden
Synonym	Steht weder in SID noch CIM zur Verfügung
Prozessbezug	Insbesondere Incident, Problem, Change und Configuration Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), SLA, QoSParameter, ServiceLevelObjective

	Entität ServiceLevelObjective
Beschreibung	Beschreibt die in einem SLA enthaltenen Zielvorgaben hinsichtlich der Qualitätseigenschaften eines Dienstes. Zur technischen Realisierung werden SLOs nach einem mit der Entität *SLOToQoSMapping* beschriebenen Schema auf QoS-Parameter abgebildet. Sie stehen in Beziehungen zu *ServiceLevelPenalties*, die Konsequenzen einer Nichteinhaltung von SLOs beschreiben und sind mit einer BusinessCriticality versehen.
Synonym	Nicht vorhanden in CIM; In SID existiert eine Klasse mit gleichem Namen und ähnlicher Semantik
Prozessbezug	Service-Level Management, Incident Management, Problem Management, Availability Management, Capacity Management, Financial Management, Continuity Management
Verwandte Entitäten	SLA, QoSParameter,SLOToQoSMapping, ServiceLevelPenalties

	Entität ServiceLevelPenalty
Beschreibung	Diese Entität repräsentiert die Auswirkungen von Nichteinhaltungen eines *SLAs* bzw. *ServiceLevelObjectives*. Jeder ServiceLevelPenalty ist eine bestimmte Aktion zugeordnet, die die Form der Ausgleichsleistung beschreibt. Ferner stellt diese Entität die Basis für die Berechnung einer *businessCriticality* von *ServiceLevelObjectives* dar.
Synonym	Ein ähnliches Konzept wurde in CIM nicht definiert. Das Pendant zu dieser Entität in SID stellt die Klasse *ServiceLevelSpecConsequence* dar.
Prozessbezug	Service-Level, Incident, Problem, Availability, Capacity, Financial und Continuity Management
Verwandte Entitäten	SLA, ServiceLevelObjective

Entität SLATemplate	
Beschreibung	Das *SLATemplate* fungiert als Container für generische und unparametrisierte SLA-Beschreibungen, die dazu benutzt werden können, gemeinsame Eigenschaften von SLAs für eine bestimmte Klasse von Diensten zu erfassen
Synonym	Gegenwärtig ist weder in SID noch CIM eine entsprechende Klasse vorhanden
Prozessbezug	Insbesondere Incident,Problem, Change und Configuration Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), SLA

Entität AgreementType	
Beschreibung	Die Entität *AgreementType* dient dazu, den Typ des SLAs zu dokumentieren. Sie beschreibt, ob es sich um ein *Operational Level Agreement* oder einen *Underpinning Contract* handelt und ob der SLA als *Service-based*, *Customer-based* oder *Multi-level SLA* konzipiert ist
Synonym	Wird in SID als Attribut der *ServiceLevelSpecification* ausgedrückt und ist in CIM nicht vorhanden
Prozessbezug	Insbesondere Incident,Problem, Change und Configuration Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), SLA, SLA-Template

A.6 Dienstnutzer und -kunden

Entität ServiceUser	
Beschreibung	Repräsentiert den Dienstnutzer, der die Nutzungsfunktionalität des Dienstes an einem *SAP* in Anspruch nimmt. Der *ServiceUser* tritt dabei als Rolle einer bestimmten *Party* auf, über die die tatsächlichen Kontaktinformationen bereitgestellt werden.
Synonym	CIM verfügt nicht über die Party-Abstraktion, es wird aber eine Klasse *User* im *CIM_User* Modell definiert, die noch der Klasse *Role* zugeordnet werden müsste. In SID werden Nutzer und Kunden nicht unterschieden, beide Entitäten werden unter der Klasse *Customer* zusammengefasst.
Prozessbezug	Wird von allen Prozessen benötigt.
Verwandte Entitäten	Role, Party,ServiceFunctionalityAccesPoint

Entität ServiceCustomer	
Beschreibung	Diese Entität beschreibt den Dienstkunden, der in vertragliche Vereinbarungen(*SLA*) hinsichtlich des Dienstes eintritt (*agrees* Assoziation). Weiterhin besteht eine *usesCSM* Beziehung zu dem *CSMAccessPoint*, die eine Nutzung der Customer Service Management Funktionalität reflektiert.
Synonym	In CIM existiert zwar eine Klasse *Role*, die Zuordnung zu einer *OrganizationalUnit* müsste allerdings noch definiert werden. In SID weist die Klasse *Customer* Ähnlichkeit zu dieser Entität auf.
Prozessbezug	Wird von allen Prozessen benötigt
Verwandte Entitäten	Role, Party, CSMAccessPoint

Entität ServiceProvider	
Beschreibung	Die Entität *ServiceProvider* ist über die *provisions* Assoziation mit dem *Service* verknüpft und erlaubt somit eine Zuordnung beteiligter Provider zu einem Dienst
Synonym	In SID ist eine entsprechende Rolle vorhanden, in CIM muss diese über die Klasse *Role* dargestellt werden
Prozessbezug	Wird von allen Prozessen benötigt.
Verwandte Entitäten	ManagedServiceElement (Superklasse), Role, Party, Service

Entität Party und Role	
Beschreibung	Durch die Assoziation zwischen den beiden Entitäten wird es möglich, einer *Party* verschiedene Rollen (*Role*) in der Diensterbringungskette zuzuordnen.
Synonym	Gleichlautende Rollen sind in SID und CIM vorhanden, werden aber abweichend modelliert
Prozessbezug	Wird von allen Prozessen benötigt.
Verwandte Entitäten	ManagedServiceElement (Superklasse),Service, ServiceSpecification, Person, Organization, ServiceCustomer, ServiceUser, ServiceProvider

Entität Person und Organization	
Beschreibung	Bei einer *Party* kann es sich gleichermaßen um eine Einzelperson oder eine Organisation handeln, was durch die Spezialisierung zu *Person* bzw. *Organization* modelliert wird. Diese Entitäten sind als Verweise auf entsprechende Datenquellen intendiert
Synonym	Gleichlautende Rollen sind in SID und CIM vorhanden, werden aber abweichend modelliert
Prozessbezug	Wird von allen Prozessen benötigt
Verwandte Entitäten	ManagedServiceElement (Superklasse), Party, Role

A.7 Resource

Entität Resource	
Beschreibung	Diese Entität repräsentiert Netz- und Applikationskomponenten, die an der Bereitstellung eines Dienstes beteiligt sind. Sie dient als Verweis auf entsprechende Informationen aus anderen Managementmodellen und wird nicht innerhalb der Service-MIB gepflegt bzw. konkretisiert.
Synonym	Eine vergleichbare Semantik trägt die Klassen *ManagedSystemElement* im *CIM_Core* Modell, bzw. die Klasse *Resource* in SID.
Prozessbezug	Wird von allen Prozessen benutzt.
Verwandte Entitäten	ManagedServiceElement (Superklasse), Service, ServiceResourceDependency

A.8 Abhängigkeitsbeziehungen

Entität Dependency	
Beschreibung	Die abstrakte Basisentität *Dependency* fungiert als gemeinsamer Vorfahre aller zur Abhängigkeitsmodellierung verwandter Entitäten
Synonym	In CIM existiert eine gleich lautende Klasse zur Darstellung von Abhängigkeitshierarchien, in SID ist ein entsprechendes Konzept nicht vorhanden
Prozessbezug	Insbesondere Incident,Problem, Change und Configuration Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), OrganizationalDependency, FunctionalDependency

