Das *MELODY*-Managementsystem für verteilte Anwendungen[1]

Ernö Kovàcs
Universität Stuttgart, Institut für Parallele und Verteilte Höchstleistungsrechner (IPVR),
Breitwiesenstr. 20-22, 70565 Stuttgart
e-mail: ernoe.kovacs@informatik.uni-stuttgart.de

Kurzfassung:

Verteilte Anwendungen besitzen eine inhärente Komplexität, die sich in der Praxis durch aufwendige und personalintensive Installation, Wartung und Problemsuche bemerkbar macht. Dieser Komplexität muß mit geeigneten Managementwerkzeugen reduziert werden. Das *MELODY*-Managementsystem unterstützt die Integration von Managementfunktionen in verteilte Anwendungen. Es realisiert ein objekt-orientiertes Informationsmodell, welches direkt in die Komponenten einer verteilten Anwendung integriert wird. Die realisierten Objektdienste verbergen die notwendigen Kommunikationsvorgänge vor der Anwendung. Die unterstützende Managementplattform ermöglicht einen ortstransparenten, effizienten Zugriff unter Berücksichtigung vorgegebener Aktualitätsbedingungen. Managementanwendungen werden durch die Bereitstellung geeigneter Funktionen zur Überwachung und Steuerung des Systems unterstützt.

Abstract:

Distributed applications contain an inherent complexity which in praxis become apparent through high costs for installation, maintenance and problem determination. This complexity must be reduced by appropriate management tools. The *MELODY* management system supports the integration of management functions into distributed applications. It realizes an object-oriented information model which is directly integrated into the component of a distributed application. The realized object services hides the necessary communication procedures from the application. The supporting management platform enables a location transparent, efficient access with respect to preset actuality requirements. Management applications are supported through functions to monitor and control the system.

1. Einleitung

Das Forschungsgebiet Management verteilter Systeme hat in den letzten Jahren stark an Bedeutung gewonnen. Die schnell zunehmende Verbreitung von Arbeitsplatzrechnern und vor allen Dingen die Verbindung dieser Rechner mit Hilfe schneller Rechnernetze führte zu dem Einsatz immer mehr verteilter Systeme. Insbesondere neue, innovative Anwendungen wie Multimedia-Kommunikation, kooperatives Arbeiten, Informationsaustausch in offenen Netzen und die Abbildung ganzer Arbeitsabläufe auf Abläufe in einem verteilten System bieten neue, interessante Anwendungsmöglichkeiten. Verteilte Systeme besitzen jedoch eine inhärente Komplexität. Zwischen den verschiedenen Teilen des Systems besteht ein hohes Maß an Abhängigkeiten. Verteilte Systeme bestehen aus sehr vielen Komponenten, die gezielt und fehlerfrei zusammenwirken müssen, damit eine Gesamtleistung erbracht werden kann. Die Aktivitäten im System beschränken sich nicht auf einen Knoten, sondern involvieren mehrere Rechner gleichzeitig, so daß es schwer ist, einen Überblick über alle beteiligten Komponenten zu erhalten. Zu der Komplexität trägt auch die große Zahl an unterschiedlichen Systemen bei, die heutzutage zusammenwirken. Dabei ist es nicht nur die Heterogenität der Hard- und Software problematisch, sondern auch die Vielfalt an einzelnen Systemen, die gleiche oder ähnliche Funktionalität haben. Höhere Flexibilität, geringere Kosten, bessere Ressourcennutzung und ein verbesserter Nutzerkomfort stehen heutzutage einem erhöhten Aufwand bei der Verwaltung des Gesamtsystems gegenüber. Diese Situation erfordert die Realisierung geeigneter Managementwerkzeuge. Es fehlt jedoch - trotz vieler

---

[1] Die Ergebnisse dieser Arbeit entstanden teilweise im Rahmen des Studienvertrag S142 “Management verteilter Systeme” mit der IBM Deutschland Informationssysteme GmbH.

Fortschritte - an einer allgemein anerkannten Methodik und an einheitlichen Konzepten zur Erstellung solcher Managementwerkzeugen.

Managementdienste wurden jeweils für spezielle Systeme entwickelt. Sie deckten damit immer nur bestimmte Teilbereiche ab und unterschieden sich in der Vorgehensweise und in den eingesetzten Konzepten. Erst der Trend zu offenen, heterogenen Systemen brachte den Wunsch zu einer einheitlichen, standardisierten Vorgehensweise. Ausgangspunkt für diese Forschungsaktivitäten war das Management der Rechnernetze - das Netzwerkmanagement. Daneben ergaben sich auch Anforderungen an ein einheitliches Systemmanagement, d.h. an ein Management der angeschlossenen Endsysteme. Die rasante Entwicklung und der Einsatz verteilter Anwendungen erfordert auch die Untersuchung von Managementfunktionen für diesen Einsatzbereich. Heutzutage ist das Ziel eine integrierte Managementlösung für alle Zielbereiche verteilter Systeme (vergleiche dazu auch [Hege91], [HAV93]). Trotz großer Fortschritte in der Entwicklung einer gemeinsamen Methodik bringt jeder Zielbereich jeweils eigene Probleme mit sich, die neue Anforderungen an die Managementfunktionen und an die unterstützenden Werkzeuge stellen. Vielfach sieht es so aus, als würden neue, verbesserte Konzepte hinzukommen müssen, um die gestellten Aufgaben zu lösen. Auf dem Gebiet der verteilten Systeme ist immer noch eine hohe Innovationsgeschwindigkeit festzustellen, so daß sich auch weiterhin ständige Anpassungen und Neubewertungen der geforderten Managementaufgaben ergeben werden.

Das Management verteilter Anwendungssysteme beschäftigt sich mit den Anwendungsprogrammen, die verteilt auf den unterschiedlichen Knoten des Systems laufen. Auch hier ist es sinnvoll - und für den praktischen Einsatz unumgänglich - verschiedene Funktionsbereiche des Managements (z.B. Konfigurations-, Fehler-, Leistungs-, Abrechnungs- oder Sicherheitsmanagement) zu unterscheiden. Diese einzelnen Bereiche liefern qualitative und quantitative Aussagen über die Abläufe in dem System unter jeweils einem bestimmten Gesichtspunkt. So muß das Konfigurationsmanagement z.B. Informationen über in dem Netz vorhandene zentrale Server liefern. Dazu zählen - neben vielen anderen - statistische Aussagen wie momentane oder durchschnittliche Verfügbarkeit, Informationen über den Knoten, auf dem ein Server gerade läuft, oder mit wem er gerade in einer langdauernden Verbindung steht. Für das Fehlermanagement sind neben Aussagen wie Fehlerraten beim Aufbau von Verbindungen oder nicht erfolgreich durchgeführte RPC-Aufrufe, besonders aktuelle Fehlermeldungen über nicht verfügbare oder gerade abgestürzte zentrale Komponenten wünschenswert. Das Leistungsmanagement kann anhand der Anzahl durchgeführter RPC-Aufrufe Engpässe im System ausfindig machen. Dabei können die Zähler durchaus den einzelnen Operationen, die an einem Interface angeboten werden, zugeordnet werden. Ganz im Sinne eines integrierten Managements lassen sich viele Vorgehensweisen aus dem Netz- und Systemmanagement auf das Management verteilter Anwendungen übertragen. Es ergeben sich aber noch weitere interessante Anforderungen, auf die wir in diesem Beitrag eingehen werden.

