Proceedings of the *GI-Fachtagung Kommunikation in Verteilten Systemen (KiVS) 97*,
Braunschweig, Germany, February 1997

Interoperable Architekturen als Basis eines integrierten Managements

Alexander Keller[*] *Bernhard Neumair*
Münchner Netzmanagement Team
Institut für Informatik, Universität München
Oettingenstrasse 67, 80538 München, Germany
E-Mail: {keller|neumair}@informatik.uni-muenchen.de

Zusammenfassung

Managementarchitekturen stellen die Basis für integriertes Management von heterogenen verteilten Systemen und Kommunikationsnetzen dar. Derzeit existieren mehrere standardisierte Architekturen, die teilweise in Konkurrenz zueinander stehen. Da sich vermutlich nicht eine einzige Architektur für alle Einsatzbereiche durchsetzen wird, wird es wohl zu einem Nebeneinander von verschiedenen Ansätzen kommen. Für wirklich integriertes Management müssen also Übergänge geschaffen werden, die eine nahtlose Kombination der Architekturen erlauben. Der Beitrag wird die grundlegenden Alternativen für solche Übergänge darstellen und anschließend eine Analyse der prinzipiellen Unterschiede der Architekturen skizzieren. Auf dieser Basis werden dann konkrete Projekte vorgestellt, die zum einen eine Integration von CORBA-konformen Managementanwendungen in eine vorhandene, CMIP- bzw. SNMP-basierte Plattform und zum anderen Management-Gateways zwischen verschiedenen Architekturen prototypisch realisiert haben.

1 Einführung und Motivation

Die zunehmende Komplexität und Heterogenität verteilter Systeme stellt für ihre Betreiber eine große Herausforderung dar. Heutige umfangreiche DV-Infrastrukturen können nur noch dann mit vertretbarem Aufwand administriert werden, wenn man integrierte Managementlösungen einsetzt. Solche Lösungen auf der Basis standardisierter Managementarchitekturen ([3, 14]) sollen die Betreiber dabei unterstützen, eine optimale Versorgung der Benutzer mit akzeptablem Aufwand sicherzustellen.

In letzter Zeit trat auf dem Weg zum integrierten Management allerdings eine zusätzliche Komplikation auf: neben proprietären Ansätzen wurden *mehrere* Managementarchitekturen standardisiert, die teilweise in Konkurrenz zueinander stehen. So ist das bekannte Internet-Management (*SNMP-Management*) im Bereich des LAN-Managements weit verbreitet, während im Bereich der Telekommunikationsnetze vor allem auf das OSI-Management gesetzt wird. Mit dem *Desktop Management Interface* (DMI) der DMTF wird derzeit versucht, für das Management von Endsystemen wie PCs oder Workstations einen weiteren Standard zu etablieren. Neuerdings werden auch die Arbeiten der OMG (*Object Management Architecture - OMA, CORBA*) für die Bereiche des Endsystem- und Anwendungsmanagements zunehmend beachtet. Dem OMG-Ansatz werden heute gute Chancen auf große Verbreitung für die Entwicklung verteilter Anwendungen eingeräumt. Bewahrheitet sich dies, wird er auch für *integriertes* Management sehr wichtig werden, obwohl er nicht wie die anderen Architekturen speziell auf das Management einer DV-Infrastruktur ausgerichtet, sondern prinzipiell für alle verteilten Anwendungen gedacht ist.

In verteilten und heterogenen Systemumgebungen ist das Management natürlich eine verteilte Anwendung, deren Komponenten von verschiedenen Herstellern stammen. Zentraler Bestandteil einer Managementarchitektur ist damit die Definition eines Beschreibungsrahmens für die zu Managementzwecken ausgetauschte Information. Diese Information wird heute meist objektorientiert modelliert. Der Informationsaustausch findet also z.B. durch Lesen oder Setzen von Attributen sogenannter Managementobjekte oder durch Anwendung von Methoden auf diesen Objekten statt. Je detaillierter diese

[*]Die Arbeiten des Autors wurden gefördert durch das IBM European Networking Center, Heidelberg

Information festgelegt ist, desto effizienter ist Management auf der Basis der jeweiligen Architektur möglich. Weiterhin ist offensichtlich ein Kommunikationsprotokoll für diesen Austausch – ein sogenanntes Managementprotokoll – festzulegen.

Vermutlich wird sich nicht eine einzige Managementarchitektur durchsetzen; es wird wohl zu einem Nebeneinander von verschiedenen Architekturen kommen. Wirklich integriertes Management setzt also voraus, daß Übergänge geschaffen werden, die eine nahtlose Kombination der Architekturen erlauben (siehe auch [1, 9]). Prinzipiell gibt es hier *3 Alternativen*:

- „*Management-Gateway*“: Der Übergang wird durch ein Zwischensystem realisiert, das sowohl eine Übersetzung der Managementinformation als auch die Umsetzung der Managementprotokolle vornimmt. Zuerst müssen also die verschiedenen Spezifikationssprachen, in denen die Managementobjekte beschrieben sind, algorithmisch ineinander überführt werden. Besonders interessant ist die Abbildung der Sprachen des OSI- und des Internet-Managements auf den OMG-Ansatz, da in den beiden erstgenannten Fällen umfangreiche Objektspezifikationen speziell für Managementzwecke existieren, die für CORBA-basiertes Management übernommen werden können. Dieser statische Aspekt der Kooperation von Managementarchitekturen, die sogenannte *Specification Translation*, wird mit Hilfe von Spezifikations-Compilern behandelt. Zur Laufzeit müssen dann noch die Protokolle bzw. deren Elemente umgesetzt werden, was als *Interaction Translation* bezeichnet wird. Abb. 1 zeigt ein mögliches Szenario.

- „*Multiarchitekturelle Plattform*“: Hier erfolgt der Übergang innerhalb der Managementplattform, also des managenden Systems, das damit mehrere oder alle „Management-Sprachen spricht“. Damit ist auch keine direkte Umsetzung der Protokolle notwendig, sondern jeweils nur eine Abbildung auf das entsprechende Kommunikations-API der Plattform. Häufig findet die Übersetzung der Managementinformation nicht in der Infrastruktur der Plattform statt, sondern muß in den Anwendungen durchgeführt werden. Damit schlagen dann die Unterschiede der Architekturen bis in die Anwendungen durch; es bleibt häufig bei einem bloßen Nebeneinander der Architekturen unter einer gemeinsamen Oberfläche.

- „*Multiarchitektureller Agent*“: In diesem Fall erfolgen die notwendigen Abbildungen im Agenten, also im zu administrierenden System; hier „spricht“ der Agent mehrere oder alle „Management-Sprachen“ (siehe auch [11, 6]). Bei dieser Alternative fällt allerdings der höchste (Implementierungs-)Aufwand an, da naturgemäß die Anzahl an Agenten die der Managementsysteme bei weitem übersteigt. Insbesondere für einfache Komponenten ist eine solche Mehrsprachigkeit nicht sinnvoll realisierbar. Für den Betreiber einer DV-Infrastruktur ist dieser Ansatz irrelevant, da der Quelltext des Agenten vorliegen muß. Dies ist jedoch in der Regel nicht der Fall. Das Papier wird sich deshalb im folgenden auf die ersten beiden Alternativen beschränken.

Ziel ist bei allen Varianten, die Existenz verschiedener Managementarchitekturen für Benutzer, also die Betreiber einer DV-Infrastruktur, und die Managementanwendungen möglichst weitgehend zu verschatten. Man möchte dem Benutzer eine möglichst einheitliche Sicht auf alle zu administrierenden Systeme bieten und die Implementierung der Anwendungen effizient gestalten.

Im weiteren werden in Abschnitt 2 skizzenhaft die Managementarchitekturen verglichen und die grundsätzlichen Möglichkeiten für Übergänge gezeigt. Anschließend werden konkrete Projekte vorgestellt, die zum einen prototypisch Teile einer multiarchitekturellen Plattform implementiert (Abschnitt 3) und zum anderen Gateways zwischen CORBA und SNMP bzw. zwischen CMIP und SNMP (Abschnitt 4) realisiert haben.