Entität OrganizationalDependency	
Beschreibung	Aus einer Verfeinerung der Basisentität *Dependency* geht die zur Beschreibung von organisatorischen Abhängigkeiten intendierte Entität *OrganizationalDependency* hervor.
Synonym	Organisatorische Abhängigkeiten werden in SID und CIM nicht explizit modelliert
Prozessbezug	Vorwiegend Incident, Problem, Capacity, Configuration, Continuity Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), Dependency

Entität FunctionalDependency	
Beschreibung	Dient als gemeinsamer Vorfahre für alle funktionalen Abhägigkeiten zwischen Diensten und zwischen Diensten und Ressourcen
Synonym	SID und CIM verfügen über spezialisierte Abängigkeitsbeziehungen, eine Superklasse für funktionale Abhängigkeiten wurde nicht vorgesehen
Prozessbezug	Insbesondere Incident,Problem, Change und Configuration Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), Dependency, CompositeFunctionalDependency, ServiceResourceDependency, InterServiceDependency

Entität CompositeFunctionalDependency	
Beschreibung	Die Entität *CompositeFunctionalDependency* dient als Container für funktionale Abhängigkeiten. Somit können mehrere Abhängigkeitsbeziehungen als Einheit betrachtet werden, um beispielsweise Fehler zu erkennen, die von einer bestimmten Ressource verursacht werden
Synonym	Die Komposition von funktionalen Abhängigkeiten wird in SID und CIM nicht explizit adressiert
Prozessbezug	Insbesondere Incident,Problem, Change und Configuration Management
Verwandte Entitäten	ManagedServiceElement (Superklasse), Dependency, CompositeFunctionalDependency, ServiceResourceDependency, InterServiceDependency

Entität InterServiceDependency	
Beschreibung	Beschreibt funktionale Abhängigkeitsbeziehungen zwischen Diensten. Der abhängige Dienst tritt dabei in der Assoziationsrolle *dependent* auf, die (Sub-)Dienste zu denen die Abhängigkeit besteht in der Rolle *antecedent*. Weiterhin zeigt das von der Superklasse FunctionalDependency geerbte Attribut strength an, inwiefern sich ein Ausfall des *antecedent*-Dienstes auf die Funktionalität des *dependent*-Dienstes auswirkt.
Synonym	In CIM besitzt die Klasse *ServiceServiceDependency* aus dem *CIM_Core* Modell eine ähnliche Semantik. SID sieht eine vergleichbare Abhängigkeitsmodellierung nicht vor, am nächsten kommt die Assoziation *CFServiceRequiresRFService*.
Prozessbezug	Vorwiegend Incident, Problem, Capacity, Configuration, Continuity Management
Verwandte Entitäten	Service, Dependency, FunctionalDependency

Entität ServiceResourceDependency	
Beschreibung	Diese Entität repräsentiert funktionale Abhängigkeiten zwischen Diensten und Ressourcen. Letztere bekleiden dabei die Assoziationsrolle *antecedent*, der abhängige Dienst tritt als Rolle *dependent* auf. Potentielle Auswirkungen eines Ressourcenausfalls werden mit dem vererbten Attribut *strength* beschrieben.
Synonym	Das Pendant in CIM stellt die Klasse *DeviceServiceDependency* aus dem *CIM_Core* Modell dar. Eine ähnliche Funktion erfüllt in SID die Assoziation *LogicalResourcesImplementRFS*. Allerdings unterscheidet SID zwischen Customer- und Ressource-Facing Service, wodurch die Abhängigkeitsmodellierung inhärent eine abweichende Semantik besitzt.
Prozessbezug	Vorwiegend Incident, Problem, Capacity, Configuration, Continuity Management
Verwandte Entitäten	Service, Resource, Dependency, FunctionalDependency

Anhang B

SISL XML-Syntax

Nachfolgend wird das vollständige XML-Schema der Service Information Specification Language aufgeführt.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="resource" type="resource"/>
  <xs:complexType name="resource">
    <xs:sequence>
      <xs:element name="source" type="xs:string"/>
      <xs:element name="identifier" type="identifier"/>
      <xs:element name="description" type="description"/>
      <xs:element name="sourceAttrib" type="sourceAttrib"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="function" type="function"/>
  <xs:complexType name="function">
    <xs:sequence>
      <xs:element name="method" type="xs:string"/>
      <xs:element name="return" type="xs:string"/>
      <xs:element name="identifier" type="identifier"/>
      <xs:element name="description" type="description"/>
      <xs:element name="parameters" type="parameters"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="serviceAttribute" type="serviceAttribute"/>
```

```
  <xs:complexType name="serviceAttribute">
24    <xs:sequence>
        <xs:element name="aggregation" type="aggregation"/>
26      <xs:element name="identifier" type="identifier"/>
        <xs:element name="description" type="description"/>
28    </xs:sequence>
  </xs:complexType>
30  <xs:element name="service" type="service"/>
  <xs:complexType name="service">
32    <xs:sequence>
        <xs:element name="serviceAttribute" type="serviceAttribute"/>
34      <xs:element name="identifier" type="identifier"/>
      </xs:sequence>
36  </xs:complexType>
  <xs:element name="identifier" type="identifier"/>
38  <xs:complexType name="identifier">
      <xs:sequence>
40      <xs:element name="id" type="xs:string"/>
      </xs:sequence>
42  </xs:complexType>
  <xs:element name="sourceAttrib" type="sourceAttrib"/>
44  <xs:complexType name="sourceAttrib">
      <xs:sequence>
46      <xs:element name="interval" type="xs:float"/>
        <xs:element name="return" type="xs:string"/>
48      <xs:element name="identifier" type="identifier"/>
      </xs:sequence>
50  </xs:complexType>
  <xs:element name="parameters" type="parameters"/>
52  <xs:complexType name="parameters">
      <xs:sequence>
54      <xs:element name="valueset" type="valueset"/>
      </xs:sequence>
56  </xs:complexType>
  <xs:element name="valueset" type="valueset"/>
58  <xs:complexType name="valueset">
      <xs:sequence>
60      <xs:element name="resourceRef" type="resourceRef"/>
        <xs:element name="functionRef" type="functionRef"/>
62    </xs:sequence>
  </xs:complexType>
64  <xs:element name="resourceRef" type="resourceRef"/>
  <xs:complexType name="resourceRef">
66    <xs:sequence>
        <xs:element name="id" type="xs:string"/>
68    </xs:sequence>
  </xs:complexType>
70  <xs:element name="functionRef" type="functionRef"/>
  <xs:complexType name="functionRef">
72    <xs:sequence>
        <xs:element name="id" type="xs:string"/>
```