Der vorliegende Beitrag weist folgende Struktur auf: Zuerst wird ein einfaches Modell für verteilte Anwendungen eingeführt, welches eine gemeinsame Basis für das Verständnis verteilter Systeme bietet. Danach wird auf die Probleme eingegangen, die sich aus dem Bereich Anwendungsmanagement ergeben. Dann wird das *Melody*-Managementsystems beschrieben, welches insbesondere das Management verteilter Anwendungen in großen Netzen betrachtet. Der Schwerpunkt wird hierbei insbesondere auf die neuen Anforderungen gelegt, die sich aus dem Anwendungsmanagement ergeben.

2. Modell einer verteilten Anwendung

Dieser Abschnitt führt ein Modell für verteilte Anwendungen ein. Dieses Modell betont die Gemeinsamkeit verschiedener Architekturen für verteilte Systeme. Damit werden natürlich systembedingte Eigenschaften einzelner Systeme zugunsten eines verallgemeinerten Modells vernachlässigt. Naturgemäß kann ein solcherart vereinfachtes Modell nicht alle unterschiedlichen Aspekte eines verteilten Systems erfassen. Betrachtet man eine verteilte Anwendung jedoch auf diesem Abstraktionsniveau, so lassen sich leichter technologisch oder systembedingte Unterschiede vernachlässigen und Gemeinsamkeiten sich besser hervorheben.

Eine verteilte Anwendung besteht aus Teilkomponenten, die auf den Knoten eines verteilten Systems ablaufen. Diese Teilkomponenten kommunizieren um gemeinsam eine Leistung zu erbringen. Die Kommunikation erfolgt mit Hilfe eines Kommunikationsstacks (z.B. TCP/IP oder OSI), der verschiedene Kommunikationsdienste anbietet und ein Kommunikationsprotokoll verwendet. Ein Komponente wird durch die verwendeten *Schnittstellen*, den *Komponententyp* und die *Rolle* der Komponente beschrieben.

Eine Schnittstelle beschreibt die Kommunikationsform, die eine Komponente in dem jeweiligen System verwendet. In einem Client-Server System enthält die Schnittstelle eines Servers die Operationen, die ein Client aufrufen kann. Die Schnittstelle des Clienten besteht dann aus den Diensten, mit denen eine entfernte Operation aufgerufen wird. Eine Teilkomponente kann mehrere Schnittstellen besitzen, die jeweils Client oder Server-Funktionalitäten besitzen können. In unserem Modell haben Schnittstellen einen Typ. Dieser Typ definiert die Art der Interaktionen (asynchron oder synchron), das verwendete Kommunikationsmuster (einfacher Nachrichtenaustausch, Request-Reply, Datenstrom), die Rolle des Kommunikationspartners (z.B. Client - Server, Erzeuger - Verbraucher), den verwendeten Kommunikationsstack (z.B. TCP/IP oder OSI) und weitere charakteristische Eigenschaften. Die Schnittstelle erfaßt damit eine Menge von zusammengehörenden, kommunikationsrelevanten Eigenschaften, die eine Komponente besitzt. Beispiele für eine ähnliche Definition von Schnittstellen eine Komponente lassen sich bereits in [BiNe84], in dem ANSA-Modell ([ANSA91]), in der Definition des DCE-Kommunikationsmodells ([OSF91]) oder in [ZFD93] finden.

Der Komponententyp legt die Verarbeitungsfunktionalität der Komponente fest. Dies ist insbesondere wichtig für Komponenten, bei denen die Schnittstelle nicht eindeutig die Funktionalität festlegt. In einer Verarbeitungspipline, in der jede Komponente das gleiche Interface besitzt, bestimmt der Komponententyp, welche Aktionen eine Komponente auf den Elementen des Datenstroms durchführt. Naturgemäß gibt es einen engen Zusammenhang zwischen dem Typ der Schnittstelle und dem Typ der Komponente, jedoch kann ein bestimmter Komponententyp unterschiedliche Schnittstellen haben, genauso wie eine Schnittstelle zu unterschiedlichen Komponententypen passen kann.

Komponenten mit der gleichen Schnittstelle und dem gleichen Komponententyp können zur Laufzeit unterschiedliche Rollen einnehmen. Eine Rolle beschreibt die individuelle Ausprägung, die eine Komponente in dem System einnimmt. In einem Pool aus gleichwertigen Servern unterscheidet man meist keine verschiedenen Rollen. In vielen Fällen gibt es aber ausgezeichnete Rollen, die eine spezielle Eigenschaft des Servers definieren. Ein Beispiel hierfür ist der Wurzelserver einer verteilten, hierarchischen Namensverwaltung (Beispiel DNS), der die gleiche Funktionalität wie alle anderen hat, auf Grund seiner besonderen Bedeutung aber eine eigene Rolle besitzt. Die Rolle beschreibt aber auch die Zugehörigkeit zu verschiedenen Organisationen (der Dokumentenserver der Abteilung X) oder spezielle Funktionen (z.B.: Ersatz-Server) bezeichnen. Eine Komponente kann durchaus mehrere Rollen annehmen. Die Rollen können sich in Lauf der Zeit ändern.

Komponenten interagieren miteinander. Eine Interaktion involviert jeweils zwei oder mehr Komponenten. Welche Komponenten miteinander kommunizieren, kann entweder durch eine statisch festgelegt Bindung, jeweils am Anfang einer Reihe von Nachrichtenübertragungen (eine Session) oder von Interaktion zu Interaktion neu bestimmt werden.