2 Vergleich der Managementarchitekturen

Der folgende Vergleich der verschiedenen, heute definierten Managementarchitekturen (siehe auch ([5, 4]) soll die Basis legen für die Beschreibung der möglichen Architekturübergänge. Der Schwerpunkt wird dabei nicht auf einem Vergleich bekannter Architekturen wie dem OSI-Management oder dem Internet-Management liegen, sondern auf einer Gegenüberstellung mit dem neueren CORBA-Ansatz der OMG als Architektur für das Netz- und Systemmanagement.

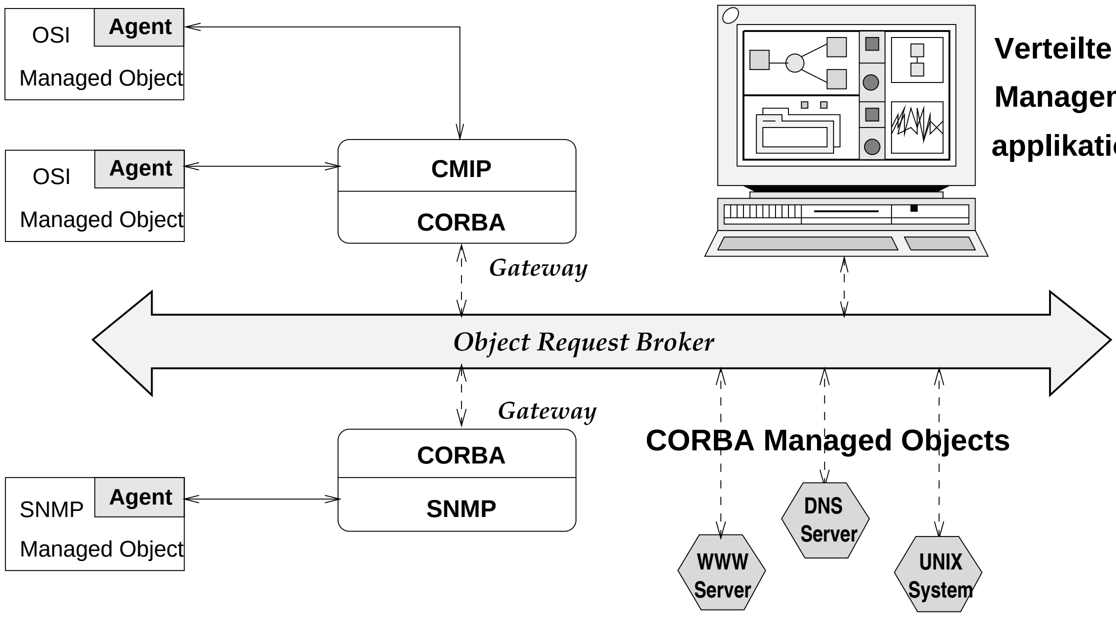

Abbildung 1: Interoperabilitätsszenario für Managementarchitekturen

2.1 Ziel und Inhalt einer Managementarchitektur

Wie in jeder Architektur für verteilte Systeme müssen auch für die verteilte Anwendung *Management* die grundlegenden Interaktionsprinzipien und die ausgetauschte Information festgelegt werden. Managementarchitekturen müssen deshalb u.a. die folgenden Bereiche abdecken:

- *Zu Managementzwecken auszutauschende Information:* Wie in der Einleitung erwähnt, wird derzeit zur Spezifikation der Information der objektorientierte Ansatz favorisiert. Es muß also ein Beschreibungsrahmen für die zu definierenden Managementobjekte festgelegt werden. Dabei ist grundsätzlich zu entscheiden, ob und wie weit dieser Rahmen bereits speziell auf das Management oder sogar auf spezifische Managementbereiche zugeschnitten wird oder eher allgemein gehalten werden soll (Abschnitt 2.3).

- *Kommunikationsprotokoll:* Bei der Festlegung eines Managementprotokolls, die ja in verteilter Umgebung offensichtlich nötig ist, ist grundsätzlich die Entscheidung zu treffen, ob für die beim Management auftretenden Kommunikationsvorgänge (Abfragen von Information über Ressourcen, Austausch von Steuerinformation, (asynchrone) Ereignismeldungen) ein komplett neues Protokoll definiert werden soll, für das dann natürlich eigene Protokollinstanzen zu implementieren sind. Alternativ dazu könnten auch vorhandene Mechanismen und Protokolle (z.B. RPCs etc.) angepaßt werden, indem z.B. die ausgetauschte (Protokoll-)Information verfeinert und mit einer speziellen Semantik versehen wird. Im RPC-Beispiel würde sich dies z.B. auf die aufrufbaren Prozeduren und deren Semantik beziehen (Abschnitt 2.4).

- *Infrastruktur-Dienste:* Neben den reinen Kommunikationsdiensten sind noch weitere Dienste zu definieren, die für deren effektive und effiziente Nutzung unbedingt notwendig sind. Darunter fallen z.B. Dienste für die Namensverwaltung und für das Ereignismanagement. Solche Dienstdefinitionen können, da sie bei hinreichend genauer Spezifikation ja in verschiedenen Systemen realisiert werden können, auch die Skalierbarkeit des Managements verbessern. Dies gilt z.B. für Überwachungsdienste, die bei Bedarf delegiert werden können (Abschnitt 2.5).

2.2 Grundsätzliche Architektur-Varianten

Die in der Einleitung erwähnten existierenden Architekturen unterscheiden sich grundsätzlich, was die im Abschnitt 2.1 skizzierten Kriterien und Entscheidungen anbelangt. Man kann sie einteilen in:

- *Managementarchitekturen im engeren Sinn: OSI-Management*, *Internet-Management* und *DMTF DMI* sind von vornherein speziell für Managementzwecke entwickelt worden. Es wurden jeweils praktisch von Grund auf ein eigenes Managementinformationsmodell, ein eigenes Protokoll nur für das Management und teilweise auch managementspezifische Infrastrukturdienste definiert.

- *Managementarchitekturen im weiteren Sinn:* Die *Object Management Architecture (OMA)* der OMG für verteilte objektorientierte Programmierung erlaubt in erster Linie ortsungebundene Kooperation von Objekten in verteilter Umgebung. Sie ist, im Gegensatz zu den bisher genannten Architekturen, nicht speziell auf das Management der DV-Infrastruktur ausgerichtet, sondern prinzipiell für alle verteilten Anwendungen gedacht. Neuerdings gewinnt sie an Bedeutung für das integrierte Management, da vor allem Produkte im Bereich des Managements von Endsystemen auf OMG-Technologie aufsetzen bzw. aufsetzen werden. Soll sie in heterogener Umgebung einsetzbar sein, müssen die allgemein gehaltenen Konzepte und Methoden spezialisiert und verfeinert werden. Man setzt dabei allerdings auf einem höheren Niveau auf als bei den vorigen Ansätzen. Die *X/Open Systems Management Working Group (XoTGsysMan)* sowie die *OMG Telecom Special Interest Group* befassen sich u.a. speziell mit Infrastrukturdiensten, die auf das Management zugeschnitten sind. Stützt man sich, wie hier vorgesehen, weiter als bei den Managementarchitekturen im engeren Sinn auf eine vorhandene Infrastruktur ab, bestehen zum einen natürlich Aussichten auf Einsparungen bei SW- und Installationskosten für den Anwender. Zum anderen können Entwickler von Managementsoftware dann Werkzeuge benutzen, die nicht ausschließlich für Managementapplikationen einsetzbar sind. Die Einarbeitungszeit, z.B. bezogen auf die Modellierung von Information, wird deutlich kürzer.