```
74      </xs:sequence>
    </xs:complexType>
76  <xs:element name="condition" type="condition"/>
    <xs:complexType name="condition">
78    <xs:sequence>
        <xs:element name="identifier" type="identifier"/>
80      <xs:element name="description" type="description"/>
        <xs:element name="boolExpression" type="boolExpression"/>
82    </xs:sequence>
    </xs:complexType>
84  <xs:element name="declaration" type="declaration"/>
    <xs:complexType name="declaration">
86    <xs:sequence>
        <xs:element name="valueset" type="valueset"/>
88    </xs:sequence>
    </xs:complexType>
90  <xs:element name="boolExpression" type="boolExpression"/>
    <xs:complexType name="boolExpression">
92    <xs:sequence>
        <xs:element name="expression" type="xs:string"/>
94    </xs:sequence>
    </xs:complexType>
96  <xs:element name="aggregation" type="aggregation"/>
    <xs:complexType name="aggregation">
98    <xs:sequence>
        <xs:element name="resource" type="resource"/>
100     <xs:element name="function" type="function"/>
        <xs:element name="description" type="description"/>
102     <xs:element name="notification" type="notification"/>
        <xs:element name="identifier" type="identifier"/>
104   </xs:sequence>
    </xs:complexType>
106 <xs:element name="description" type="description"/>
    <xs:complexType name="description">
108   <xs:sequence>
        <xs:element name="author" type="xs:string"/>
110     <xs:element name="date" type="xs:string"/>
        <xs:element name="text" type="xs:string"/>
112   </xs:sequence>
    </xs:complexType>
114 <xs:element name="notification" type="notification"/>
    <xs:complexType name="notification">
116   <xs:sequence>
        <xs:element name="condition" type="condition"/>
118     <xs:element name="declaration" type="declaration"/>
        <xs:element name="description" type="description"/>
120   </xs:sequence>
    </xs:complexType>
122 </xs:schema>
```

Codebeispiel B.1: SISL XMLSchema

Abkürzungsverzeichnis

ARIS	Architektur integrierter Informationssysteme
BPEL	Business Process Execution Language
BPMN	Business Process Modelling Notation
BSI	British Standards Institution
CAB	Change Advisory Board
CI	Configuration Item
CIM	Common Information Model
CIMOM	CIM Object Manager
CMDB	Configuration Management Database
CMM	Capability Maturity Model
CobiT	Control Objectives for information and related Technology
CPE	Customer Premises Equipment
CSM	Customer Service Management

DEN	Directory Enabled Networks
DEN-ng	Directory Enabled Networks New Generation
DMTF	Distributed Management Task Force
eTOM	enhanced Telecom Operations Map
ETSI	European Telecommunications Standards Institute
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol
IT	Information Technology
ITIL	IT Infrastructure Library
ITSM	IT Service Management
itSMF	IT Service Management Forum
LRZ	Leibniz Supercomputing Center
MDA	Model Driven Architecture
MIB	Management Information Base
MNM	Munich Network Management Team
MOF	Managed Object Format
MTBF	Mean Time Between Failures
MTTR	Mean Time To Repair
MWN	Munich Scientific Network (Münchener Wissenschaftsnetz)

NGOSS	Next Generation Operation Support and Software
OGC	Office of Government Commerce
OLA	Operation Level Agreement
OSI	Open Systems Interconnection
PDF	Portable Document Format
QoS	Quality of Service
SAP	Service Access Point
SID	Shared Information/Data Model
SLA	Service Level Agreement
SLM	Service Level Management
SLO	Service Level Objective
SNMP	Simple Network Management Protocol
SOA	Service Oriented Architecture
TTL	Time To Live
UC	Underpinning Contract
UML	Unified Modelling Language
WBEM	Web Based Enterprise Management
WSDL	Web Services Description Language
XML	Extensible Markup Language

Abbildungsverzeichnis

1.1 Vorgehensmodell dieser Arbeit 9

2.1 Managementobjekte und -objektklassen 12
2.2 Unterscheidung zwischen Informations- und Datenmodellen in Anlehnung an [PS03] 14
2.3 Web-Hosting Dienst am LRZ 19
2.4 Vereinfachtes Grid-Computing Szenario aus [DSgF07] . 24

3.1 Das Dienstmodell des CIM Core Models 41
3.2 eTOM Level 0 Sicht aus [GB902] 44
3.3 Dienstmodell in SID 47
3.4 Service Delivery und Service Support 53
3.5 Aufgaben der CMDB 55
3.6 Der mehrstufige Entwicklungsprozess nach [SWMFR02] 57
3.7 Customer-centric Template Model aus [DR02] 62
3.8 Sprachelemte von WSLA in Anlehnung an [KL03] . . . 69
3.9 Architektureller Aufbau des HP Service Quality Managers in Anlehnung an [HPO] 71
3.10 Die Architektur von InfoVista in Anlehnung an [Ner01] 72

4.1 Der Service-MIB Ansatz 81
4.2 Beispielmodell für Dienstattribut und Aggregation . . . 83

4.3 Methodik zur Synthese von Dienstmanagementinformationen . . . 84
4.4 Teilschritte zur Abbildung von Komponenten- auf Dienstmanagementinformation . . . 89
4.5 Abbildung der Methodik auf die weitere Kapitelstruktur dieser Arbeit . . . 91

5.1 Vorgehen in der Analysephase . . . 94
5.2 MNM-Basismodell nach [GHK+01] . . . 98
5.3 Dienstsicht des MNM-Dienstmodells nach [GHK+01] . . . 101
5.4 Dienstlebenszyklus . . . 106
5.5 Topologie des Web-Hosting Dienstes . . . 110
5.6 Ableitung von Dienstmanagementinformationen aus Prozessen . . . 120
5.7 Generisches ITIL-Prozeßmodell nach [Off01] . . . 122
5.8 Der Incident-Management Prozess nach [Off00] . . . 126
5.9 Betrachtete Anwendungsfälle des Incident-Managements 130
5.10 Der Service-Level Management Prozess nach [Off00] . . . 137
5.11 Betrachtete Anwendungsfälle des Service-Level–Managements . . . 139

6.1 Vorgehen in der Spezifikationsphase . . . 152
6.2 Klassenmodell der WurzelEntität . . . 158
6.3 Klassenmodell von Dienst und Dienstspezifikation . . . 162
6.4 Klassendiagramm zu der Entität ServiceAccessPoint . . . 165
6.5 Klassendiagramm zu der Entität QoS . . . 172
6.6 Klassendiagramm zu der Entität SLA . . . 177
6.7 Klassendiagramm rollenbezogener Entitäten . . . 179
6.8 Klassendiagramm für Abhängigkeiten . . . 183
6.9 Basismodell der Service-MIB (Implementierungsteil) . . . 187
6.10 Kategorien für Diensteigenschaften . . . 189
6.11 Grundlegende Sprachelemente der SISL . . . 211
6.12 Das Sprachelement SISL Resource . . . 212
6.13 Abbildungsvorschriften in SISL . . . 213
6.14 SISL Conditions . . . 214
6.15 Das SISL-Schema im Überblick . . . 215
6.16 Objektdiagramm von ServiceResourceDependencies . . . 217

7.1 Architektur zur Synthese von Dienstmanagementinformation aus [DSgF07] 230
7.2 Datenaggregation entlang der verschiedenen SMONA Schichten in Anlehnung an [DS05] 236
7.3 Integration der Service-MIB in das CIM Core Model . . 240
7.4 Integration der Service-MIB in WBEM 241