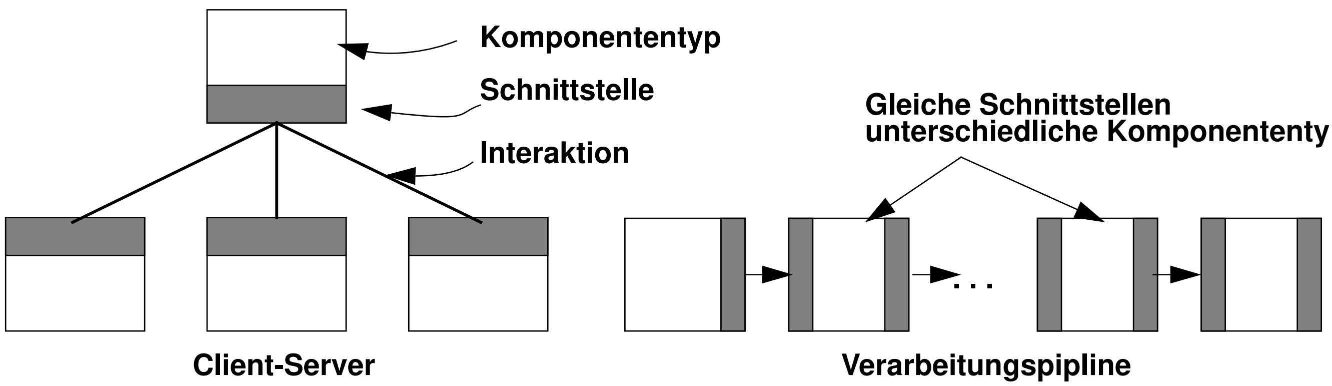

Bild 1 : Beispiele für verteilte Anwendungen

3. Management einer verteilten Anwendungen

Das Management einer verteilten Anwendung erfordert die Überwachung und die Steuerung der verschiedenen Komponenten des verteilten Systems. Wie in anderen Managementbereichen ist es auch hier zweckmäßig, die verschiedenen Aufgabenbereiche (z.B. Konfigurations-, Fehler-, Leistungs-, Sicherheits- und Abrechnungsmanagement) zu unterscheiden. Bekannte Lösungen, wie sie z.B. in [HeAb93] für das integrierte Netz- und Systemmanagement beschrieben werden, lassen sich dann auf Management verteilter Anwendungen übertragen.

Im einzelnen lassen sich in den einzelnen Managementgebieten u.a. folgende Aufgaben für verteilte Anwendungen bestimmen:

- Konfigurationsmanagement:

  Zustandsüberwachung der Komponenten, Identifikation von Eigenschaften einer Komponente (Typ, Rolle, Lokation), Beschreibung des verwendeten Kommunikationssystems (Typ, konkrete Eigenschaften, notwendige Zustandsinformation), Informationen über die Interaktionen (statische und dynamische Bindungen, aktueller Zustand), Installation, Starten und Stoppen von Komponenten, Anpassen von Operationsparameter, Änderungen der Gesamtkonfiguration

- Fehlermanagement

  Erkennen von Störungen, Durchführen von Tests, automatisches Reagieren auf Störungen (beispielsweise durch erneutes Starten einer Komponente), Unterstützung eines Operators bei manuellen Fehlererkennen und Beheben, langfristige Beobachtungen des Systems (Anzahl der Verbindungsaufbaufehler, unterbrochene Verbindungen), Konsistenzprüfungen, Sammlung von Fehlernachrichten

- Leistungsmanagement

  Überwachen von Leistungsparameter einer Schnittstelle (Anzahl eingetroffener oder ausgehender Anfragen, Anzahl der noch nicht verarbeiteten Anfragen, Volumen der übermittelten Daten, Wartezeiten), Überwachung der Qualitätsmerkmale einer Komponente (Gesamtverzögerung eines Auftrags, durchschnittliche Bearbeitungsdauer einzelner Vorgänge, Gesamtdurchsatz), Ermittlung von Leistungsengpässen, Maßnahmen zur Vermeidung von Leistungsengpässen

- Abrechnungsmanagement

  Ermittlung und Aufzeichnen der Benutzer spezifischen Kosten, Definition der abzurechnenden Arbeitsschritte und der assoziierten Kosten, Definition der zusätzlichen Information, die zur Belegen der Abrechnung nötig sind

- Sicherheitsmanagement

  Beobachtung und Kontrolle der Sicherheitsmechanismen, Überwachung der Authentifizierung und der Zugriffskontrolle, Aufspüren von Einbruchsversuchen

Neben diesen aus der OSI-Standardisierung bekannten Aufgabenbereichen läßt sich auch ein Trend zu umfassenderen Planungs- und Organisationsunterstützung feststellen. So gibt es Ansätze zur langfristigen Konfigurations- und Kapazitätsplanung, zur Unterstützung von Inventarisierung, zur finanziellen Planung und zur Ermittlung der Gesamtkostensituation, sowie zur Verwaltung und Bearbeitung von Problemberichten (Trouble-Tickets, siehe z.B. [VaJa93]). Diese Ansätze sollen hier aber nicht weiter verfolgt werden.Im folgenden wollen wir uns auf die Managementunterstützung in der Betriebsphase eines verteilten Systems konzentrieren.

4. Das *MELODY*-Managementsystem: Grundannahmen

Das *MELODY*-Managementsystem (*MELODY* - Management Environment for Large Open Distributed sYstems) zielt auf das Management verteilter Anwendungssysteme vor allen Dingen in der Betriebsphase. Zu den Grundannahmen bei der Entwicklung des Systems zählt die Unabhängigkeit von einer Entwicklungsplattform für verteilte Systeme wie es z.B. das Distributed Computing Environment (DCE, siehe [OSF91], [Schi92]) darstellt. Eine solche Entwicklungsumgebung bietet natürlich reiche Unterstützung zur Realisierung der gestellten Aufgaben. Insbesondere bereits vorhandene Funktionen zum Management können für verschiedene Testzwecke genutzt werden. Dies birgt aber die Gefahr in sich, sich vor allen Dingen auf die vorgegebenen Möglichkeiten zu konzentrieren. Eine solche Entwicklungsumgebungen stellt für uns daher einen Prüfstein dar, an dem sich die Eignung der entwickelten Konzepte und Werkzeuge zeigen soll.

Eine weitere Grundannahme bei der Konzeption des Managementsystems fordert, daß möglichst viele Erkenntnisse bisheriger Managementsysteme - z.B. ein einheitliches, objektorientiertes Informationsmodell, ein standardisiertes Managementprotokoll oder die bisher definierten Services zum Zugriff auf die verteilte Information - berücksichtigt werden, jedoch in Hinblick auf das neue Zielgebiet nochmals auf ihre Eignung überprüft werden. Wo es nötig ist, sollen neue Konzepte entwickelt und neue Abstraktionen dem Systemprogrammierer an die Hand gegeben werden.