2.3 Ausgetauschte Information

Im Rahmen eines sogenannten Informationsmodells muß für jede Architektur festgelegt werden, wie Syntax und Semantik der auszutauschenden Managementinformation definiert werden. Grundsätzlich muß entschieden werden, ob aus Gründen der Interoperabilität die allgemein denkbaren Möglichkeiten objektorientierter Spezifikationen weiter einzuschränken und zu spezialisieren sind oder ob aus Gründen der Flexibilität des Ansatzes darauf verzichtet werden soll. Zusätzlich ist festzulegen, ob man es bei einem Beschreibungsrahmen belassen möchte oder auf dieser Basis dann zur Förderung der Interoperabilität bereits konkrete Information spezifiziert werden soll. Dabei können folgende Eigenschaften der Modelle unterschieden werden:

- *Struktur der Information:* Hierzu existieren die beiden folgenden grundsätzlich verschiedenen Alternativen.
  - Der OMG-Ansatz definiert (bisher) ausschließlich eine Schnittstellen-Beschreibungssprache, die *Interface Definition Language (IDL)*. Mit dieser C++-ähnlichen Sprache werden keinerlei weitere, managementspezifische Annahmen über die Struktur von Managementobjekten getroffen; es erfolgen keine weiteren Einschränkungen der denkbaren Objektwelt.
  - Die managementspezifischen Ansätze sind hier (vor allem aus Gründen der Interoperabilität) sehr viel restriktiver; es werden wesentliche Annahmen über die Struktur der Objekte gemacht. So legt das OSI-Informationsmodell z.B. fest, daß ein Managementobjekt durch Attribute, Aktionen und Notifikationen zu charakterisieren ist und gibt sehr detailliert sogenannte Templates vor. Im Internet-Management und bei DMI sind Managementobjekte nur einfache Variable oder Tabellen, die ebenfalls über spezielle Templates beschrieben sind.

- *Datentypen:* Wie bei anderen Spezifikationssprachen sind Basis-Datentypen festzulegen, die für die Charakterisierung der Objekte bzw. der Attribute verwendet werden können. Die managementspezifischen Ansätze geben hier teilweise bereits eine große Zahl an Typen vor, die für Managementzwecke nützlich sind. Beispiele hierfür sind Zähler, Schwellwerte, Pegel etc., die z.B. aus `integer`-Typen abgeleitet sind. Solche Festlegungen fehlen beim OMG-Ansatz völlig.

- *Semantik der Zugriffe:* Hier ist die Frage zu klären, welche Objektzugriffe die Architektur unterstützt. Die managementspezifischen Ansätze legen die Semantik möglicher Zugriffe von vornherein sehr detailliert fest. Die Basis dafür findet man in den obigen Restriktionen der Struktur von Objekten.

  Der OMG-Ansatz macht naturgemäß weniger Vorgaben an mögliche Objektzugriffe, was seinen Grund im allgemeiner gehaltenen Objektmodell hat. Um problemlose Interoperabilität in heterogener Umgebung für das Management sicherzustellen, sind dann noch weitergehende Festlegungen nötig.

- *Semantik der Objekte selbst:* Integriertes Management in heterogener Umgebung ist nur dann sinnvoll möglich, wenn auch an die Semantik der Objekte selbst Vorgaben gemacht werden. Einfache Beispiele hierzu sind die Semantik von Statusvariablen (`operational`, `up` etc.) und Statusübergängen (`restarting`, `rebooting` etc.) oder die Parameter von Ereignismeldungen (`coming up`, `going down` etc.).

  Derartige Vorgaben werden i.a. durch generische, nicht instantiierbare Objektklassen bzw. Interfaces realisiert, deren Eigenschaften weitervererbt werden können, was Einheitlichkeit und damit Interoperabilität fördert. In diesem Bereich ist das grundsätzliche Vorgehen beim OSI-Management und beim OMG-Ansatz sehr ähnlich: Im Rahmen bestimmter sogenannter *Systems Management Functions* bzw. *CORBA-Services* und *-Facilities* (siehe auch Abschnitt 2.5) werden nicht getrennt implementierbare Dienste bzw. Funktionen spezifiziert, sondern Vorgaben bezüglich der Semantik bestimmter Attribute, Methoden oder Object Interfaces gemacht. Diese sollen bei der Definition instantiierbarer Objektklassen bzw. Interfaces ererbt bzw. importiert werden.

  Internet-Management bzw. DMI liefern weniger Vorgaben. Hier baut man eher darauf, daß Informationsspezifizierer erkennen, wenn Gleiches schon einmal anderswo festgelegt wurde, und sich dann entsprechend daran ausrichten.

Insgesamt liefern die Definitionen der managementspezifischen Ansätze (vgl. Abschnitt 2.2) naturgemäß eine Interoperabilität des Managements in heterogener Umgebung, die beim OMG-Ansatz trotz der Arbeit von Gruppen wie der XoTGsysMan noch nicht gegeben ist. Andererseits hat dieser Ansatz den Vorteil, daß für ihn, weil er eben für viele andere verteilte Anwendungen ebenfalls einsetzbar ist, zukünftig eine größere Auswahl an Entwicklungs- bzw. Modellierungswerkzeugen als bei den anderen Architekturen zur Verfügung stehen wird.

Eine Möglichkeit zur Kombination der Vorteile beider Ansätze könnte darin bestehen, die weitergehenden Festlegungen der „klassischen" Managementarchitekturen in OMG-IDL zu übersetzen bzw. zu übernehmen (siehe auch Abschnitte 3.2 und 4.1). Damit könnte man einerseits die allgemein verwendbare Infrastruktur und die Entwicklungswerkzeuge für Managementzwecke nutzen, würde aber andererseits die bei den spezifischen Ansätzen erreichte Interoperabilität auch nicht aufgeben.

2.4 Kommunikationsprotokoll

Analog zu den Informationsmodellen stehen auch bei den Kommunikationsprotokollen mehrere hochspezialisierte Protokolle einem sehr allgemein gehaltenen Protokoll gegenüber. CMIP, das Managementprotokoll des OSI-Managements, und das bekannte SNMP sind extrem abgestimmt auf Managementinteraktionen und auf die jeweiligen Informationsmodelle. Die in Abschnitt 2.3 dargestellten möglichen Zugriffe auf Objekte werden direkt auf PDU-Strukturen abgebildet. Spezifika dieser Kommunikationsprotokolle und -dienste wie *Scoping & Filtering* und die protokolltechnische Behandlung des Allomorphiekonzepts bei CMIP bzw. der `GETNEXT`-Operator bei SNMP sind kaum auf andere Einsatzbereiche übertragbar.

Dem stehen die ORB-Dienste bzw. ein Inter-ORB-Protokoll gegenüber, das „nur" einen „objektorientierten RPC" definiert, also die Übertragung vollkommen beliebiger Methodenaufrufe und der zugehörigen Ergebnisse.

Genau wie bei den Informationsmodellen ist die Allgemeinheit des OMG-Protokolls aus Managementsicht hinderlich im Hinblick auf Interoperabilität in heterogener Umgebung. Übersetzt man aber die managementspezifischen Informationsdefinitionen aus der OSI- und Internet-Welt in IDL, schränkt man die mit ORB-Dienstaufrufen auf Managementobjekten aufrufbaren Methoden ein; die skizzierten Interoperabilitätsprobleme können nicht auftreten. Man kann dann relativ einfach vorhandene CORBA-Infrastrukturen für Managementkommunikation im engeren Sinn nutzen.

2.5 Infrastruktur-Dienste

Wie in Abschnitt 2.1 erwähnt, muß für jede Managementarchitektur eine Infrastruktur aus Basisdiensten definiert werden, die die Nutzung der Kommunikationsdienste erleichtern oder überhaupt erst möglich machen.

Führend in diesem Bereich ist das OSI-Management, das im Rahmen diverser Systems Management Functions umfangreiche derartige Basisdienste[1] definiert. Beispiele sind Dienste für die Handhabung asynchroner Ereignismeldungen, für die Schwellwertüberwachung oder für die Vorverarbeitung großer Datenmengen.

Auch die OMG hat in Form diverser *CORBA Services* solche Dienste definiert und erweitert diese laufend um neue. Die XoTGsysMan und die OMG Telecom SIG arbeiten nun daran, diese nicht managementspezifischen Dienste in Form sogenannter *CORBA Facilities* auf Managementzwecke abgestimmt zu verfeinern, wobei einige Konzepte aus dem OSI-Management hier wiederzufinden sind ([10]). Dies soll an zwei Beispielen erläutert werden:

- *Gruppieren von Objekten und Definition von Zielvorgaben:* Der *Managed Set Service* erlaubt es, zu Managementzwecken zusammengehörige Objekte zu gruppieren und diese Gruppen zu administrieren. Darunter fallen Tätigkeiten wie die Aufnahme und Streichung von Mitgliedern oder das Einholen von Informationen über Mitglieder einer bestimmten Gruppe. Mit dem *Policy Management Service* können dann Zielvorgaben (*Policies*) zu solchen Objektgruppen definiert und ihre Einhaltung überwacht werden. Er soll Administratoren dabei helfen, das Verhalten eines Managementsystems ihrer spezifischen Umgebung anzupassen. Diese Dienste sind weitgehend vergleichbar mit der *Management Domain and Management Policy Management Function* (ISO 10164-19) des OSI-Managements. Im Internet-Management sind keine derartigen Funktionen vordefiniert; Domains und Policies sind Sache der Anwendung.