8.1 Phasen der Methodik 250

Tabellenverzeichnis

2.1 Anforderungskatalog 36

3.1 Bewertung bestehender Arbeiten 78

5.1 Konkretisierung der Informationsanforderungen in Entitäten 117
5.2 Konkretisierung der Informationsanforderungen in Entitäten (*Fortsetzung*) 118
5.3 Konkretisierung der Informationsanforderungen in Entitäten 144
5.4 Informationsanforderungen hinsichtlich von Dienstattributen 145
5.5 Informationsanforderungen hinsichtlich von Dienstattributen (Fortsetzung) 146
5.6 Erfüllung der Anforderungen durch die Analysephase . . 149

6.1 Entität Beispielentität 156
6.2 Entität ManagedServiceElement 158
6.3 Entität ServiceSpecification 160
6.4 Entität Service 161
6.5 Entität ServiceAccessPoint 166
6.6 Entität ServiceQualitySpecification 169

6.7 Entität ServiceQuality 170
6.8 Entität QoSService 171
6.9 Entität SLA 174
6.10 Entität ServiceLevelObjective 175
6.11 Entität ServiceLevelPenalty 176
6.12 Entität ServiceUser 180
6.13 Entität ServiceCustomer 181
6.14 Entität Resource 182
6.15 Entität InterServiceDependency 184
6.16 Entität ServiceResourceDependency 185
6.17 Attribute zur ganzheitlichen Betrachtung fehlerbezogener Aspekte des Dienstes 194
6.18 Attribute im Kontext von Fehlern in dienstrealisierenden Komponenten 197
6.19 Attribute zur Berichterstellung gegenüber Dienstkunden im Bereich Incident Management 198
6.20 Attribute zur ganzheitlichen Betrachtung von Leistungsaspekten des Dienstes 201
6.21 Attribute zur ganzheitlichen Betrachtung von Leistungsaspekten des Dienstes (Fortsetzung) 203
6.22 Attribute zur Beschreibung von Leistungsgrößen dienstrealisierender Komponenten 205
6.23 Attribute zur Berichterstellung gegenüber Dienstkunden im Bereich Service-Level Management 207
6.24 Beispielattribut OperationalStatus 216
6.25 Beispielattribut OperationalStatus mit Abbildungsvorschriften 218
6.26 Erfüllung der Anforderungen durch die Spezifikationsphase 225
6.27 Erfüllung der Anforderungen durch die Spezifikationsphase (Fortsetzung) 226

7.1 Erfüllung der Anforderungen durch die Überwachungs- und Nutzungsphase 246
7.2 Geamtbewertung des Lösungsansatzes dieser Arbeit . . 248

Literaturverzeichnis

[AAG+04] Agarwal, M., K. Appleby, M. Gupta, G. Kar, A. Neogi und A. Sailer: *Problem Determination Using Dependency Graphs and Run-Time Behavior Models.* In: *Proceedings of the 15th International Workshop on Distributed Systems: Operations and Management (DSOM 2004)*, Seiten 171–182, Davis, California, USA, November 2004. IFIP.

[Aal04] *Verbreitung und Nutzen des prozessorientierten IT-Managements - Wo steht ITIL*, 2004. Ergebnisse der Umfrage.

[AK01] Alexander Keller, Heather Kreger, Karl Schopmeyer: *Towards a CIM Schema for RunTime Application Management.* In: INRIA (Herausgeber): *12th IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM 2001)*, Nancy, France, October 2001. IFIP/IEEE.

[Ale00] Alexander Keller and Gautam Kar: *Dynamic Dependencies in Application Service Management.* In: Hamid R. Arabnia (Herausgeber): *Proceedings of the International Conference on Parallel and Distributed Pro-*

cessing Techniques and Applications (PDPTA 2000), Las Vegas, Nevada, USA, June 2000. CSREA Press.

[Ale03] Alexander Clemm and Anil Bansal: *Auto-Discovery at the Network and Service Management Layer.* In: *Integrated Network Management VII, Managing It All, IFIP/IEEE Eighth International Symposium on Integrated Network Management (IM 2003)*, Band 246, Colorado Springs, USA, März 2003. IFIP/IEEE, Kluwer Academic Publishers.

[BB03] Barmouta, Alexander und Rajkumar Buyya: *GridBank: A Grid Accounting Services Architecture (GASA) for Distributed Systems Sharing and Integration.* ipdps, 00:245a, 2003.

[BBC+71] Bhushan, A., B. Braden, W. Crowther, E. Harslem, J. Heafner, A. McKenzie, J. Melvin, B. Sundberg, D. Watson und J. White: *RFC 172: The File Transfer Protocol.* RFC, IETF, Juni 1971.

[BBC+98] Blake, S., D. Black, M. Carlson, E. Davies, Z. Wang und W. Weiss: *RFC 2475: An Architecture for Differentiated Service.* RFC, IETF, Dezember 1998.

[BEP+94] Brower, D., Ed., B. Purvy, A. Daniel, M. Sinykin und J. Smith: *RFC 1697: Relational Database Management System (RDBMS) Management Information Base (MIB) using SMIv2.* RFC, IETF, August 1994.

[Bit05] Bitterich, C.: *Klassifizierung und Modellierung von Dienstmanagement–Informationen — ein Design–Pattern basierter Ansatz .* Diplomarbeit, Ludwig–Maximilians–Universität München, Dezember 2005.

[BKP02] Bon, Jan van, Georges Kemmerling und Dick Pondman: *IT Service Management – Eine Einführung.* Jan van Haren Publishing, ISBN 90-806713-5-5, 2002. 228 S.

[Bla92] Black, Uyless: *Network Management Standards - The OSI, SNMP and CMOL Protocols.* McGraw-Hill, 1992.

[Bre04] Brenner, M.: *Vom LAN zum Kommunikationsnetz — Netze und Protokolle*, Kapitel enhanced Telecom Operations Map (eTOM). Interest Verlag, Deutschland, Juni 2004.

[Bre06] Brenner, M.: *Classifying ITIL Processes — A Taxonomy under Tool Support Aspects.* In: *Proceedings of First IEEE/IFIP International Workshop on Business–Driven IT Management (BDIM 06)*, Band 2006, Seiten 19–28, Vancouver, Canada, April 2006. IEEE.

[Bre07] Brenner, M.: *Werkzeugunterstützung für ITIL-orientiertes Dienstmanagement – Ein Modellbasierter Ansatz.* Dissertation, Ludwig–Maximilians–Universität München, Juli 2007.

[BSSG06] Brenner, M., M. Sailer, T. Schaaf und M. Garschhammer: *CMDB — Yet Another MIB? On Reusing Management Model Concepts in ITIL Configuration Management.* In: *Large Scale Management of Distributed Systems (Proceedings of DSOM 2006 — 17th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management)*, Band 2006 der Reihe *Lecture Notes in Computer Science, Volume 4269/2006*, Seiten 269–280. Springer Berlin / Heidelberg, Oktober 2006.

[CCKJ03] Choi, Taesang, Hyungseok Chung, Changhoon Kim und Taesoo Jeong: *Design and Implementation of an Information Model for Integrated Configuration and Performance Management of MPLS-TE/VPN/QoS.* In: *Integrated Network Management VII, Managing It All, IFIP/IEEE Eighth International Symposium on Integrated Network Management (IM 2003)*, Band 246, Colorado Springs, USA, März 2003. IFIP/IEEE, Kluwer Academic Publishers.

[CCMW01] Christensen, Erik, Francisco Curbera, Greg Meredith und Sanjiva Weerawarana: *Web Services Description Language (WSDL) 1.1.* Technischer Bericht, W3C, März 2001. W3C Note.

[CFSD90] Case, J.D., M. Fedor, M.L. Schoffstall und C. Davin: *RFC 1157: Simple Network Management Protocol (SNMP).* RFC, IETF, Mai 1990.

[CIM03] *The Value of the Common Information Model (Why CIM?).* Technischer Bericht, Distributed Management Task Force, Juni 2003.

