TECHNISCHE
UNIVERSITÄT
MÜNCHEN

INSTITUT FÜR INFORMATIK

Sonderforschungsbereich 342:
Methoden und Werkzeuge für die Nutzung
paralleler Rechnerarchitekturen

Strategien für verteiltes Last- und Ressourcenmanagement

Niels Reimer

TUM-I9818
SFB-Bericht Nr. 342/08/98 A
August 98

TUM–INFO–08-I9818-150/1.–FI

Anforderungen an: Prof. Dr. A. Bode
Sprecher SFB 342
Institut für Informatik
Technische Universität München
D-80290 München, Germany

Druck: Fakultät für Informatik der
Technischen Universität München

Strategien für verteiltes Last– und Ressourcenmanagement

Niels Reimer

reimer@informatik.tu-muenchen.de

20. August 1998

Zusammenfassung

Im Rahmen des MoDiS–Projektes, das sich mit der Konstruktion verteilter Systeme insbesondere verteilter Betriebssysteme beschäftigt, spielen Ressourcenmanagementstrategien eine wesentliche Rolle. In diesem Bericht wird dieser Bereich im Kontext des Gesamtsystems näher dargestellt. Wesentliche Ausgangspunkte sind hierbei eine Klassifikation von Strategien hinsichtlich ihrer Eigenschaften und Kosten, Metriken für strategische Entscheidungen, ein Systembeschreibungsmodell aus dem sich nicht nur der derzeitige Zustand, sondern auch die Wege der zukünftigen Entwicklung ableiten lassen. Dies alles wird verflochten mit der Erzeugung und Nutzung von Ressourcenangebots- und -anforderungsprofilen, so daß auf dieser umfassenden Informationsgrundlage sinnvolle Entscheidungen gefällt werden können. Weitergehend wird ein Ansatz zur Managementverbesserung durch Adaption skizziert. Ein Ausblick auf weiter zu vertiefende Aspekte und auszuführende Realisierungsarbeiten beschließt den Bericht.

Inhaltsverzeichnis

1 Einleitung

Dieser Bericht soll den aktuellen Stand meiner Arbeiten im Bereich des verteilten Ressourcenmanagements zusammenfassen und dabei das Augenmerk auf die Aspekte lenken, die mit den strategischen Entscheidungen in Zusammenhang stehen. Es wird für den folgenden Bericht davon ausgegangen, daß der Leser mit den Konzepten und Gedanken von verteilt, parallelen kooperativen Systemen (VPK-Systemen) hinreichend vertraut ist. Im Bericht [1] und Antrag [2] zur Förderperiode 1998 bis 2000 des SFB 342 findet sich eine umfassende Darstellung dazu im Abschnitt des Teilprojektes A8. Der vorliegende Bericht betrachtet das verteilte Ressourcenmanagement stets im Kontext von VPK-Systemen mit ihrer geschachtelten Strukturierung und Dynamik. Formuliert werden diese Systeme in der Sprache INSEL, deren Konzepte sich in [3] finden. Eine weitere einführende Veröffentlichung ist [4]. Die Syntaxdefinition der Sprache INSEL ist in [5] festgelegt. Es soll betont werden, daß hier nicht der Anspruch erhoben wird, eine in sich geschlossene und vollständige Arbeit zu präsentieren. Dieser Bericht soll nur den aktuellen Stand dokumentieren, und daher darf es nicht verwundern, wenn an zahlreichen Stellen offene Fragen bleiben und nur der verbindende Grundgedanke dargestellt wird.