- *Handhabung asynchroner Ereignismeldungen:* Das OSI-Management liefert mit der *Event Report Management Function* (ISO 10164-5) sehr flexible Mechanismen, mit denen eine Anwendung z.B. Ereignisse eines bestimmten Typs abonnieren oder Ereignismeldungen nach frei wählbaren Kriterien filtern lassen kann. Ähnliche Funktionalität innerhalb der OMA stellt der *Object Event Service* bereit, allerdings ohne (managementrelevante) Filtermöglichkeiten. Letzteres ist für Managementanwendungen ein schwerwiegender Nachteil. Die OMG Telecom SIG plant deshalb, hier mit dem *Notification Service* eine Erweiterung vorzunehmen. Einfache Funktionen zum Ereignismanagement, allerdings ebenfalls ohne Filtermöglichkeiten, finden sich im Rahmen der *Remote Network Monitoring MIB* (RFC 1757) auch im Internet-Management.

Managementdienste wie z.B. die entfernte Überwachung von Schwellwerten, zu denen im OSI-Management flexible und komplexe Mechanismen existieren (*Metric Objects and Attributes*, ISO 10164-11), werden im Rahmen der OMA bisher nicht behandelt. Da diese Dienste aber alle mit Hilfe des Management-Informationsmodells definiert wurden, könnte man aus diesen Beschreibungen IDL-Schnittstellen generieren, evtl. vorhandene Implementierungen damit kapseln und so OSI-Managementfunktionen auch per CORBA verfügbar machen (siehe auch Abschnitt 3.2).

[1] In diesen SMFs strebt man also im Gegensatz zu den in Abschnitt 2.3 erwähnten SMFs nicht die Verfeinerung der Semantik von Managementobjekten, sondern die Definition abgesetzt implementierbarer Dienste an.

2.6 Fazit

CORBA als gemeinsame Infrastruktur für viele Anwendungen incl. Management ist aus heutiger Sicht sehr zukunftsträchtig. Noch fehlen allerdings viele Festlegungen, die speziell für die verteilte Anwendung *Management* unbedingt notwendig sind, obwohl bereits CORBA-basierte Managementprodukte wie z.B. IBM TME 10 am Markt verfügbar sind. Dies gilt nicht nur für die fehlende Definition vieler konkreter Managementobjekte, wie sie im OSI-Management oder im Internet-Management zu finden sind. Es gilt auch, wie in den vorigen Abschnitten ausgeführt, für die Verfeinerung und Spezialisierung der Semantik der Zugriffe auf Objekte, für das Managementprotokoll zwischen den verschiedenen Akteuren des Managements und für die Basisdienste der Infrastruktur.

Würde man viele managementspezifische Spezifikationen aus der Internet- und OSI-Welt in die CORBA-Welt übertragen, würde man nicht nur die Interoperabilität von CORBA-basierten Managementsystemen und Komponenten in heterogener Umgebung verbessern bzw. sicherstellen, sondern auch die Übergänge zu Managementkomponenten, die auf den „klassischen “ Managementarchitekturen basieren, ermöglichen bzw. erleichtern.

Dies könnte dann technisch mit den oben erwähnten und in Abschnitt 3 bzw. 4 noch genauer ausgeführten multiarchitekturellen Plattformen oder Management-Gateways realisiert werden. Man würde damit einerseits vorhandene Investitionen retten (z.B. die große Zahl an SNMP-Agenten), und andererseits gleichzeitig neue CORBA-Investitionen auch für das Management nutzen.

3 Multiarchitekturelle Plattform

Managementplattformen (wie z.B. *IBM TME 10 NetView for AIX* oder *HP OpenView*) haben sich seit geraumer Zeit als praktikabler Weg für integriertes Management etabliert und bilden den Kern heutiger Managementlösungen. Sie sind die Basis, auf der die eigentlichen Managementapplikationen (wie z.B. *Cisco Works*, eine Anwendung für das Router-Management) ablaufen. Typische Plattformfunktionalitäten sind die Kommunikation mit den zu managenden Ressourcen über ein standardisiertes Managementprotokoll, die Verwaltung der Datenbank, in der die Netztopologie sowie Informationen zu den Ressourcen gehalten werden, sowie die Bereitstellung einer graphischen Benutzerschnittstelle, die flexibel anpaßbar ist. Eine weitere grundlegende Eigenschaft von Plattformen ist die Fähigkeit, asynchrone Ereignismeldungen von den Agenten entgegenzunehmen und zu verarbeiten. Wir zeigen in den folgenden Abschnitten am Beispiel von *IBM TME 10 NetView for AIX*, wie eine am Markt erhältliche SNMP-Managementplattform um Fähigkeiten für das Ereignis– und Topologiemanagement CORBA-konformer Agenten erweitert werden kann, was einen wichtigen Schritt auf dem Weg zu einer multiarchitekturellen Plattform darstellt.

Als CORBA-Entwicklungssystem kam das *IBM SOMobjects Developer Toolkit* zum Einsatz; es umfaßt einen IDL-Compiler, die CORBA-Laufzeitumgebung sowie mehrere zu den OMG-Standards konforme Dienste [15].

3.1 Erweiterung von *IBM TME 10 NetView* zu einer multiarchitekturellen Plattform

Sollen Plattformen als Basis für die Integration unterschiedlicher Managementarchitekturen verwendet werden, ist es naheliegend, unmittelbar an der Kommunikationsschnittstelle der Plattform anzusetzen, um so die Heterogenität verschiedener Managementprotokolle so weit „unten“ wie möglich abzuhandeln. Ein typischer Vertreter dieses Gedankens ist XMP/XOM (X/Open Management Protocol, X/Open OSI-Abstract-Data Manipulation), dessen Zielsetzung darin besteht, eine einheitliche Programmierschnittstelle für die Managementprotokolle CMIP und SNMP bzw. die Darstellung von ASN.1-Datentypen in C bereitzustellen, um einheitliches Management unabhängig von der verwendeten Managementarchitektur zu erlauben. Dieser Versuch der Vereinheitlichung muß jedoch aus heutiger Sicht als gescheitert betrachtet werden, da sich die Heterogenität der Managementprotokolle nicht verbergen ließ. Das sehr komplexe XMP wird daher in heutigen Produkten (wie zum Beispiel der *IBM NetView TMN Support*

Facility, einem auf NetView basierenden OSI-Managementsystem) ausschließlich als CMIP-API[2] eingesetzt; SNMP-konforme Ressourcen werden in bestehenden Produkten über ein separates einfaches SNMP-API angesteuert.

Wir haben uns für unsere Implementierung aus folgenden Gründen gegen eine auf XMP/XOM basierende Lösung entschieden:

- Der XMP-Stack wird bei NetView durch den sog. *Postmaster*-Dämon verwaltet. Eine CORBA-Ereignismeldung ist, im Gegensatz zu SNMP oder CMIP, keine Protokolldateneinheit mit festgelegter Struktur, sondern ein beliebiger Methodenaufruf auf einem *Consumer*-Objekt. Im vorliegenden Falle würde der Postmaster in der Rolle des Consumers (s.u.) agieren; es folgt, daß er um jede neu eingeführte Ereignismeldung erweitert werden müßte. Aufgrund des zu erwartenden Aufwands ist dies nicht praktikabel, da eine solche Erweiterung beim Einbringen jeder neuen CORBA-Ressource erfolgen müßte.

- Um solche Erweiterungen vornehmen zu können, hätte der Quellcode des *Postmaster*-Dämons und einiger anderer NetView-Komponenten vorliegen müssen; diese Forderung ist bei kommerziellen Produkten natürlich illusorisch.

Eine Alternative zu XMP ist der unmittelbare Zugriff auf die plattformspezifischen Infrastrukturdienste, wie z.B. Topologiedatenbank, Ereignisfilter oder Ressourcenverwaltung; jede dieser Komponenten verfügt über ein eigenes API. Die Nutzung dieser produktspezifischen und nicht standardisierten Funktionen impliziert allerdings den Verlust der Unabhängigkeit von bestehenden Plattformimplementierungen: die Lösung ist damit nur unter vermutlich hohem Aufwand auf andere Plattformen zu portieren. Die oben beschriebenen Argumente ließen jedoch keine andere Wahl zu.