Bei der Konzeption des Gesamtsystems wurden bekannte Grundkonzepte existierender Managementsysteme berücksichtigt. Das System verwendet das Manager-Agenten Modell. Dabei realisieren die Agenten den Zugriff auf die Managementinformation eines Knoten des Systems. Sie stellen diese Information in einer abstrakten Darstellung in der Form einer Managementinformationsbasis (MIB) zur Verfügung. Die MIB enthält dabei in erster Linie die Grunddaten die für die einzelnen Managementaufgaben benötigt werden. Die Managementanwendungen sammeln diese Informationen und realisieren damit die notwendigen Managementwerkzeuge. Agenten realisieren jeweils auch die Kontrolloperationen zur Manipulation des verteilten Systems. Diese Vorgehensweise ermöglicht eine strikte Trennung zwischen der Anwendung und dem Managementsystem. Im Prinzip kann bei verteilten Anwendungen ein Manager direkt mit den entsprechenden Komponenten interagieren. Dadurch verliert man aber die Unabhängigkeit des Managementsystems von der zu kontrollierenden Anwendung, muß sich an die verwendeten Kommunikationsstacks und Informationsmodelle anpassen und kann insbesondere bei Fehlersituationen nicht unabhängig von der Anwendung handeln.

Bei der Realisierung einer Umgebung zum Management eines solchen Systems gibt es drei wesentliche Arbeitsbereiche:

- Integration der Managementfunktionalität in die verteilten Teilkomponenten der Anwendung, so daß die lokalen Agenten auf die Komponenten einwirken und diese kontrollieren können.
- Unterstützung bei der Erstellung von Managementanwendungen
- Realisierung einer Managementplattform, die unterstützende Dienste und generische Kommunikationsfunktionen für beide Seiten (Anwendung und Manager) bereitstellt

Im folgenden werden die Anforderungen vorgestellt, die dabei von dem Managementsystem zu lösen sind.

4.1 Integration der Managementfunktionalität in verteilte Anwendungen

Diese Aufgabe wird als die Instrumentierung einer Anwendung bezeichnet. Geeignete Instrumente (z.B: Sensoren und Regler wie in [MCWB91]) müssen in die einzelne Komponenten eingebracht werden. Damit können Management-relevant Informationen, die in einer Komponente vorhanden sind, dem Agenten zur Verfügung gestellt werden. Eine Komponente beschreibt u.a. die vorhandenen Schnittstellen, den eigenen Typ und die eigene Rolle. Weiterhin muß die Anwendung die Änderungen von Werten durch das Managementsystem in einer kontrollierten Weise zulassen und gegebenenfalls auch komplexe Operationen auf Veranlassung des Managementsystems durchführen. Zur Unterstützung eines ereignisgesteuerten Managements kann eine Teilkomponente von sich aus bestimmte Ereignisse oder Zustände bekanntgeben und dafür Meldungen an das Managementsystem weiterleiten. Neben den zu lösenden operationalen Problemen (Wie kann ich Daten aus dem Prozeß erhalten, wie kann ich eine Managementoperation von außen initiieren?), gilt es darüberhinaus auch Kontrollaspekte (Wann darf ich eine Operation durchführen?) zu berücksichtigen, denn je nach Zustand der Anwendung kann es zulässig sein, Managementoperationen durchzuführen oder nicht.

In bisher existierenden Managementplattformen kann das Managementsystem meist nur indirekt über systemabhängige, speziell auf die jeweilige Anwendung zugeschnittene Hilfsprogramme Managementoperationen durchführen. Es gibt keine direkte, eindeutig definierte Schnittstelle zwischen Anwendungskomponenten und dem Managementsystem. Für diesen Bereich gilt es, eine einfache Schnittstelle zwischen der Teilkomponente und einem Managementagenten auf dem jeweiligen Knoten zu entwickeln. Über diese Schnittstelle müssen sich Komponenten dynamisch bei dem Managementsystem an- und abmelden können. Sie muß einerseits die mit den Managementaufgaben verbundenen Kommunikationsaufgaben verbergen, andererseits aber der Komponente auch genügende Kontrollmöglichkeiten über die Managementoperationen geben, um unzulässige Störungen durch das Management, z.B. in einem kritischen Abschnitt, zu verhindern.

Diese interne Schnittstelle zwischen den Informationen in einer Komponente und dem Managementsystem ist in verschiedenen Forschungsprojekten identifiziert worden (z.B. in [KPW91]). Es zeichnet sich ab, daß hier eine standardisierte Schnittstelle entwickelt wird. Dadurch können auch unterschiedliche Managementsystem, mit unterschiedlichen Agenten und Kommunikationsprotokollen, auf die gleichen Informationen zugreifen. Ein entsprechender Ansatz wird in [SyTa93] für das Netzwerk- und Systemmanagement beschrieben.

4.2 Unterstützung bei der Erstellung der Managementfunktionen

Bisherige Schnittstellen zur Realisierung einer Managementanwendung orientierten sich fast ausschließlich an den zur Verfügung stehenden Operationen des Managementprotokolls. Im Bereich des Anwendungsmanagements kann das Management aber durch weitere Funktionalitäten unterstützt werden.

Im Gegensatz zum Netzwerk- oder Systemmanagement, bei dem das Zielsystem eindeutig über eine Rechnernetzadresse identifiziert werden kann, sollte beim Management verteilter Anwendungssystemen ein ortstransparenter Zugriff auf die Teilkomponenten durchgeführt werden können. Dazu wird ein geeignetes, global eindeutiges Namensschema benötigt. Diese Namen müssen dann auf den Ort der Komponente abgebildet werden. Gleichzeitig hat sich aber auch das Absuchen (Browsen) der Knoten als ein wirkungsvoller Mechanismus für das Management verteilter Systeme erwiesen. Daher sollen Namensschemata auf unterschiedlichen Abstraktionsniveaus möglich sein. Weiterhin müssen noch Namen, die durch Entwicklungsplattformen für verteilte Systeme vorgegeben werden, berücksichtigt werden. Diese sollen ebenfalls zur Benennung von Objekten des Managementsystems vorgesehen werden. Dazu ist meist ein Abbildungsschritt in der Form einer Directory-Suche, gefolgt von einer Anfrage an das Zielsystem nach dem gesuchten Managementobjekt, erforderlich. Die Managementanwendung sollte die nötigen Arbeitsschritte durch die entsprechende Managementplattform realisiert lassen.

Eine Managementanwendung spezifiziert, an welchen Objekten sie langfristig Interesse hat. Sie sollte sich im Anschluß daran auch nicht um geeignete Optimierungen für einen effizienten Zugriff (z.B. durch die Vorgabe eines periodischen Zugriffs mit einem festen Intervall) kümmern müssen. Vielmehr sollte sie die Anforderungen an die Aktualität der betrachteten Daten formulieren und der Managementplattform dieses Wissen zur Verfügung stellen. Diese sollte dann eine geeignete Zugriffsstrategie (direkter Zugriff auf die Information, periodisches Updaten einer lokalen Kopie oder das Versenden von Änderungsnachrichten) auswählen. Die Plattform kann dabei unter Berücksichtigung der aktuellen Situation eine optimale Strategie wählen.