1.1 Gliederung

Dieser Bericht gliedert sich im weiteren wie folgt: Im nächsten Abschnitt werden Begriffe einführend definiert und erklärt, die im gesamten Bericht gebraucht werden. Die restlichen Begriffe werden jeweils an der Stelle ihres ersten Auftretens oder zu Beginn des entsprechenden Abschnittes eingeführt. Es folgt die Darstellung des grundsätzlichen Gedankengangs bei dem die Konzeption des verteilten Ressourcenmanagements beschrieben wird und wie unser Ansatz der Problemstellung begegnen will. Daraus leiten sich dann kanonisch zwei Hauptrichtungen ab. Erstens ist die Fragestellung nach dem direkten Ressourcenmanagement der betreuten Aktivitäten wichtig. Zweitens eröffnet sich mit Betrachtung des Prozesses der Entscheidungsfindung das Gebiet der Verbesserung und Anpassung des Managements an sich.

Als Grundlage für beide Richtungen wird im Abschnitt 2.1 ein Beschreibungsmodell des Systemzustandes und der möglichen Systemfortschritte eingeführt. In diesem Modell werden Ereignisse präzisiert, die dann Auslöser von Managementsmaßnahmen sind. Die Erfassung des anschließend eingeführten Berechnungsfortschrittes orientiert sich ebenfalls an den Ereignissen. Der folgende Abschnitt 2.4 widmet sich dann den Punkten Metriken, Kosten und Quantifizierbarkeit der Ressourcenverfügbarkeit. Ausgehend von dieser nun geschaffenen begrifflichen und modellierungstechnischen Grundlage auf hohem Abstraktionsniveau wird am Beispiel einer Lastplazierungs- oder Lastmigrationsentscheidung eine Strategie des direkten Ressourcenmanagements bis hinunter zu sehr konkreten Aspekten beschrieben. Dieser Abschnitt erklärt die Begriffe und die Nutzung der Anforderungs- und Ressourcenprofile. Der folgende Abschnitt widmet sich dann wieder auf höherem Abstraktionsniveau der Verbesserung des direkten Managements. Dazu wird eine Klassifikation der Strategien motiviert und eingeführt. Anschließend wird auf die Adaption und die Aspekte ihrer Aufwandsabschätzung eingegangen. Zusammenfassend beschließen dann die Abschnitte 4 und 5 diesen Bericht und zeigen die weiteren geplanten Schritte und offene Fragestellungen auf.

1.2 Begriffe

Hier werden nun einige zentrale Begriffe eingeführt. Das Management gliedert sich in zwei Bereiche, erstens das **direkte Ressourcenmanagement** (RM) und zweitens das **Meta-Management** (MM). Diese beiden Begriffe werden durch die funktionale Sichtweise unterschieden. Unter das RM fällt alles, was die Fragen: „Was passiert?“ und „Wo passiert etwas?“ beantwortet. Das MM hingegen gibt Antwort auf die Fragen „Wie wird entschieden?“ und „Wie kann der Entscheidungsprozeß verbessert werden?“.

Damit ist das RM für die Bedürfnisse der Objekte verantwortlich und versorgt diese mit allen Ressourcen, die benötigt werden. Das bedeutet Plazierungs– und Migrationsentscheidungen fallen in die Zuständigkeit des RM. Gleiches gilt für die Zuteilung virtueller Adressen, Zugriff und Realisierung geteilter Objekte, Rücknahme freiwerdender Ressourcen und ähnliches mehr.

Das MM hingegen dient zur Verbesserung des RM. Diese Verbesserungen beziehen sich nicht nur auf die Qualität der Entscheidungen, sondern auch auf die damit verbundenen Kosten und Leistungsmerkmale, die mit den einzelnen Strategien einhergehen (Aufwand und Zeit zur Bestimmung der Entscheidung, Haltung und Erhebung von Daten). Eine **Strategie** ist ein algorithmisches Verfahren, das das Vorgehen der Entscheidungsfindung gemäß gewisser Zielvorstellungen beschreibt.