Beide Alternativen haben durch die Heterogenität der Informationsmodelle die Einschränkung, daß auf Managementobjekten einer Protokolldomäne nur die durch das entsprechende Protokoll erlaubten Funktionen anwendbar sind; *Scoping* und *Filtering*, essentieller Bestandteil des OSI–Modells, kann nicht für das Management von SNMP–Ressourcen verwendet werden (vergleiche dazu Abschnitt 4.3). Auf multiarchitekturellen Plattformen wird daher bereits an oberster Stelle der Topologiehierarchie zwischen beiden Protokollwelten unterschieden. Das in der Einleitung beschriebene Problem des Fortlebens unterschiedlicher Architekturen unter einer gemeinsamen Oberfläche bleibt bei diesem Integrationsansatz bestehen (siehe auch [9]).

Der Schwerpunkt des Integrationskonzeptes liegt auf einem möglichst hohen Verfügbarkeitsgrad von Infrastruktur-Diensten, wie sie in Abschnitt 2.5 beschrieben wurden. Dies kann auf zweierlei Arten geschehen:

- Falls die *Dienste* der einen Architektur *keine Entsprechung* in der anderen betrachteten Architektur haben, kann man (siehe Abschnitt 2.6) durch die Überführung der Beschreibungen dieser Dienste in das entsprechende Informationsmodell die Managementfunktionalität einer Architektur von einer anderen Managementarchitektur aus nutzen. Damit könnte man z.B. in der OSI-Architektur existierende Mechanismen zur Schwellwertüberwachung von CORBA-Objekten aus verwenden. Dieser Weg hin zu Interoperabilität von Infrastruktur-Diensten wird in Abschnitt 4 ausführlicher beschrieben. Es sei hier nur kurz erwähnt, daß die Beschreibungen der Dienstschnittstellen algorithmisch in das Informationsmodell der Zielarchitektur überführt werden müssen (vgl. Abschnitt 2.3). Dort werden Proxy-Objekte erzeugt, die die Dienstaufrufe innerhalb der einen Architektur auf die Dienstimplementierungen der anderen transparent für den Dienstnutzer weiterleiten.

- Liegen *vergleichbare Infrastruktur-Dienste* in beiden Architekturen vor, müssen sie aufeinander abgebildet werden. Dies gilt für die in den folgenden Abschnitten betrachteten Applikationen, die die Topologie von CORBA-Agenten ermitteln (siehe Abschnitt 3.3) bzw. die von ihnen ausgesendeten CORBA-Events auf SNMP-Traps (und umgekehrt) abbilden (siehe Abschnitt 3.2). Hierbei werden ebenfalls Infrastruktur-Dienste genutzt, die die Managementplattform bereitstellt, um die ermittelten Informationen einer CORBA-konformen Managementapplikation (siehe auch

[2] In der Tat war XMP das bisher einzige standardisierte CMIP-API; ein C++ API für CMIP befindet sich momentan in der Standardisierungsphase (siehe dazu Abschnitt 4.3)

[16]) zur Weiterverarbeitung anzubieten. Die Plattform fungiert dabei als Brücke zwischen den CORBA-Agenten und einer CORBA-Managementapplikation (siehe Abbildung 2).

3.2 Nutzung von Infrastruktur-Diensten für ein integriertes Ereignismanagement

Die Verarbeitung asynchroner Ereignismeldungen durch eine Managementplattform geschieht i.w. in folgenden Schritten:

- Von den Agenten der überwachten Systeme werden (im Fehlerfalle oder bei Überschreitung vorher definierter Schwellwerte) Ereignismeldungen an die Managementplattform gesandt, die diese empfängt und nach festgelegten Regeln filtert. Hauptaufgabe der Filterung ist es also, aus den eingehenden Ereignis-Rohdaten Managementinformation zu gewinnen. Filterkriterien sind beispielsweise die Art der Ressource, der Typ und der Zeitpunkt der Ereignismeldung oder die Häufigkeit des Auftretens.

- Die gefilterte Managementinformation wird anschließend der Managementapplikation bzw. dem Administrator zur Verfügung gestellt. Die durch den Administrator ausgelösten Aktionen werden über die Plattform an die Ressourcen zur Ausführung übermittelt.

Die Aufgabe bestand nun darin, die vorhandenen NetView-Dienste für das Management von Ereignissen so zu erweitern, daß sowohl SNMP-Traps als auch CORBA-Events von einer zentralen Stelle empfangen werden können. Unser Ansatz basiert auf dem CORBA Event Service [13], der Mittel für die asynchrone Kommunikation zwischen verteilten Objekten spezifiziert. Hierbei wird zwischen Objekten, die Ereignisse erzeugen (sog. *Supplier*) und empfangenden Objekten (sog. *Consumer*) unterschieden. Die Kommunikation zwischen diesen Objektarten erfolgt durch Aufruf einer Methode des Consumers durch den Supplier; zur Entkopplung der beiden werden sogenannte *Event Channels* eingesetzt, die ihrerseits sämtliche Ereignisse, die von Suppliern kommen, an diejenigen Consumer weiterleiten, die sich für die betreffenden Events registriert haben. Man erhält somit die Möglichkeit, durch „Multiplexing“ ein Ereignis mehreren Consumern bzw. die Ereignisse sämtlicher Supplier einem Consumer zuzustellen. Letzterer Fall spiegelt unsere Anwendung der Event Channels wider.

Da NetView selbst nicht in der Lage ist, CORBA-Events zu verarbeiten, mußten diese im *ersten Schritt* in SNMP-Traps umgewandelt werden. Es wurde also ein Event-Gateway implementiert, das die NetView-Ereignisdienste kapselt[3]. Nachdem die Ereignismeldungen empfangen wurden, können sie nun ebenso wie SNMP-Traps durch die NetView-Dienste gefiltert, protokolliert und an der graphischen Benutzerschnittstelle angezeigt werden. Es hat sich gezeigt, daß durch die Abstützung auf vorhandene Plattformdienste bereits mit geringem Aufwand gute Ergebnisse erzielt werden konnten.

Der *zweite Schritt* bestand darin, einer CORBA-Managementapplikation die bereits von NetView gefilterten Ereignisse zuzustellen, um diese geeignet weiterverarbeiten zu können. Hierbei agiert nun die Plattform als Event-Supplier und die CORBA-Applikation als Consumer, die wiederum durch Event Channels entkoppelt sind. Das entsprechende Objekt, das den NetView-Dienst kapselt, wurde in Anlehnung an die OSI Event Report Management Function `EFD_Supplier` genannt, da seine Funktionalität identisch zu der eines OSI *Event Forwarding Discriminator (EFD)* ist: Die Managementapplikation registriert sich beim `EFD_Supplier`-Objekt für den Empfang gefilterter Ereignismeldungen einer bestimmten Art durch das Setzen eines Attributs des `EFD_Suppliers` auf den gewünschten Filterstring. Als Filterstring können alle in NetView erlaubten Filterregeln verwendet werden. Für jede Filterart kann ein `EFD_Supplier`-Objekt nach Bedarf erzeugt werden, an dem beliebig viele Consumer über Event Channels angeschlossen sein können.

3.3 Verwaltung der Topologie von CORBA-Managementagenten

Die Topologieverwaltung überwachter Ressourcen zählt zu den Kernaufgaben von Managementplattformen. Sie gliedert sich in folgende Teilschritte: Information über vorhandene Systeme und deren momentanen Zustand (up, down, usw.) wird ermittelt und in der Plattformdatenbank abgespeichert

[3] Das Kapseln von nicht CORBA-konformen Modulen ist der gebräuchlichste Weg, um bestehenden Programmcode für neue objektorientierte Systeme zugreifbar zu machen.