Für die in einem verteilten System vorkommenden asynchronen Vorgänge, z.B. bei dem parallelen Zugriff auf Managementinformation innerhalb verschiedener Knoten des Systems, sollten den Anwendungsprogrammierern geeignete Abstraktionen zur Verfügung stehen, um die eintreffenden Ereignisse passend behandeln zu können.

Daneben gibt es noch eine Reihe weitere praktische Anforderungen, z.B. an geeigneter Unterstützung für die Präsentation der Managementinformation, für die permanente Speicherung z.B. in Datenbanken, für die Generierung und Auswertung von Meßreihen oder die Erstellung von Reports.

4.3 Anforderungen an die Managementplattform

Die Managementplattform muß beiden Teilen, der Anwendung und dem Manager, geeignete Kommunikationsunterstützung anbieten. So muß zur Realisierung der globalen, ortstransparenten Namen ein verteiltes Auskunftssystem verwendet werden, um die Informationen über die laufenden Komponenten des verteilten Systems zu speichern. Auch hier sollen die notwendigen Kommunikationsvorgänge verborgen werden. Aus der Sicht einer Managementanwendung muß die Managementplattform die Funktionen eines Management Request Brokers (vergl. [OMG91], [Geih92]) bieten, d.h. zu einem gegebenen Auftrag mit einer darin enthaltenen Objektreferenz muß der Ort des Objektes und die Methode, die zur Ausführung des Auftrages aufgerufen werden muß, gefunden werden, der Request an die entsprechende Stelle weitergeleitet und dort ausgeführt werden. Je nach Objekttyp und beteiligtem Zielsystem kann es nötig sein, anstelle der oben geforderten direkten Schnittstelle zur Anwendung auch auf ein lokales Objektrepository zuzugreifen oder beispielsweise eine Hilfsprogramm aufzurufen, welches die entsprechende Information zur Verfügung stellt. Die Ergebnisse müssen im Anschluß daran wieder zurück an den Absender gesendet werden.

Desweiteren sollte die Managementplattform Triggerfunktionen enthalten, die auf bestimmte, allgemeine Ereignisse reagieren, die nicht in den Anwendungen realisiert sind. Dazu zählt beispielsweise das An- und Abmelden einer neuen Komponente, das Eintreten einer bestimmten Situation oder die Verletzung einer bestimmten Konsistenzbedingung. Diese Ereignisse sollen dann per Ereignismeldungen an interessierte Managers abgesendet werden. Die Managementplattform übernimmt hier die Event-Generierung und das Event-Forwarding.

Wie oben gefordert, soll die Managementplattform für Objekte, die den Manager interessieren, einen Cache bereithalten. Für unterschiedliche Manager auf dem gleichen Knoten soll ein einheitlicher Cache existieren. Da die in verteilten Anwendungssystemen enthaltene Managementinformation sehr unterschiedliche Eigenschaften besitzen kann, muß die Managementplattform den Zugriff auf die einzelnen Daten automatisch gemäß den Eigenschaften und den Anforderungen optimieren. Daneben stellt die Managementplattform Funktionen zum Absuchen von Netzknoten, zum selektiven Zugriff auf Information (z.B. mit Hilfe von Bereichsangaben und Filter) zur Verfügung.

5. Das *MELODY*-Managementsystem

In dem *MELODY*-Projekt wird untersucht, wie die oben angesprochenen Anforderungen realisiert werden können. Im folgenden wird die Systemarchitektur und das eingesetzte Informationsmodell beschrieben. Dabei wird auf die Realisierung der internen Komponenten-Agent Schnittstelle, auf die Realisierung der Ortstransparenz und den effizienten Zugriff auf Managementinformation eingegangen.

5.1 Die Architektur des *MELODY*-Managementsystems

Das *MELODY*-Managementsystem unterscheidet drei logischen Komponenten: Manager, Anwendungskomponente und Managementagenten.

- Anwendungskomponenten sind Gegenstand des Managements. Sie beinhalten die Informationen, auf denen die Managementanwendungen operieren. Aus der Sicht einer Anwendung werden nur die Objekte manipuliert, die die Managementinformation darstellen. Die Kommunikation zwischen Komponente und Agent ist in den Methoden der Objekte verborgen.

- Manager realisieren die Managementaufgaben. Sie operieren ebenfalls auf logisch lokalen Objekten. Diese Objekte stellen jeweils einen Schnappschuß des realen Objektes auf dem entfernten System da. In den Methoden der Statthalterobjekten sind die Kommunikationsvorgänge mit dem Agenten verborgen. Die einzige Ausnahme bildet der Fall, in dem die Managementanwendung explizit parallel arbeiten möchte. In diesem Fall werden die Objektaufrufe asynchron abgearbeitet, d.h. die Anwendung erhält die Antworten des Systems nicht direkt mit dem Ende des Methodenaufrufs, sondern als extern eintreffendes Ereignis. Zur Behandlung von asynchronen Meldungen stehen zwei Modell zur Verfügung. Zum einen das blockierende Warten auf eine Meldung, zum zweiten das Aufbauen einer Warteschlange mit eintreffenden Meldungen, die dann später abgearbeitet werden können.
- Agenten realisieren die Kommunikation zwischen Managern und Komponenten. Das entsprechende Kommunikationprotokoll enthält eine Reihe von Botschaften, mit denen Objekte und ihre Typinformation, Werte von Attributen, Aktionsaufrufe, Ereignisse (mitsamt Ereignisparameter) und Managementsystem-interne Botschaften übertragen werden können. Die Agenten realisieren einen ortstransparenten Zugriff. Ein Manager präsentiert einen Auftrag mit dazugehöriger Objektreferenz. Der Agent lokalisiert daraufhin das gewünschte Objekt und übertragt den Auftrag an das entfernte System. Der Agent realisiert auch den Cache für die Manager und überwacht die geforderte Aktualität.