So wird im Rahmen des MM durch Beobachtung, Protokollierung, Extrapolation und Managementkooperation der Entscheidungsprozeß durch Flexibilisierung (Strategiewahl) und Anpassung der Strategieparameter dynamisch verbessert. Dieser Prozeß wird **Adaption** genannt und nutzt die Möglichkeiten des Managementinstrumentariums systematisch. Dabei soll unter **Strategiewahl** verstanden werden, daß aus zur Verfügung stehenden Strategien eine Entsprechende ausgewählt wird und für die zukünftige Entscheidungsfindung benutzt werden soll. Bei der **Strategieanpassung** hingegen werden Parameter der Strategie (z. B. Schwellwerte und Gewichte, die Größe des berücksichtigten Zeitraums, innerhalb dessen Ereignisse eine Auswirkung auf aktuelle Entscheidungen haben sollen; usw.) angepaßt.

In diesem Zusammenhang ist eine Klassifikation der Strategien erforderlich, um eine sinnvolle Strategiewahl bzw. Anpassung der Parameter methodisch zu fundieren. Dies wird später im einleitenden Abschnitt des Meta-Managements ausführlich dargestellt.

2 Konzepte für das Ressourcenmanagement

Insbesondere vor dem Hintergrund, daß Ressourcenkonsumenten wechselnde Anforderungen haben, für die es keine universelle Strategie geben kann, erscheint das Beschreiten des evolutionären Weges für ein Ressourcenmanagement als sinnvoll. Durch dieses Verfahren folgt man dem Wunsch, dem Optimum möglichst nahe zu sein oder sich zumindest beständig anzunähern. Vorteilhaft ist in diesem Zusammenhang der Gesamtsystemansatz, da alle Information und sämtliche Einflüsse bekannt sind bzw. erfaßt werden und darauf gezielt reagiert werden kann. Insbesondere erlaubt der Ansatz in Zusammenhang mit dem evolutionären Weg ein dynamisches Einschränken und Erweitern des Blickwinkels auf die relevante Information. Wesentliches Kennzeichen dieses Weges ist der fortwährende Wandel durch Anpassung und/oder Ersetzung der verwendeten Strategien. Hierbei sind mehrere Dinge im Vorfeld zu untersuchen: Erstens ist eine Modellierung des Systemzustands und seiner möglichen Entwicklung sowie des Systemfortschritts zu konstruieren. Zudem muß geklärt sein, welche Ereignisse Auslöser für die Ressourcenmanagementmaßnahmen (RMM) und Strategieänderungen sind. Dabei wird die Verknüpfung der Systemfortschritte mit den für das Management relevanten Ereignissen genutzt. Zweitens muß Beurteilung des Berechnungsfortschrittes erreicht werden. Zusätzlich sind zahlreiche Metriken und zu erfassende Kosten zu definieren.

Darüberhinaus sind die wesentlichen Mechanismen und Instrumente für das Management zur Verfügung zu stellen. So bedarf es der Möglichkeit Teile des Codes der Strategien auszutauschen, neu zu übersetzen und ins laufende System einzubinden. Ebenso gilt für Parameter der Strategien und Mechanismen, die sonst üblicherweise vor äußeren Zugriffen geschützt sind, daß sie unter gewissen Umständen veränderbar sein müssen. Desweiteren finden zahlreiche strategische Entscheidungen in diesen Managementinstrumenten statt, so daß bei der angestrebten Integration von Management und den verfügbaren Werkzeugen (Übersetzer, Binder, Modifikation von Kernabläufen usw.) eine exakte Trennung bzw. Lokalisierung der Strategie schwierig ist. Für einen ersten Schritt wird beim derzeitigen Stand der Arbeit vereinfachend davon ausgegangen, daß der abstrakte Manager einer Akteursphäre die wesentlichen Entscheidungen zentral fällt und selbst noch nicht in und auf die vielfältigen Werkzeuge zerfallen und verteilt ist. Damit sind Strategieänderungen einfacher erreichbar. Wenn der Gedanke der Zersplitterung des Managements umgesetzt werden soll, so betrachte man einfach die im jeweiligen Werkzeug realisierte Strategie getrennt vom restlichen System und findet dann Verhältnisse vor, bei der der außenstehende abstrakte Manager Anweisungen gibt, wie die Strategie gezielt zu verändern ist.