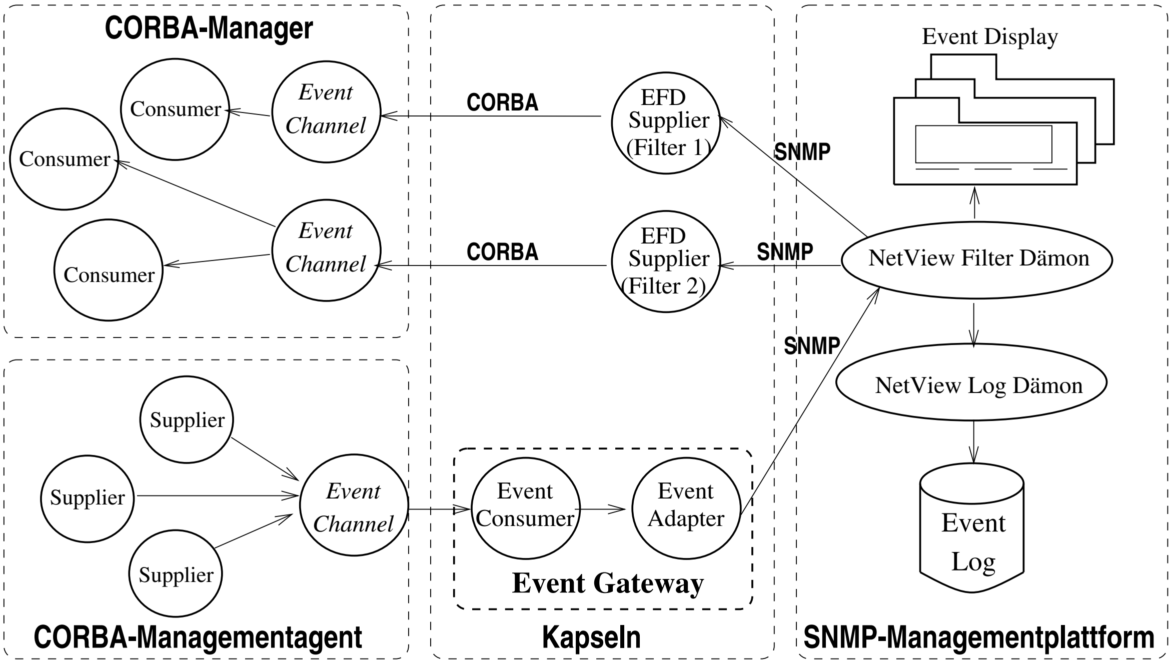

Abbildung 2: Verarbeitung von CORBA-Events bei einer multiarchitekturellen Plattform

(Discovery-Funktion). Ein übersichtliches Modell dieser Information (die Zustände der Ressourcen werden durch farbliche Kennzeichnung der entsprechenden Symbole dargestellt) wird anschließend an der graphischen Benutzeroberfläche der Plattform angezeigt. Um der Dynamik der überwachten Ressourcen gerecht zu werden, wiederholt sich dieser Prozeß in regelmäßigen Zeitabständen, der gewöhnlich durch die Plattform initiiert wird (*Polling*). Um der dadurch entstehenden hohen Netzlast zu begegnen, haben wir bei unserer Entwicklung einen ereignisgesteuerten Ansatz verfolgt, der nur dann Meldungen an die Plattform verschickt, wenn sich Änderungen an der Topologie ergeben haben.

Wir werden uns im folgenden auf die Beschreibung der Implementierung einer Discovery-Applikation für CORBA-Managementobjekte (dies können sowohl Anwendungen als auch Systeme sein) beschränken und aufzeigen, wie die Anbindung an den Topologiedienst von *NetView for AIX* erfolgte. Die Datenbank der Plattform wird somit zu einer integrierten Informationsbasis für SNMP– und CORBA–Managementobjekte.

Die Discovery-Applikation basiert auf dem NetView *Generalized Topology Manager (GTM)*, der abstrakte Datenobjekte zur Darstellung von Ressourcen bereitstellt und über ein API zugänglich ist. Dieses API wurde, analog zu unserem Vorgehen beim Ereignismanagement, durch CORBA-Interfaces gekapselt, um seine Nutzung durch die CORBA-basierte Discovery-Applikation überhaupt erst zu ermöglichen. Damit der GTM CORBA-Objekte geeignet darstellen kann, benötigt er von der Discovery-Applikation folgende Angaben: die insgesamt existierenden Applikationen oder Dienste, die Systeme (Hosts) und die Server (bzw. die Referenzen der Objektklassen, aus denen sie bestehen), die auf den Systemen laufen, sowie die Objektinstanzen, die auf einem Server aktiv sind. Diese Informationen wurden folgendermaßen ermittelt: das CORBA *Implementation Repository* enthält Angaben über verfügbare Systeme, auf ihnen laufende Server und eine Liste der durch letztere unterstützten Objektklassen. Die zu den jeweiligen Objektklassen existierenden Objektinstanzen werden durch den CORBA LifeCycle Service bereitgestellt. Über die Methoden der `Naming Context` Objekte (spezifiziert im CORBA Naming Service) können alle darin enthaltenen `Name Bindings` abgefragt werden; man kann damit die Namen und Objektreferenzen aller registrierten Objektinstanzen ermitteln.

Unsere Erfahrungen bei der Implementierung haben gezeigt, daß durch die Abstützung auf standardisierte bzw. von der Plattform zur Verfügung gestellte Infrastruktur-Dienste mit akzeptablem Aufwand eine brauchbare Lösung zur Topologieverwaltung erstellt werden konnte.

4 Management-Gateways

Management-Gateways, auch als *Proxies* bezeichnet, befinden sich an den Grenzen von Protokolldomänen und haben die Aufgabe, die Unterschiede zwischen den beteiligten Managementarchitekturen zu überbrücken. Es handelt sich dabei keinesfalls nur um reine Konverter für Protokolldateneinheiten; es muß insbesondere auch eine Abbildung von Managementinformation vorgenommen werden (siehe Abschnitt 1).

Im Gegensatz zu den beiden anderen in der Einleitung beschriebenen Integrationsansätzen verfolgt der Gateway-basierte Ansatz das Ziel, Koexistenz und Kooperation zwischen unterschiedlichen Architekturen zu gewährleisten, ohne Modifikationen an Plattformen oder Agentensystemen vornehmen zu müssen. Die Vorteile der Integration durch multiarchitekturelle Plattformen gelten somit auch für Management-Gateways. Man erreicht sogar einen höheren Grad an Flexibilität, da weder Abhängigkeiten zu konkreten Plattform-Implementierungen (siehe Abschnitt 3), noch Eingriffe auf Agentenseite (wie im Fall der multiarchitekturellen Agenten) erforderlich sind. Management-Gateways spielen eine Doppelrolle: aus der Sicht des Managers erscheinen sie als ein Agent, der sich in der gleichen Protokollwelt befindet; für Agenten sieht das Gateway wie ein Manager aus.

Wir werden im folgenden auf derzeit standardisierte, vielversprechende Ansätze zur Spezifikation von Management-Gateways eingehen und über Erfahrungen aus eigenen Projekten zur Implementierung von CORBA/SNMP und CMIP/SNMP–Gateways berichten. Wir beschränken uns auf diese Gateway-Prototypen, da sie wegen der weiten Verbreitung von SNMP-Agenten besonders wichtig sind. Die Prinzipien sind jedoch auch im Fall von Gateways zwischen CORBA und der OSI–Managementarchitektur gültig und ohne weiteres auf diese übertragbar.

4.1 Entwurfshilfen für Management-Gateways

Die Spezifizierung von Algorithmen zur Implementierung von Management-Gateways ist das Ziel eines von X/Open und dem NM Forum gemeinsam durchgeführten Projektes, das den Titel *Joint Inter Domain Management (JIDM)* trägt. Das Projekt behandelt beide in Abschnitt 1 beschriebenen Phasen des Übergangs zwischen unterschiedlichen Managementarchitekturen: die *Specification Translation* und die *Interaction Translation*. In der ersten Phase soll die Darstellung eines Objekts aus einer Architekturdomäne in einer anderen möglich gemacht werden. Das Ergebnis der zweiten Phase sind architekturtransparente Datenaustauschmechanismen zwischen Objekten, die sich in unterschiedlichen Domänen befinden. Während sich die Specification Translation gegenwärtig am Ende der Definitionsphase befindet (siehe [17]), haben die Arbeiten an der Interaction Translation erst vor kurzem begonnen. Zweifellos existieren Parallelen zur IIMC (*ISO–Internet Management Coexistence*) Initiative, jedoch ist das Spektrum von JIDM breiter angelegt: während IIMC auf die Interoperabilität zwischen OSI– und Internet–Management abzielte, wird hier die Kopplung der OSI– und Internet–Architekturen mit CORBA betrachtet; man will die Stärken einer Architektur für die jeweils anderen verfügbar machen, um zu einer umfassenden integrierten Managementlösung zu kommen.