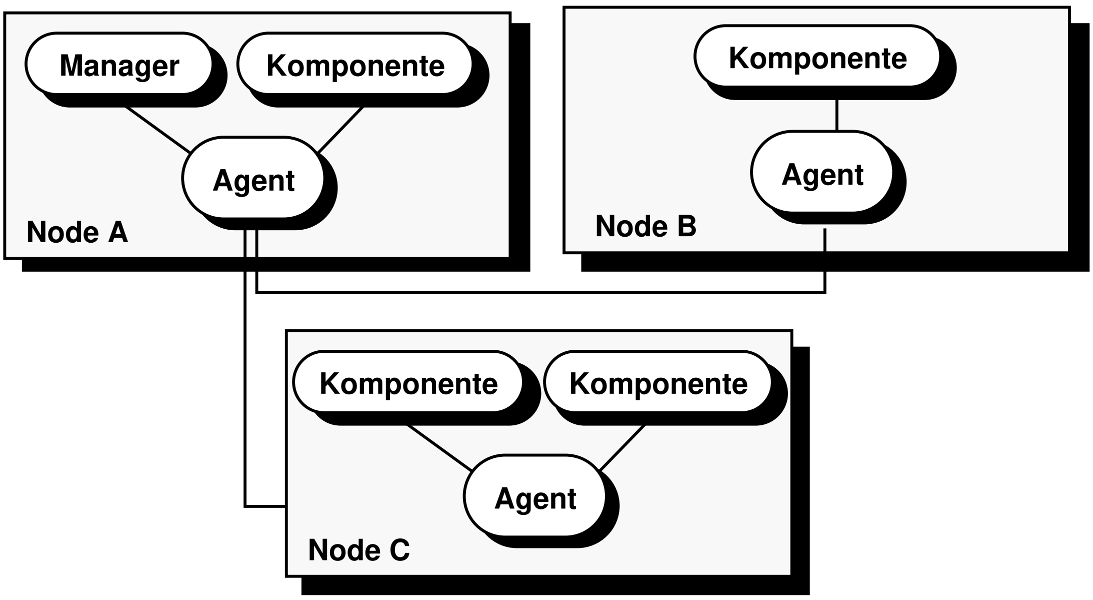

Bild 2 : Architektur des *Melody*-Managementsystems

5.2 Das Informationsmodell

In Anlehnung an das OSI-Informationsmodell wird die Managementinformation einer Teilkomponente mit Hilfe eines Baums von Objekten modelliert. Jedes Objekt korrespondiert zu einem bestimmten Teil der Komponente. Das Wurzelobjekt repräsentiert die gesamte Anwendung und beschreibt auch den Komponententyp. Die untergeordneten Objekte repräsentieren jeweils einen Teilaspekt, beispielsweise die Rolle der Komponente oder die entsprechenden Schnittstellen. Die Objekte besitzen Attribute, die entsprechende Eigenschaften der Objekte darstellen. Attribute können gelesen und geschrieben werden. Ein Objekt besitzt Aktionen, die von außen angestoßen werden können. Desweiteren kann ein Objekt von sich aus Ereignisnachrichten auslösen, die von dem Managementsystem an die entsprechenden Manager weitergeleitet werden.

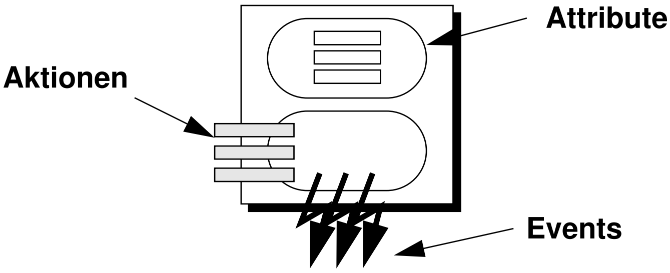

Bild 3 : Objektmodell der Managementinformation

Objekte gehören zu einer Klasse. Klassen werden mit Hilfe eines Vererbungsbaums definiert. Dabei ist Mehrfachvererbung zugelassen. Die Attributmenge, die zu einer Klasse gehört, ist statisch festgelegt. Ein Objekt enthält immer seine Strukturbeschreibung. Diese kann auch abgefragt werden und steht dann einer generischen Managementfunktion zur Verfügung. Da die Objekte stark typisiert sind, könnte diese Aufgabe auch von einem verteilten Typmanager realisiert werden, wie er z.B. im Moment in der ODP-Standardisierung diskutiert wird ([ISO92b], [ISO92a]). Die Vorgehensweise des *MELODY*-Systems macht die Typinformation der Objekte immer mit dem Objekt zusammen verfügbar, so daß eine Reihe generischer Managementfunktionen - z.B. ein generischer Browser - realisiert werden kann.

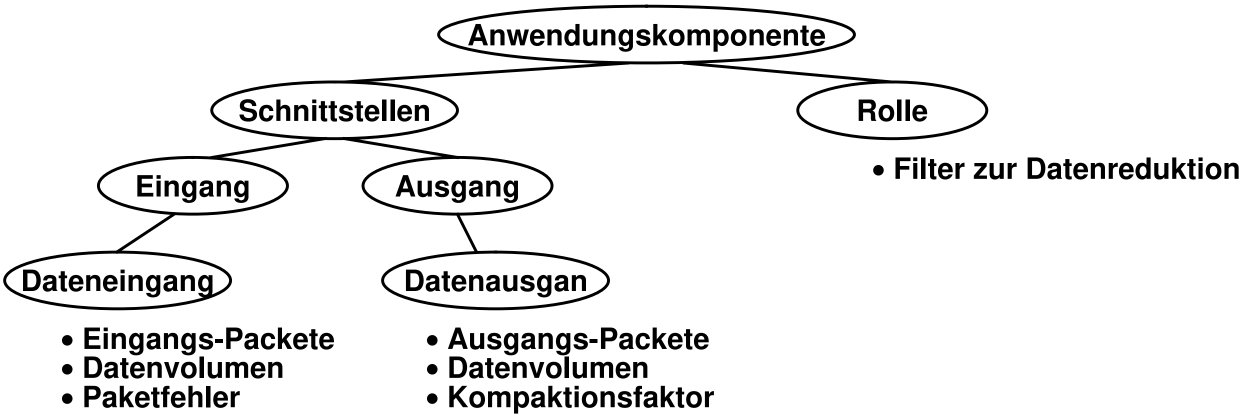

Bild 4 : Anwendungskomponente mit zwei Schnittstellen und einigen Attributen

5.3 Instrumentierung der Anwendung

Die Objekte des Informationsmodells werden 1:1 auf Objekte der Programmiersprache (in diesem Fall C++) abgebildet. Nach der Definition der Objektklasse - bei der im wesentlichen die Oberklassen, die Attribute und die Abbildung der Aktionen auf vorhandene Prozeduren definiert wird, müssen nur noch Instanzen in der entsprechenden Programmiersprache erzeugt werden. Bei der Instanziierung werden die Objekte für die Managementaufgaben vorbereitet. Sie müssen danach noch dem Managementsystem mit einer geeigneten Operation (*Announce*) bekanntgemacht werden und können an Ende wieder zurückgezogen werden (*Revoke*). Eine Anwendung sieht nur lokale Objekte, die sie mit Methodenaufrufen manipuliert. Die Kommunikation mit dem Agenten ist in den Methodenaufrufen verborgen.