[CIM06] Distributed Management Task Force (DMTF): *Common Information Model (CIM) Version 2.9*, April 2006.

[CMRW96] Case, J., K. McCloghrie, M. Rose und S. Waldbusser: *RFC 1902: Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2).* RFC, IETF, Januar 1996.

[CR99] Caswell, D. und S. Ramanathan: *Using Service Models for Management of Internet Services.* In: *HP Technical Report HPL-1999-43, HP Laboratories*, Palo Alto, California, USA, March 1999.

[Dan07] Danciu, Vitalian A.: *Application of policy-based techniques to process-oriented IT service management.* Dissertation, Ludwig–Maximilians–Universität München, Juli 2007.

[DHHS06] Danciu, V., A. Hanemann, H.-G. Hegering und M. Sailer: *IT Service Management: Getting the View.* In: Kern, E. M., H.-G. Hegering und B. Brügge (Herausgeber): *Managing Development and Application of Digital Technologies.* Springer–Verlag, Juni 2006.

[Dir05] Dirscherl, A.: *Entwicklung eines Managementkonzepts für den Dienst Voice over IP.* Diplomarbeit, Technische Universität München, Februar 2005.

[DK03] Debusmann, Markus und Alexander Keller: *SLA-driven Management of Distributed Systems using the*

Common Information Model. In: *Integrated Network Management VII, Managing It All, IFIP/IEEE Eighth International Symposium on Integrated Network Management (IM 2003)*, Band 246, Colorado Springs, USA, März 2003. IFIP/IEEE, Kluwer Academic Publishers.

[DMT07a] DMTF: *Specification for CIM Operation over HTTP*, Januar 2007.

[DMT07b] DMTF: *Specification for Representation of CIM in XML*, Januar 2007.

[DR00] Dreo Rodosek, G.: *Service Management Platform: The Next Step in Management Tools*. In: *Proceedings of the 11th Annual IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM 2000)*, Lecture Notes in Computer Science (LNCS). Springer, Dezember 2000.

[DR02] Dreo Rodosek, G.: *A Framework for IT Service Management*. Habilitation, Ludwig–Maximilians–Universität München, Juni 2002.

[DR03] Dreo Rodosek, G.: *A Generic Model for IT Services and Service Management*. In: *Integrated Network Management VII, Managing It All, IFIP/IEEE Eighth International Symposium on Integrated Network Management (IM 2003)*, Band 246, Seiten 171–184, Colorado Springs, USA, März 2003. IFIP/IEEE, Kluwer Academic Publishers.

[DRL01] Dreo Rodosek, G. und L. Lewis: *Dynamic Service Provisioning: A User–Centric Approach*. In: *Proceedings of the 12th Annual IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM 2001)*, Seiten 37–48, Nancy, France, Oktober 2001. IFIP/IEEE, INRIA Press.

[DS05] Danciu, V. und M. Sailer: *A monitoring architecture supporting service management data composition*. In: *Proceedings of the 12th Annual Workshop of HP OpenView*

University Association, Nummer 972–9171–48–3, Seiten 393–396, Porto, Portugal, Juli 2005. HP.

[DSgF07] Danciu, V., M. Sailer und N. gentschen Felde: *Declarative Specification of Service Management Attributes.* In: *Proceedings of the 10th IFIP/IEEE International Conference on Integrated Network Management (IM 2007)*, Munich, Germany, Mai 2007. IFIP/IEEE.

[Dür06] Dürr, M.: *Entwicklung von Adaptern für Datenquellen auf Linux–Systemen.* Systementwicklungsprojekt, Ludwig–Maximilians–Universität München, Dezember 2006.

[Ens02] Ensel, C.: *Abhängigkeitsmodellierung im IT Management: Erstellung eines neuen, auf Neuronalen Netzen basierenden Ansatzes.* Dissertation, Ludwig–Maximilians–Universität München, Juni 2002.

[FGM+97] Fielding, R., J. Gettys, J. Mogul, H. Frystyk und T. Berners-Lee: *RFC 2068: Hypertext Transfer Protocol – HTTP/1.1.* RFC, IETF, Januar 1997.

[FK98] Fründ, Svend und Jari Koistinen: *QML: A Language for Quality of Service Specification.* Report HPL-98-10, Software Technology Laboratory, Hewlett-Packard Company, September 1998.

[FK04] Foster, I. und C. Kesselmann (Herausgeber): *The Grid: Bluprint for a New Computing Infrastructure.* Morgan Kaufmann Publishers, 2 Auflage, 2004.

[Fow96] Fowler, Martin: *Analysis Patterns: Reusable Object Models.* Addison-Wesley Professional, 1996.

[Gar04a] Garbani, Jean-Pierre: *Infrastructure Change Management Is The Key To Business Service Management.* Forrester Research, September 2004.

[Gar04b] GARSCHHAMMER, M.: *Dienstgütebehandlung im Dienstlebenszyklus — von der formalen Spezifikation zur rechnergestützten Umsetzung.* Dissertation, Ludwig–Maximilians–Universität München, Juli 2004.

[GB902] TELEMANAGEMENT FORUM: *enhanced Telecom Operations Map (eTOM), The Business Process Framework For The Information and Communications Services Industry*, Juni 2002.

[GB904a] TELEMANAGEMENT FORUM: *Shared Information/Data (SID) Addendum 4SO – Service Overview Business Entity Definitions*, August 2004.

[GB904b] TELEMANAGEMENT FORUM: *Wireless Service Measurement - Key Quality Indicators*, März 2004.

[GB905] TELEMANAGEMENT FORUM: *SLA Management Handbook: Volume 2 - Concepts and Principles*, Juli 2005.

[GHH+02] GARSCHHAMMER, M., R. HAUCK, H.-G. HEGERING, B. KEMPTER, I. RADISIC, H. ROELLE und H. SCHMIDT: *A Case–Driven Methodology for Applying the MNM Service Model.* In: STADLER, R. und M. ULEMA (Herausgeber): *Proceedings of the 8th International IFIP/IEEE Network Operations and Management Symposium (NOMS 2002)*, Seiten 697–710, Florence, Italy, April 2002. IFIP/IEEE, IEEE Publishing.

[GHJV95] GAMMA, E., R. HELM, R. JOHNSON und J. VLISSIDES: *Design Patterns — Elements of Reusable Object–Oriented Software.* Addison-Wesley Professional Computing Series. Addison-Wesley, 1995. ISBN 0-201-63361-2.

[GHK+01] GARSCHHAMMER, M., R. HAUCK, B. KEMPTER, I. RADISIC, H. ROELLE und H. SCHMIDT: *The MNM Service Model — Refined Views on Generic Service Management.* Journal of Communications and Networks, 3(4):297–306, Dezember 2001.

[Gop02] Gopal, Rajeev: *Unifying Network Configuration and Service Assurance with a Service Modeling Language*. In: Stadler, R. und Ulema M. (Herausgeber): *Proceedings of the 8th International IFIP/IEEE Network Operati ons and Management Symposium (NOMS 2002)*, Seiten 711–725, Florence, Italy, April 2002. IFIP/IEEE, IEEE Publishing.

[HAN99] Hegering, H.-G., S. Abeck und B. Neumair: *Integrated Management of Networked Systems – Concepts, Architectures and their Operational Application*. Morgan Kaufmann Publishers, ISBN 1-55860-571-1, 1999. 651 p.

[Han07] Hanemann: *Automated IT Service Fault Diagnosis Based on Event Correlation Techniques*. Dissertation, Ludwig–Maximilians–Universität München, Juli 2007.