In der Literatur unterscheidet man zwischen zwei Designvarianten für Management-Gateways ([1]): Ein Gateway ist *stateless* (zustandslos), wenn in ihm keine Werte von Attributinstanzen der jeweiligen Managementobjektklassen gespeichert werden. Andernfalls handelt es sich um ein *stateful* (zustandsbehaftetes) Gateway. Aus dieser Definition folgt unmittelbar, daß stateless Gateways per se nicht über Cachingmechanismen für Attribute verfügen. Sie stellen die einfacher zu implementierende Gateway-Variante dar, da sie die Daten, die sie von Managern oder Agenten erhalten, lediglich an die entsprechenden Stellen weiterleiten.[4] Wir haben uns für die beschriebenen CORBA/SNMP– und CMIP/SNMP–Gateways jeweils für die zustandlose Variante entschieden, wobei jedoch das Gateway Kenntnis über die SNMP-MIBs derjenigen Ressourcen haben muß, die über das Gateway angesteuert werden. Dies impliziert, daß diese MIBs zuerst in das Informationsmodell des jeweiligen Managers durch die oben beschriebenen JIDM– bzw. IIMC–Algorithmen überführt und anschließend dem Gateway bekanntgemacht werden müssen.

[4] Neben diesen Einteilungen gibt es natürlich noch weitere (feinere) Abstufungen. Dies hängt mit der grundlegenden Designentscheidung zusammen, welche Funktionalität zur Umsetzung der Informations- und Kommunikationsmodelle an welcher Stelle (Manager oder Gateway) angesiedelt wird.

4.2 CORBA/SNMP-Gateway

Das Gateway für den Übergang von der CORBA– in die SNMP–Managementarchitektur muß folgende Anforderungen erfüllen:

1. Darstellung der SNMP–MIBs für den Manager in dessen eigenem Informationsmodell
2. Modifikationen weder am Manager noch an den Agenten, d.h. *vollkommen transparenter* Übergang aus Sicht dieser Systeme (Das Gateway muß daher Proxy-Objekte für sämtliche SNMP–Agenten vorhalten, damit ein CORBA-Request eines Managers an ein Proxy-Objekt durch einen entsprechenden Reply beantwortet wird).
3. Operationen, die der Manager durch Requests an den Proxy-Objekten durchführt (lesen, schreiben, erzeugen, löschen), müssen durch das Gateway interpretiert und an die entsprechenden SNMP-Agenten weitergeleitet werden. Umgekehrt müssen von den SNMP-Agenten stammende asynchrone Ereignismeldungen (sog. *Traps*) in CORBA-Events umgesetzt werden.

Abbildung 3 gibt einen Überblick über das Gateway. Jedes Proxy-Objekt repräsentiert eine Instanz der

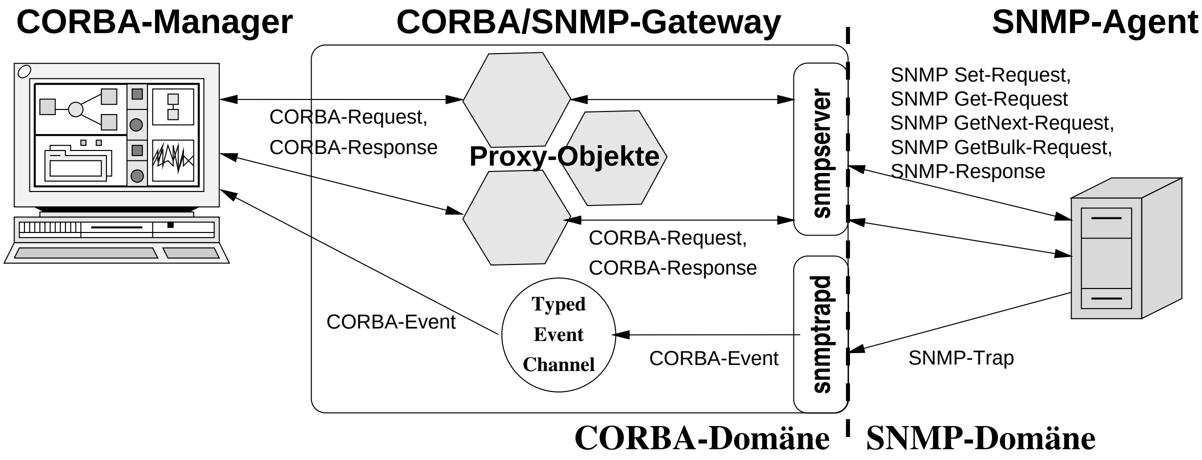

Abbildung 3: Ablauf der Kommunikation beim CORBA/SNMP-Gateway

nach dem JIDM–Algorithmus erzeugten Objektklasse, die SNMP-spezifische Informationen zu diesem Objekt bereithält (UDP-Socket, Object ID, Community String). Sobald ein Manager einen Request an ein Proxy-Objekt verschickt, ruft dieses eine entsprechende Methode auf dem `snmpserver`-Objekt unter Hinzufügen der SNMP-spezifischen Parameter auf. Das `snmpserver`-Objekt kann mit den mitgelieferten Parametern eine vollständige SNMP-PDU erzeugen, die anschließend durch den SNMP-Agenten beantwortet wird. Das Ergebnis wird nun als Response-Message vom `snmpserver`-Objekt zum aufrufenden Proxy-Objekt weitergeleitet, welches das Ergebnis nun seinerseits als Response zum Manager zurückschickt.

SNMP-Traps werden auf dem UDP-Port 162 des Gateways durch das `snmptrapd`-Objekt angenommen und an einen (für jede Art von Trap spezifischen) typisierten Event-Channel in Form eines CORBA-Requests weitergeleitet. Wie in Abschnitt 3.2 beschrieben, kann ein Event-Channel-Objekt den erhaltenen Event an diejenigen CORBA-Objekte (d.h. Manager) weiterleiten, die sich für die Zustellung solcher Ereignisse registriert haben.

Der Ansatz ist daher auch für die Verwendung mehrerer CORBA-Managementsysteme ausgelegt. Die Erfahrungen haben gezeigt, daß die gewählte Lösung unter Rückgriff auf bestehende Infrastruktur-Dienste elegant implementierbar ist ([7]); allerdings bleibt die Skalierbarkeit dieses Ansatzes durch das Vorhandensein der transformierten MIBs sämtlicher SNMP-Agenten im Gateway fraglich. Es ging uns jedoch primär darum, die prinzipielle Machbarkeit eines solchen Gateways aufzuzeigen.

4.3 CMIP/SNMP-Gateway

Das CMIP/SNMP-Gateway nutzt die IIMC-Erkenntnisse zur Interoperabilität zwischen den OSI– und Internet–Architekturen und soll Internet–Managementagenten in ein OSI–basiertes Management integrieren. Als OSI–konformes Managementsystem wurde die *IBM NetView TMN Support Facility* eingesetzt; die Implementierung des Gateways selbst geschah mit der *IBM TMN Workbench*, einem Entwicklungssystem für OSI-Agenten, das detailliert in [2] beschrieben ist. Zur Kommunikation zwischen Manager und Gateway wird CMIP eingesetzt; das Protokoll zwischen Gateway und den Agenten ist SNMP in der Version 2. Der herausragende Vorteil der IBM TMN-Entwicklungsumgebung liegt in der Verwendung eines neuartigen CMIP-API, das zur Zeit vom NM Forum genormt wird [12]: es ist damit möglich, OSI–Agenten unter Umgehung von XMP und dessen Komplexität zu implementieren. Der Entwickler wird vollständig vom Managementprotokoll abgeschirmt und kann aus den ihm in der OSI–Notation vorliegenden Beschreibungen der Managementschnittstellen direkt Schnittstellen in der Programmiersprache C++ generieren. Die eigentliche Managementfunktionalität wird in die Rümpfe sog. *Callbacks* eingetragen. Der Einarbeitungsaufwand sinkt durch den Komfort des CMIP-API erheblich.