Managementsystem und Anwendung sind durch Mechanismen zur Interprozesskommunikation miteinander verbunden. Dabei sind verschiedene Mechanismen realisiert worden: Shared Memory, lokales TCP und UNIX-Pipes. Die unterschiedlichen Mechanismen haben verschiedene Eigenschaften, die bei dem Management verteilter

Anwendungen ausgenutzt werden können. Messungen haben beispielsweise ergeben, daß Shared Memory dabei nicht den besten Durchsatz liefert, jedoch gerade bei lesenden Zugriffen Anwendung und Managementsystem weitgehend voneinander entkoppelt. Welcher Mechanismus benutzt wird, wird durch die Anwendung definiert. Die weitere Verwendung wird transparent für die Anwendung durch die definierten C++-Objekte durchgeführt. Die Realisierung dieser Vorgehensweise wird in [Lemk93] und in [Jahk93] beschrieben.

Zur Durchführung von Managementoperationen gibt es zwei Vorgehensweisen. Im transparenten Fall werden die Managementoperationen nebenläufig zu der Anwendung ausgeführt. Zum Schutz von kritischen Abschnitten in der Anwendung können Managementoperationen zeitweise verzögert werden. Im zweiten Fall werden die Managementoperationen zu einem gegeben Zeitpunkt (an einem Managementinteraktionspunkt) direkt unter der Kontrolle der Anwendung durchgeführt. Beide Möglichkeiten können alternativ verwendet werden.

5.4 Das Objektmodell der Managementanwendung

Der Manager hat die gleiche Sicht auf das Objekt wie die Anwendung, d.h. er kann auf Attribute zugreifen, diese verändern, parametrisierte Aktionen auslösen und auf plötzlich eintreffende Events reagieren. Er muß sich jedoch auf Grund der Verteilung des Systems bewußt sein, daß er keine absolut konsistente Sicht auf das Objekt hat. Die dafür notwendigen Sperr- und Updateprotokolle sind für Managementaufgaben in der Regel zu aufwendig und auch nicht notwendig. Innerhalb des *Melody*-Systems wurde daher der Begriff der *beschränkten Inkonsistenz* eingeführt. Dies bedeutet, daß ein Manager immer auf ein logisches Schnappschußobjekt des originalen Objekts zugreift. Der Schnappschuß reflektiert einen Zustand des Objektes, wie er vor einer gewissen Zeit der Fall war. Der Manager kann die dabei auftretende Inkonsistenz beschränken, indem er mit der Hilfe von Aktualitätsprädikaten seine Anforderungen an die Abweichung zwischen Schnappschußobjekt und Originalobjekt spezifiziert. Beispielsweise kann ein Aktualitätsprädikat fordern, daß ein bestimmter Wert nicht älter als 20 Sekunden sein darf. Die Managementplattform stellt ihm die Informationen dann mit der geforderten Aktualität zur Verfügung.

Eine Managementanwendung gibt die Objekte, auf die sie fortlaufend zugreifen möchte, dem Managementsystem bekannt (*Attach*). Dieses holt dann die Objekte in den lokalen Cache. Die Objekte in dem Cache werden mit Hilfe der gleichen lokalen Mechanismen zur Interprozesskommunikation (d.h. shared Memory, TCP oder UNIX-Pipes) den Managern zur Verfügung gestellt, wie die Objekte in der Anwendungskomponente. Dadurch kann der Manager die Objekte lokale ansprechen. Der Agent übernimmt für die Managementanwendung die Aktualisierung der Information im Cache. Das Managementsystem wählt auf Grund der spezifizierten Aktualitätsanforderungen und auf Grund der aktuellen Situation eine geeignete Updatestrategie aus. Je nach gewählter Strategie und aktuellem Zustand wird eine Anfrage nach einem Managementattribut aus dem lokalen Cache oder von der original Quelle besorgt. Das System garantiert hierbei, daß die geforderte Aktualität eingehalten werden kann. Gleichzeitig sorgt es durch eine ständige Anpassung der Update-Strategien an die aktuelle Situation für eine Minimierung der notwendigen Netzwerknachrichten. Die Mechanismen zur Auswahl eines geeigneten Updatestrategie werden in [Kova93] beschrieben. In [Helb92] wird die dazugehörige Implementierung dokumentiert.

5.5 Ortstransparente Zugriff

Die Top-Level Managementobjekte (MO) einer Anwendung werden in einen globalen Namensraum eingetragen. Dazu wurde ein Domänenkonzept entwickelt. Domänen können entweder hierarchisch andere Domänen oder Verweise auf Managementobjekte enthalten. Die Konkatenierung der Domänennamen und des Namens des Verweises ergibt einen eindeutigen Bezeichner für eine Top-Level MO und damit für den entsprechenden Namensteil. Die Eintragung in den Domänenbaum wird durch das Managementsystem vorgenommen. Dabei kann die Anwendung bestimmen, in welche Domäne und unter welchem Namen der eigene Verweis eingetragen wird. Wird der Name des Verweises weggelassen, so wird ein künstlicher, eindeutiger Identifikator erzeugt. Wird hingegen die Domäne weggelassen, so wird auf Grund des Typs des Top-Level MOs (welcher auch den Typ der Anwendungskomponente angeben soll) eine Standarddomäne ausgewählt. Mehrere Domänen für eine Komponente sind möglich, ebenso das

Abfragen des Domänenbaum. Im Augenblick ist der Domänenbaum mit Hilfe des Directory-Dienstes des DCE realisiert. Bei der Anmeldung eines Objektes durch die Managementanwendung (*Attach*) erfolgt die notwendigen Namensresolution und die Objektlokalisation.

6. Zusammenfassung und Ausblick

In diesem Bericht wurden die Grundkonzepte für das Management verteilter Anwendungen eingeführt und ausführlich motiviert. Es wurden die in dem *Melody*-Managementsystem realisierten Konzepte dargestellt. Anwendungskomponenten können geeignete instrumentiert werden, so daß sie für Managementaufgaben verfügbar sind. Dabei wurde auf eine strikte Trennung zwischen den Aufgaben der Anwendungskomponente und den Managementaufgaben geachtet. Mit Hilfe eines objekt-orientierten Informationsmodells und der Realisierung entsprechender Objekte erfolgt die Modellierung der Managementinformation und die lokale Realisierung. Managementaktivitäten können nebenläufig zu den normalen Aktionen einer Anwendung durchgeführt oder bewußt von der Anwendung zugelassen werden.