Am Beispiel der Bewertung der Stellen des verteilten Systems wird dann die Abstraktionsebene verlassen und dargestellt, wie eine Strategie zur Auswahl eines Plazierungsziels vorgeht. Dabei werden die verschiedenen Ressourcenanforderungen des Lastobjektes und des aktuellen Systemzustandes berücksichtigt. Die Entscheidung bzgl. der verwendeten Information und des betrachteten Zeitraumes wirkt sich wiederum auf die Kosten und damit auf die Beurteilung des eigenen Verhaltens aus. All diese Punkte werden in den folgenden Abschnitten beschrieben.

2.1 Systemmodellierung

Das anfangs erwähnte, grundlegende Modell, das den Zustand, die Struktur und den Fortschritt des gesamten Systems beschreibt, wird nun vorgestellt. Für die beiden Punkte Systemzustand und –struktur erscheint die Repräsentation als Graph am geeignetesten. Der Systemfortschritt wird daher am besten durch ein Graphersetzungssystem beschrieben. Dabei steckt insbesondere in der Darstellung der Systemstruktur das Wissen über die zukünftige Entwicklung des Systems. Der Graph und sein Ersetzungssystem wird im folgenden eingehend beschrieben und an einem Beispiel wird die Entwicklung bzw. der Ablauf eines Systems dargestellt. Beim derzeitigen Stand der Arbeit werden nur Systeme betrachtet, die die Aufspaltung in und die abschließende Synchronisation von den einzelnen nebenläufigen Unterverbänden kennen. Hierbei wird durch Parameterübergabe kooperiert und dies im folgenden IT–Kooperation (Initiierungs–Terminierungs–Kooperation) genannt.

Die Knoten des Graphen sind abstrakte oder konkrete Ereignisse. Seine Kanten entsprechen der „ereignet–sich–bevor"–Relation der Ereignisse, wobei untere Ereignisse vor oberen Ereignissen stattgefunden haben.

Unter **abstrakten Ereignissen** sind Strukturen des Systems zu verstehen, die noch nicht ausgeprägt sind. Erst mit ihrer schrittweisen Konkretisierung verfeinert sich die Gliederung des Systems. Die Systemkomplexität steigt im Zuge der Verfeinerung, wobei die Freiheitsgrade durch die schrittweise Festlegung der Systemausprägung reduziert werden. Sinn und Zweck der abstrakten Ereignisse ist es, die Zukunft des Systems mittels seiner Strukturierung zu beschreiben. Sie werden mit Großbuchstaben notiert und sind hier aufgelistet:

V : Verband

D : Durchführung von Berechnungen

E : Ende eines Verbandes

S : Synchronisation nebenläufiger Verbände

Konkrete Ereignisse werden mit Kleinbuchstaben notiert. Sie beschreiben durchgeführte Schritte des Systems und stellen so die Vergangenheit dar. Es sind im einzelnen:

b : Beginnereignis eines Verbandes
(Initialisierung des Akteurs, Einreihung in Warteschlangen zur Verwaltung, ...)

d : Durchführung einer Berechnung

e : Endeereignis eines Verbandes
(weitestgehende Ressourcenfreigabe, Signalisierung des Endes an den übergeordneten Vorgänger, ...)

a : Aufspaltung in mehrere nebenläufige Verbände
(Plazierung der Objekte, Parameterübergabe, ...)

s : Synchronisation beim Abschluß nebenläufiger Verbände
(Parameterrückgabe, ...)