Allgemein lassen sich Übergänge für Protokollelemente, für die es in beiden Architekturen Entsprechungen gibt, relativ einfach aufeinander abbilden. Schwierig ist jedoch beispielsweise die Umsetzung von CMIP-Action Diensten, da es in SNMP hierzu keine äquivalente Protokolldateneinheit gibt. Dies ist ein klassisches Beispiel für die enge Verzahnung der Informations- und Kommunikationsmodelle (siehe Abschnitte 2.3 und 2.4): Funktionalität, die das OSI-Kommunikationsmodell bietet, muß in der Internet-Welt durch das Internet-Informationsmodell nachgebildet werden. Tatsächlich geschieht das Ausführen von Aktionen auf SNMP-Managementobjekten durch das Belegen von MIB-Variablen mit einer Zahl durch eine SNMP-`set` Protokolldateneinheit.
Ein weiteres Problem stellen die sehr leistungsfähigen *Scoping*– und *Filtering*–Operationen der OSI–Managementarchitektur da, die es gestatten, Managementoperationen auf ganzen Teilbäumen einer MIB durchzuführen. In früheren Gateway-Projekten [1, 8] lag das Hauptproblem oft darin, die `get`-, `set`- oder `action`-Protokolldateneinheiten auf die richtigen Objekte anzuwenden. Diese Arbeit wurde uns vom Entwicklungssystem abgenommen, da zu jedem entwickelten Agenten automatisch ein Dämon erzeugt wird, der „scoped und filtered“ Operationen in einzelne `get`-, `set`- bzw. `action`-PDUs auflöst, die einfacher auf SNMP-PDUs abgebildet werden können.

Insgesamt kann mit dem entwickelten Gateway das gesamte Spektrum der OSI-Protokollfunktionalität (insbesondere *Scoping* und *Filtering*) auf SNMP-Managementobjekte angewendet werden. Es ist damit möglich, SNMP-Ressourcen mit dem vollen OSI-Funktionsumfang zu überwachen und zu steuern, da sie aus der Sicht des Managers OSI-konform erscheinen. Dies bedeutet, daß ein Administrator, der SNMP-Agenten von einem über ein Management-Gateway verbundenen OSI-Manager aus überwacht, komfortablere und leistungsfähigere Operationen zur Verfügung hat, als ein Administrator, der dieselben SNMP-Agenten unmittelbar mit einem SNMP-Manager steuert.

Die vorgestellte Lösung ist vollkommen produktunabhängig, da sie sich auf standardisierte Managementarchitekturen abstützt. Sie funktioniert mit jedem OSI-konformen Managementsystem und allen SNMP-Agenten, sofern deren MIBs dem Gateway mit Hilfe des Specification Translation Algorithmus bekanntgegeben wurden. Allerdings ist die Lösung bisher nicht skalierbar: momentan verwaltet der Gateway-Prototyp die MIB-II und eine am Lehrstuhl entwickelte MIB für UNIX-Systemmanagement, die 195 Variablen und 15 Tabellen umfaßt. Der ausführbare Code des Gateway-Prototypen hat eine Größe von ca. 4 Megabyte.

Bei unseren Arbeiten hat sich der Gateway-basierte Ansatz zur Integration heterogener Managementarchitekturen als die tragfähigste der eingangs vorgestellten drei Integrationsalternativen herausgestellt. Wir vertreten daher die Auffassung, daß der Gateway-basierte Ansatz derjenige ist, mit dem man dem Ziel integrierten Managements in heterogenen Systemen am nächsten kommt.

5 Zusammenfassung und Ausblick

Der Beitrag hat gezeigt, daß es durchaus möglich und aussichtsreich ist, Übergänge zwischen Managementarchitekturen zu schaffen. Diese Übergänge können allerdings, auch das hat sich bei den Projekten gezeigt, wegen der Unterschiedlichkeit der Architekturen zwangsläufig nicht vollständig nahtlos sein. Es wird aber möglich sein, die relativen Meriten der einzelnen Architekturen jeweils weitgehend zu nutzen, ohne „Management-Inseln“ zu bilden. Man ist dann also einen Schritt weiter auf dem Weg zu einem integrierten Management. Gleichzeitig bedeutet dies, einerseits vorhandene Investitionen zu retten (z.B. die große Zahl an SNMP-Agenten), und andererseits neue CORBA-Investitionen auch für das Management (mit) zu nutzen. Die notwendige „Management-Spezialisierung“ von CORBA könnte, wie dargestellt wurde, in einer Übernahme und Übersetzung von MIBs aus der SNMP und CMIP-Welt bestehen. „CORBA-Facilities“ für das Systemmanagement könnten via Specification-Translation aus OSI Systems Management Function mit entsprechender Kapselung evtl. vorhandener Implementierungen erzeugt werden.

Danksagung

Die Autoren danken dem Münchner Netzmanagement Team für intensive Diskussionen zu früheren Versionen dieses Beitrags. Das MNM-Team, das von Prof. Hegering geleitet wird, ist eine Gruppe von Wissenschaftlern beider Münchner Universitäten und des Leibniz-Rechenzentrums der Bayerischen Akademie der Wissenschaften.

Literatur

[1] Abeck, S., Clemm, A., Hollberg, U.: Simply Open Network Management: An Approach for the Integration of SNMP into OSI Management Concepts. In: *Proceedings of the 3rd IFIP/IEEE International Symposium on Integrated Network Management*. North-Holland, April 1993

[2] Feridun, M., Heusler, L., Nielsen, R.: Implementing OSI Agents for TMN. IBM Research Report RZ 2759. IBM Research Division, Zurich Research Laboratory. (1995)

[3] Hegering, H.-G., Abeck, S.: Integrated Network and System Management. Addison-Wesley 1994

[4] Hegering, H.-G., Neumair, B., Gutschmidt, M.: Architekturen und Konzepte für ein integriertes Management von verteilten Systemen. Informatik Spektrum *18*(5), (Oktober 1995)

[5] Hegering, H.-G., Neumair, B., Gutschmidt, M.: Cooperative Computing and Integrated System Management — A Critical Comparison of Architectural Approaches. Journal of Network and Systems Management *2*(3), 283–316 (1994)

[6] Heilbronner, S., Keller, A., Neumair, B.: Integriertes Netz- und Systemmanagement mit modularen Agenten. In: *Proceedings of SIWORK'96, Zurich, Switzerland*. Mai 1996

[7] Höller, T.: Konzeption und Realisierung eines CORBA / SNMP Gateways. Diplomarbeit. Technische Universität München. August 1996

[8] Kalyanasundaram, P., Sethi, A.: An Application Gateway Design for OSI-Internet Management. In: *Proceedings of the 3rd IFIP/IEEE International Symposium on Integrated Network Management*. North-Holland, April 1993

[9] Kalyanasundaram, P., Sethi, A.: Interoperability Issues in Heterogeneous Network Management. Journal of Network and Systems Management *2*(2), (1994)

[10] Keller, A., Neumair, B.: Systems Management Middleware: Verteilte objektorientierte Technologien für das Systemmanagement. In Wall, D. (Hrsg.): *Organisation und Betrieb von DV-Versorgungsstrukturen*. Deutscher Universitäts-Verlag, November 1995

[11] Mazumdar, S., Brady, S., Levine, D.: Design of Protocol Independent Management Agent to Support SNMP and CMIP Queries. In: *Proceedings of the 3rd IFIP/IEEE International Symposium on Integrated Network Management*. North-Holland, April 1993

[12] TMN C++ Application Programming Interface). Issue 1.0, Draft 5 - For Public Comment NMF xxx. Network Management Forum. (Januar 1996)

[13] CORBAservices: Common Object Services Specification, Volume 1. OMG Specification. Object Management Group. (März 1996)

[14] Sloman, M. S. (Hrsg.): Network and Distributed Systems Management. Addison Wesley 1994

[15] SOMobjects Developer Toolkit Programmer's Guide Volume 2: Object Services. IBM Corporation. März 1996. First Edition

[16] Vogs, T.: Entwurf und Implementierung eines Konzepts zur Anbindung von Object Request Brokern an eine Netzmanagementplattform. Diplomarbeit. Technische Universität München. Februar 1996

[17] Inter-Domain Management Specifications: Specification Translation (Final Sanity Check Draft). Preliminary Specification Pxxx. X/Open Ltd. (September 1996)