Für eine Managementanwendung wurde ein ortstransparenter und effizienter Zugriff unter Berücksichtigung von Aktualitätsanforderungen realisiert. Eine Managementanwendung bestimmt die Objekte, die in ihrem Sichtbarkeitsbereich liegen. Die notwendigen Aufgaben zur Objektlokalisierung, zum Zugriff auf ein Objekt, zur Realisierung eines Caches und zur Abwicklung der notwendigen Kommunikationsaufgaben werden von den Managementagenten übernommen.

Bisherige Erfahrungen mit dem gewählten Ansatz zeigen, daß die Instrumentierung der Anwendung für einen Anwendungsprogrammierer gut zu verstehen und leicht einzusetzen ist. Damit wird es möglich, bereits zum Entwicklungs- und Implementierungszeitpunkt den Managementaspekt in eine Anwendung zu integrieren.

Auch die Entwicklung der Managementanwendungen konnte deutlich vereinfacht werden. Die gewählten Abstraktionen in der Form des Domänen-Konzepts und der Aktualitätsprädikate ermöglichten eine einfache Realisierung. Trotzdem müssen hier noch weitere Anstrengungen unternommen werden, um eine Managementanwendung geeignet zu unterstützen. Die Funktionen von Triggern in dem Managementsystem kann noch wesentlich erweitert werden. Beispielsweise sollte es möglich sein eine Sicht auf das verteilte System zu definieren, welche automatisch alle zu einer Anwendung gehörende Objekte enthält. Das Managementsystem kann dann geeignete Maßnahmen ergreifen, um neu erschaffene Objekte zu entdecken und in den Sichtbarkeitsbereich einzutragen. Ein solcher Mechanismus würde weitere Vereinfachungen für eine Managementanwendung mit sich bringen. In diesem Bereich sind noch viele Probleme ungelöst.

In weiteren Experimenten wird untersucht, wie mit Hilfe der entwickelten Mechanismen eine automatischen Instrumentierung einer verteilten Anwendung durchgeführt werden kann. Dies sollte gerade bei Anwendungen, die mit Hilfe einer Entwicklungsplattform wie dem DCE entwickelt werden, durch eine Modifikation der erzeugten Stubs durchaus möglich sein. Wir versprechen uns davon weitergehende Erkenntnis über die Art der Managementinformation, die von einer Anwendung zur Verfügung gestellt werden kann.

Literaturverzeichnis:

[ANSA91] ANSA. ANSAware 3.0 Implementation Manual. Manual RM.097.01, Architecture Projects Management Limited, February 1991.

[BiNe84] Andrew D. Birrell, Bruce Jay Nelson. Implementing Remote Procedure Calls. *ACM Transactions on Computer Systems*, 2(1):39–59, February 1984.

[Geih92] K. Geihs. OMG Object Request Broker. *PIK - Prxis der Informationsverarbeitung und Kommunikation*, S. 244–245, Dezember 1992.

[HAV93] H.-G. Hegering, S. Abeck, R. Valta. Integriertes Netzmanagement - Konzepte und Realisierung. In

Kommunikation in verteilten System (KiVS'93), Munich, März 1993 . Gesellschaft für Informatik.

[HeAb93] Heinz-Gerd Hegering, Sebastian Abeck. *Integriertes Netz- und Systemmanagement*. Addison-Wesley, Bonn, 1993.

[Hege91] H.-G. Hegering. Die Problematik einer Management Information Base (MIB) für ein integriertes Netzmanagement. *Informatik - Forschung und Entwicklung*, 6(4):171–185, 1991.

[Helb92] Tobias K. Helbig. Strategien zur Einhaltung von Aktualitätsanforderungen. Diplomarbeit Nr. 944, Universität Stuttgart, Institut für Parallele und Verteilte Höchstleistungsrechner, Stuttgart, Dezember 1992.

[ISO92a] ISO. ODP List of Open and Resolved Issues - June 1992. *Arbeitspapier des ISO/IEC JTC1/SC21/WG7: N7057*, May 1992.

[ISO92b] ISO. Working Draft for the Basic Reference Model of Open Distributed Processing - Part 1-4. Draft, ISO/IEC JTC 1/SC 21, May 1992.

[Jahk93] Thilo Jahke. Ein objektorientiertes Managementmodell für verteilte Anwendungen. Diplomarbeit Nr. 984, Universität Stuttgart, Institut für Parallele und Verteilte Höchstleistungsrechner, Stuttgart, Dezember 1993.

[Kova93] Ernö Kovacs. Automatic Selection of an Update Strategy for Management Data. In *Proceedings of the IEEE First International Workshop on Systems Management (IWSM'93)*, Los Angeles, April 1993.

[KPW91] Graham Knight, George Pavlou, Simon Walton. *Experience of Implementing OSI Management Facilities*, S. 259–270. Elsevier Science Publishers B.V. (North-Holland), 1991.

[Lemk93] Björn Lemke. Lokale Kommunikation in einem verteilten Managementsystem. Studienarbeit Nr.981, Universität Stuttgart, Institut für Parallele und Verteilte Höchstleistungsrechner, Stuttgart, April 1993.

[MCWB91] Keith Marzullo, Robert Cooper, Mark D. Wood, Kenneth P. Birman. Tools for Distributed Application Management. *COMPUTER*, August 1991.

[OMG91] OMG. The Common Object Request Broker: Architecture And Specification. Technischer Bericht 91.12.1, Object Management Group, December 1991.

[OSF91] OSF. *Introduction to OSF DCE*. OSF, Open Software Foundation, 11 Cambridge Center, Cambridge, MA 02142, 1.0 edition, December 1991.

[Schi92] A. Schill. Das OSF Distributed Computing Environment. *Informatik Spektrum*, 15(6):333–334, Dezember 1992.

[SyTa93] Mark Sylor, Owen Tallman. Applying Network Management Standards to System Management: the case for the Common Agent. In *IEEE First International Workshop On Systems Management*, Los Angeles, April 1993. IEEE.

[VaJa93] Robert Valta, Riaan De Jager. Deploying Group Communication Techniques in Network Management. In H.-G. Hegering, Y. Yemini (Hrsg.), *Integrated Network Management, III*, S. 751–761. IFIP Transactions, North-Holland, April 1993.

[ZFD93] Martin Zimmermann, Magdalena Feldhoffer, Oswald Drobnik. Verteilte Anwendungen: Entwurf und Realisierung. *PIK - Prxis der Informationsverarbeitung und Kommunikation*, S. 62–69, Januar-März 1993.

Literatur-Verweis:

Diese Veröffentlichung erschien als:

[Kova93] Ernö Kovacs. Das MELODY-Managementsystem f"ür verteilte Anwendungen. In Prof. Dr. K. Geihs Prof. Dr. O. Drobnik (Hrsg.), *Tagungsband zum Arbeitstreffen der GI/ITG "Entwicklung und Management verteilter Anwendungssysteme"*. GI/ITG, Oktober 1993.