[Hau01] Hauck, R.: *Architektur für die Automation der Managementinstrumentierung bausteinbasierter Anwendungen*. Dissertation, Ludwig–Maximilians–Universität München, Juli 2001.

[HKS99] Hazewinkel, H., C. Kalbfleisch und J. Schoenwaelder: *RFC 2594: Definitions of Managed Objects for WWW Services*. RFC, IETF, Mai 1999.

[HLN01] Hegering, H.-G., M. Langer und M. Nerb: *Kunde vs. Dienstleister: Die unterschiedlichen Sichten auf Dienstqualität und SLA–Management*. In: Hegering, H.-G. (Herausgeber): *Proceedings Online Congress 2001*, Band Congressband III der Reihe *Online 2001*, Seiten C320.1–C320.13, Düsseldorf, Februar 2001. Online–Verlag, Velbert, ISBN 3–89077–221–8.

[HMSS06] Hanemann, A., P. Marcu, M. Sailer und D. Schmitz: *Specification of Dependencies for IT Service Fault Management*. In: *Proceedings of the 1st International Conference on Software and Data Technologies (ICSOFT 2006)*, Band 2006, Seiten 257–260, Setubal, Portugal, September 2006. INSTICC press.

[Hom05] Hommel, W.: *An Architecture for Privacy–Aware Inter–Domain Identity Management.* In: *16th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM 2005)*, Barcelona, Spain, Oktober 2005. Springer.

[HPO] *HP OpenVIew.* http://www.hp.com.

[HS05] Hanemann, A. und M. Sailer: *A Framework for Service Quality Assurance using Event Correlation Techniques.* In: *Proceedings of the International Conference on Service Assurance with Partial and Intermittent Resources (SAPIR 2005)*, Lisbon, Portugal, Juli 2005. IARIA/IEEE.

[HSS04a] Hanemann, A., M. Sailer und D. Schmitz: *Assured Service Quality by Improved Fault Management — Service–Oriented Event Correlation.* In: *Proceedings of the 2nd International Conference on Service–Oriented Computing (ICSOC04)*, Seiten 183–192, New York City, NY, USA, November 2004. ACM SIGSOFT and SIGWEB, ACM Press.

[HSS04b] Hanemann, A., M. Sailer und D. Schmitz: *Variety of QoS — the MNM Service Model Applied to Web Hosting Services.* In: *11th International Workshop of the HP OpenView University Association (HPOVUA 2004)*, Band 2004, Paris, France, Juni 2004.

[HSS05a] Hanemann, A., M. Sailer und D. Schmitz: *A Framework for Failure Impact Analysis and Recovery with Respect to Service Level Agreements.* In: *Proceedings of the IEEE International Conference on Services Computing (SCC 2005)*, Orlando, Florida, USA, Juli 2005. IEEE.

[HSS05b] Hanemann, A., M. Sailer und D. Schmitz: *Towards a Framework for Failure Impact Analysis and Recovery with Respect to Service Level Agreements.* In: *Proceedings of the 9th IFIP/IEEE International Conference on Integrated Network Management (IM 2005)*, Nice, France, Mai 2005. IFIP/IEEE.

[HSS05c] Hanemann, A., M. Sailer und D. Schmitz: *Towards a Framework for IT Service Fault Management.* In: *Proceedings of the European University Information Systems Conference (EUNIS 2005)*, Manchester, England, Juni 2005. EUNIS.

[Inf] *Infovista Corporation.* http://www.infovista.com.

[ISO] *Information Technology – Open Systems Interconnection – Structure of Management Information.* IS 10165-X, International Organization for Standardization and International Electrotechnical Committee.

[ISO89a] *Information Processing Systems – Open Systems Interconnection – Basic Reference Model – Part 4: Management Framework.* IS 7498-4, International Organization for Standardization and International Electrotechnical Committee, 1989.

[ISO89b] ISO/IEC: *Management Framework*, 1989. ISO 7498-4.

[ISO93] ISO/IEC: *Management Information Model*, 1993. ISO 10165-1.

[IT 02] IT Service Management Forum Germany (itSMF): *IT Service Management, eine Einführung.* Van Haren Publishing, ISBN 90-806713-5-5, 2002.

[JMFW03] J.P. Martin-Flatin, D. Srivastava und A. Westerinen: *Iterative Multi-Tier Management Information Modeling.* IEEE Communications Magazine, 41(12):92–99, Dezember 2003.

[Kai99] Kaiser, T.: *Methodik zur Bestimmung der Verfügbarkeit von verteilten anwendungsorientierten Diensten.* Dissertation, Technische Universität München, April 1999.

[Kav00a] Kavasseri, R.: *RFC 2981: Event MIB.* RFC, IETF, Oktober 2000.

[Kav00b] KAVASSERI, R.: *RFC 2982: Distributed Management Expression MIB*. RFC, IETF, Oktober 2000.

[KBK00] KELLER, A., U. BLUMENTHAL und G. KAR: *Classification and Computation of Dependencies for Distributed Management*. In: TOHME, S. und M. ULEMA (Herausgeber): *Proceedings of the Fifth IEEE Symposium on Computers & Communications*, Antibes - Juan Les Pins, France, Juli 2000. IEEE.

[Kel98] KELLER, A.: *CORBA–basiertes Enterprise Management: Interoperabilität und Managementinstrumentierung verteilter kooperativer Managementsysteme in heterogener Umgebung*. Dissertation, Technische Universität München, Dezember 1998.

[KF94a] KILLE, S. und N. FREED: *RFC 1565: Network Services Monitoring MIB*. RFC, IETF, Januar 1994.

[KF94b] KILLE, S. und N. FREED: *RFC 1566: Mail Monitoring MIB*. RFC, IETF, Januar 1994.

[KGM03] K.GARG, PANKAY, MARTIN GRISS und VIJAY MACHIRAJU: *Auto-Discovering Configurations for Service Management*. Journal of Network and Systems Management, 11(2):217–239, 2003.

[KK01] KAR, GAUTAM und ALEXANDER KELLER: *An Architecture for Managing Application Services over Global Networks*. In: *IEEE INFOCOM 2001*, Seiten 1020–1027, 2001.

[KKC00] KAR, G., A. KELLER und S.B. CALO: *Managing Application Services over Service Provider Networks: Architecture and Dependency Analysis*. In: HONG, J. W. und R. WEIHMAYER (Herausgeber): *Proceedings of the 7th IEEE/IFIP Network Operations and Management Symposium (NOMS'2000)*, Seiten 61–75, Honolulu, Hawaii, USA, April 2000.

[KKPS99] Kalbfleisch, C., C. Krupczak, R. Presuhn und J. Saperia: *RFC 2564: Application Management MIB.* RFC, IETF, Mai 1999.

[KL03] Keller, Alexander und Heiko Ludwig: *The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services.* Journal of Network and Systems Management, Special Issue on E-Business Management, 11(1), März 2003.

[KS98] Krupczak, C. und J. Saperia: *RFC 2287: Definitions of System-Level Managed Objects for Applications.* RFC, IETF, Februar 1998.

[Lan06] Lange, S.: *Formalisierung von Aggregationsvorschriften für Dienstinformationen.* Diplomarbeit, Technische Universität München, Mai 2006.

[LDR03] Lavinal, Emmanuel, Thierry Desprats und Yves Raynaud: *A Conceptual Framework for Building CIM-based Ontologies.* In: *Integrated Network Management VII, Managing It All, IFIP/IEEE Eighth International Symposium on Integrated Network Management (IM 2003)*, Band 246, Colorado Springs, USA, März 2003. IFIP/IEEE, Kluwer Academic Publishers.

[Lew99] Lewis, Lundy: *Service Level Management of Enterprise Networks.* Artech House Publishers, 1999.

[LLN99] Langer, M., S. Loidl und M. Nerb: *Customer Service Management: Towards a Management Information Base for an IP Connectivity Service.* In: *Proceedings of the 4th IEEE Symposium on Computers and Communications (ISCC99)*, Sharm El Sheikh, Egypt, Juli 1999.

[LN00] Langer, M. und M. Nerb: *Customer Service Management: An Information Model for Communication Services.* In: Linnhoff-Popien, C. und H.-G. Hegering (Herausgeber): *Trends in Distributed Systems: Towards a*

Universal Service Market. Proceedings of the third International IFIP/GI Working Conference, USM 2000, Nummer 1890 in *Lecture Notes in Computer Science (LNCS)*, Munich, Germany, September 2000. Springer.

[LSE03] Lamanna, D., J. Skene und W. Emmerich: *Slang: A Language for defining Service Level Agreements*. In: *9th IEEE InternationalWorkshop on Future Trends of Distributed Computing Systems (FTDCS 2003)*, San Juan, Puerto Rico, Mai 2003.

[MF01] Martin-Flatin, J.: *Toward Universal Information Models in Enterprise Management*. In: Jonker, W. (Herausgeber): *Databases in Telecommunications II – Proc. VLDB 2001 Workshop on Databases in Telecommunications (DBTel 2001* , Rome, Italy, September 2001. Springer.

[MK94] Mansfield, G. und S. Kille: *RFC 1567: X.500 Directory Monitoring MIB*. RFC, IETF, Januar 1994.

[MLB05] Markovits, S., M. Lamm und R. Braun: *Information Modeling of Trouble: A Service Provider View*. In: *ConTEL 2005. Proceedings of the 8th International Conference on Telecommunications*, Seiten 471–478, Zagreb, Croatia, Juni 2005. IFIP/IEEE, INRIA Press.

[Ner01] Nerb, M.: *Customer Service Management als Basis für interorganisationales Dienstmanagement*. Dissertation, Technische Universität München, März 2001.

[Neu93] Neumair, B.: *Objektorientierte Modellierung von Kommunikationsressourcen für ein integriertes Performance Management*. Dissertation, Technische Universität München, Februar 1993.

[Off00] Office of Government Commerce Staff (OGC): *Service Support*. ISBN 0113300158, März 2000. 323 S.

[Off01] Office of Government Commerce Staff (OGC): *Service Delivery*. ISBN 0113300174, Mai 2001. 300 S.

[Off02] Office of Government Commerce Staff (OGC): *ICT Infrastrucure Management*. ISBN 0113308655, Oktober 2002. 297 S.

[OMG97] *CORBAservices - Event Management Service*. Document 97-12-11, Object Management Group, Dezember 1997.

[OMG99] *CORBA 2.3 - chapter 3 - IDL Syntax and Semantics*. OMG Specification formal/99-07-07, Object Management Group, Juli 1999.

[OMG01] *Meta Object Facility (MOF) v1.3.1*. OMG Specification formal/01-11-02, Object Management Group, November 2001.

[OMG04] *UML 2.0 Superstructure Specification*. Technischer Bericht ptc/04-10-02, Object Management Group, Oktober 2004.

[OWL] OWL-S Coalition: *OWL-S 1.1 release*. http://www.daml.org/services/daml-s/0.9/.

[PS03] Pras, A. und J. Schoenwaelder: *On the Difference between Information Models and Data Models*. Technischer Bericht, IETF, 2003. RFC 3444.

[Rad03] Radisic, I.: *Ein prozessorientierter, policy–basierter Ansatz für ein integriertes, dienstorientiertes Abrechnungsmanagement*. Dissertation, Ludwig–Maximilians–Universität München, Februar 2003.

[RM90] Rose, M.T. und K. McCloghrie: *RFC 1155: Structure and identification of management information for TCP/IP-based internets*. RFC, IETF, Mai 1990.

[Roe05] Roelle, H.: *Eine dienstorientierte Methodik zur Koppelung von Netz–QoS–Architekturen*. Dissertation, Ludwig–Maximilians–Universität München, Juli 2005.

[RS06] Racz, P. und B. Stiller: *A Service Model and Architecture in Support of IP Service Accounting*. In: *Proceedings of the 2006 IEEE/IFIP Network Operations and Management Symposium (NOMS)* , Band 2006, Vancouver, Canada, April 2006. IEEE/IFIP.

[Sai05] Sailer, M.: *Towards a Service Management Information Base*. In: *Proceedings of the IBM PhD Student Symposium at the 3th International Conference on Service–Oriented Computing (ICSOC05)*, Band 2005 der Reihe *RC23826*, Amsterdam, Netherlands, Dezember 2005. IBM Internal Report.

[Sch01] Schmidt, H.: *Entwurf von Service Level Agreements auf der Basis von Dienstprozessen*. Dissertation, Ludwig–Maximilians–Universität München, Juli 2001.

[Sch07] Schmitz: *Imact Analysis*. Dissertation, Ludwig–Maximilians–Universität München, 2007. to appear.

[SM01] Sailer, M. und Morciniec Michal: *Monitoring and execution for contract compliance*. Technischer Bericht HPL- 2001–261, Hewlett–Packard Research Laboratories, Bristol, UK, jul 2001. http://www.hpl.hp.com/techreports/2001/HPL-2001-261.html.

[Str02] Strassner, J.: *DEN-ng: achieving business-driven network management*. In: Stadler, R. und M. Ulema (Herausgeber): *Proceedings of the 8th International IFIP/IEEE Network Operations and Management Symposium (NOMS 2002)*, Florence, Italy, April 2002. IFIP/IEEE, IEEE Publishing.

[Str04] Strassner, John C.: *Policy-Based Network Management*. Morgan Kaufmann Publishers, ISBN 1-55860-859-1, 1 Auflage, 2004.

[SWMFR02] Schott, Julie, Andrea Westerinen, J. P. Martin-Flatin und Peter Rivera: *Common Information vs.*

Information Overload. In: Stadler, R. und M. Ulema (Herausgeber): *Proceedings of the 8th International IFIP/IEEE Network Operations and Management Symposium (NOMS 2002)*, Florence, Italy, April 2002. IFIP/IEEE, IEEE Publishing.

[VVB+03] Vergara, Jorge E. López, Víctor A. Villagrá, Julio Berrocal, Juan I. Asensio und Roney Pignaton: *Semantic Management: Application of Ontologies for the Integration of Management Information Models*. In: *Integrated Network Management VII, Managing It All, IFIP/IEEE Eighth International Symposium on Integrated Network Management (IM 2003)*, Band 246, Colorado Springs, USA, März 2003. IFIP/IEEE, Kluwer Academic Publishers.

[WSS+01] Westerinen, A., J. Schnizlein, J. Strassner, M. Scherling, B. Quinn, S. Herzog, A. Huynh, M. Carlson, J. Perry und S. Waldbusser: *RFC 3198: Terminology for Policy-Based Management*. RFC, IETF, November 2001.

[ZSCBB04] Zakaria, Benahmed Daho, Noémie Simoni, Michel Chevanne und Stephane Betge-Brezetz: *An information model for service and network management integration: from needs towards solutions*. In: *Managing Next Generation Convergence Networks and Services: Proceedings of the 2004 IEEE/IFIP Network Operations and Management Symposium (NOMS)*, Band 2004, Seoul, Korea, April 2004. IEEE/IFIP.

Index

A

Abhängigkeit **182**
Abhängigkeiten 115, 134
  Modellierung 4
Abhängigkeitsbeziehungen 31
Abrechnungsmanagement 121
Analysephase 7, 8, 85, 147
Anforderungsanalyse 7
Anforderungskatalog 27, 35, 147, 224, 247
Attribut
  SE Availability 204
  Service Attach Time 202
  Service Transactions 200
  Operational Status . . 193, 216
  SE HighestErrorRate 196
  NumberOf Incident Records 198
  Objectives Met 206
    Average End to End Service Establishment Time 203
    Average Inter SE Data Speed 204
    Average Inter SE Data Transfer 205
    Average Service Attach Time 202
    Average Service Data Transfer 201
    Average Service Requests 201
    Average Service Transaction Speed 200
    Average Service Transactions 200
    End to End Service Establishment Time 202
    Faulty Transaction Rate 193
    Faulty Transactions 193
    Incident WithNo Problem Found 198
    Incidents Fixedin Resolution Time 198

Inter SE Connectivity Status 196
Inter SE Data Transfer . 205
Inter SE Data Transfer Speed 204
Inter SE Link Utilization 197
Inter SE Transfer Errors 196
OLA Objectives Met . . . 206
Repeated Incidents 198
Resulting Penalties 207
SE Average Throughput 204
SE Average Utilization . 196
SE AverageErrorRate . . . 196
SE Highest Utilization . . 196
SE Lowest Availability . . 204
SE Throughput 204
Service Attach Count . . . 202
Service Attach Fault Rate 194
Service Attach Faults . . . 194
Service Attach Success Rate 202
Service Availability 201
Service Connection Failure Rate 194
Service Data Transfer . . 201
Service Requests 200
Service Transaction Speed 200
Service Transaction Success Rate 194
Severity Of Unmet Objectives 206
UC Objectives Met 206
Attributskategorie
ServiceFault . . 193, 196, 198
ServicePerformance 200, 202, 204, 206
auto discovery 253

B

Bewertung
Überwachungsphase 245
Analysephase 148
Gesamt 245
Nutzungsphase 246
Spezifikationsphase 226
Bottom Up Vorgehen 94

C

CIM 238
Implementierung 40
Managed Object Format . 38
Modellebenen 38
CIMOM 40, 241
CMDB 123
CORBA 59
CSM 105, 164
Customer Service Management *siehe* CSM
Customer-based SLA 174

D

Datacenter 254
Datenmodell 15
Datenmodellierungssprache . . . 16
Design-Pattern 155
Dienstagent 228
Dienstbegriff 97
Dienstgüte . . . . . *siehe* QoS
Dienstimplementierung 103
Dienstkatalog 138
Dienstkategorie 163
Dienstkonfiguration 254
Dienstlebenszyklus 28, 105
Beendigung 112

Bereitstellung 108
Betrieb 110
Planung 106
Vereinbarung 108
Dienstmanagement 2
Dienstmanagementimplementierung 104
Dienstorientierungsaspekte . . . 95
Dienstspezifikation 115
Diensttemplate 253
Diensttopologie 132
Dienstzugangspunkt . . *siehe* SAP
DMS . *siehe* Datenmodellierungssprache

E

Enthaltenseinshierarchie 52
Entität
Beispielentität 156
Beschreibungsschema . . . 156
InterServiceDependency 184
ManagedServiceElement 158
Prozessbezug 157
QoSService 171
Resource 182
Service 161
ServiceAccessPoint 166
ServiceCustomer 181
ServiceLevelObjective . . 175
ServiceLevelPenalty 176
ServiceQuality 170
ServiceQualitySpecification 169
ServiceResourceDependency 185
ServiceSpecification 160
ServiceUser 180
SLA 174
Synonym 156
Erstlösungsquote 129
eTOM 43
Event-Korrelation 133, 186

F

FCAPS 121
Fehlermanagement 121
Funktionale Eskalation 128

G

Geschäftsziele 1, 3
GRID 23, 82, 207, 215

I

IDL 59
Impact 127, 186
Incident 125
Incident-Management 125
Informationsabbildung 80
Informationsanalyse 93
Informationsanforderungen . . 104
Beendigung 113
Bereitstellung 109
Betrieb 112
Planung 107
Vereinbarung 108
Informationsbedarf
von Dienstbetreibern 4
Informationsmodell 11, 14
CIM 3, 38, 157, 238
IETF 49
Integration 34
ISO 3
SID 3, 44, 157
Informationsverfeinerung 235
Integrationsaspekte 237
Internet-MIB 50, 51

Interoganisationale Aspekte . . 31
IT-Business-Alignment 1
ITIL 53, 85, 120

K

Key Performance Indicator *siehe* KPI
Konfigurationsmanagement . . 121
Konkrete Dienstmanagementinformation 29
Konkrete Managementinformation 16
Konsistenzsicherung 245
Kontrollfluss 255
Koppeldienste 114
KPI 123, 135

L

Lastenheft 160
Laufender Betrieb 245
Leibniz Rechenzentrum . . . . *siehe* LRZ
Leistungsindikator 123
Leistungsmanagement 121
LRZ 18

M

Münchener Wissenschaftsnetz *siehe* MWN
Managed Object Format 38
Managementagenten 34
Managementarchitektur 254
Managementinformationsbasis *siehe* MIB
Managementobjekt . . . . *siehe* MO
Managementobjektklasse . . . *siehe* MOC
Managementprozesse 119
Managementsicht 81
Meta-Modellebene 254
MIB 2, 12–13
MNM Dienstmodell 97, 99
MO **12–14**, 114
MOC 13
Monitoring 4, 26, 141
Multi-level SLA 174
MWN 18

N

NGOSS 42
Nutzungsfunktionalität 102
Nutzungsphase 8, 90

O

Objektkataloge 17
Ontologie 254
Operational Agreement 138
Operational Level Agreement 174
Organisationseinheiten 20
Organisationsgrenzen 113
Outsourcing 113
Outtasking 113
OWL-S 59

P

Peering 113
Problem Record 128
Prozessüberwachung 123
Prozessausführung 122
Prozessinhaber 123
Prozessmodell 121
Prozessrahmenwerk
  eTOM 43
  ITIL 53

Q

QoS 3, 102, 167

QoS-Parameter 139
QoS-Paramter 168
Quality of Service . . . . . *siehe* QoS
Quality-of-Device 140

R

Reporting 129
Rollenkonzept 97, 114, 177

S

SAP 103, 105, 163
Service Information Specification Language . . *siehe* SISL
Service Level Agreement . . . *siehe* SLA, 22
Service Monitoring Architecture *siehe* SMONA
Service request 125
Service-based SLA 174
Service-MIB
  Übersicht 79
  Analysephase 85
  Anpassungen 245
  Basismodell 153, 186
  Bewertung 245
  Designrichtlinien . . . 100, 104
  Integration 238
  Methodik 83
  Nutzungsphase 90
  Optimierung 245
  SISL 154
  Spezifikationsphase 85
  Ueberwachungsphase 88
Service-Qualität 125
Sicherheitsmanagement 121
SID
  Aufbau 45
  Dienstmodell 48
Single-Point-Of-Contact 126
SISL 154, **207**
  Abbildungsvorschriften . 212
  Aggregationsoperationen 209
  Benachrichtigungsoptionen 209, 213
  Komponentenparameter 213
  Schema 215
  Schwellwerte 209
SLA . . . 2, 103, 122, 132, 139, 173
SLA-Template 173
SMONA 229
  Adapter Configurator . . . 233
  RichEvent Composer . . . 233
  Service Attribute Factory 233
SMONA Schicht
  Applikationsschicht 234
  Integration 233
  Plattformspezifisch 231
  Plattformunabhängige . . 232
  Ressource 231
Spezifikationsphase . . . 8, 85, **153**
Standard
  DMTF 38
  IETF 49
  OGC 53
  tmforum 42
Strafzahlung 141
Szenario
  Grid-Computing 23
  Web-Hosting 18
Szenariospezifische Anpassung 90

T

Technologieunabhängigkeit . . 120
Top Down Vorgehen 94

U

Ueberwachungsphase 8, 88
Underpinning Contract . 138, 174
Urgency 127

V

Verfügbarkeit 88, 125
Virtueller Webserver 19

W

WBEM 40, 91, 242
Web service 60
    composition 59
    WSDL 59
Web-Hosting Dienst 20
Wiederbenutzungsaspekt 229
Wiederherstellungszeit 132
Wissendatenbank 128
Work around 125, 134

X

xmlCIM 241