Der Umfang der Berechnung, die mit dem Ereignis d verbunden ist, hat eine noch festzulegende Granularität, die sich insbesondere auf die Schärfe der Vorhersagen und den damit verbundenen Aufwand auswirkt. Beispielsweise können darunter alle Operationen einer Schleifeniteration verstanden werden, so daß der damit verbundene Aufwand als Basis für die Abschätzung der Dauer weiterer Schleifeniterationen bzw. d-Ereignisse betrachtet werden kann. Ebenso kann das d-Ereignis nur eine einzelne Programm–Anweisung[1] repräsentieren oder auch einzelne Teile der kanonischen Operation darstellen. Damit steigt die Auflösung der Betrachtung jedoch auch der Verwaltungsaufwand der Beobachtungsdaten. Diese Skalierbarkeit ist eine Stärke des Modells, da man die Granularität den letztlich noch nicht exakt präzisierbaren Erfordernissen des Managements anpassen kann.

Die Graphersetzungsregeln sind nachfolgend in Abbildung 1 dargestellt.

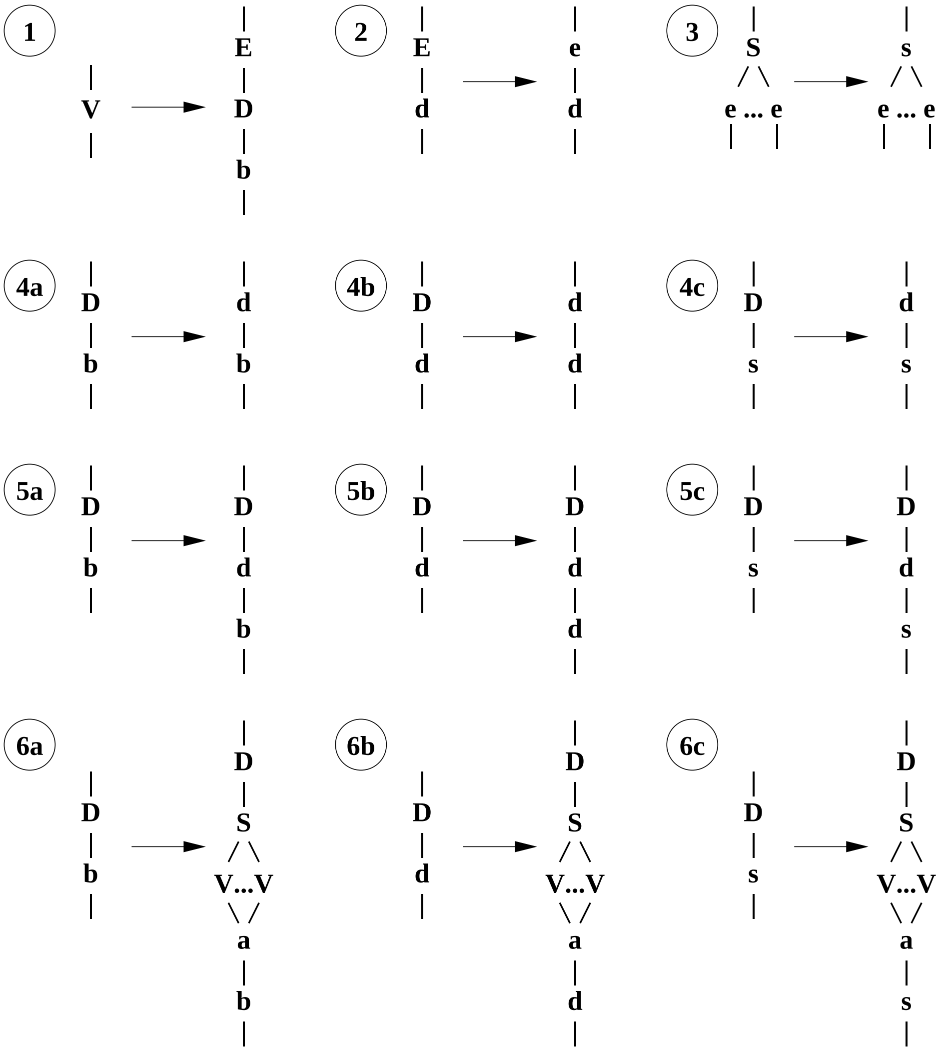

Abbildung 1: Die Ersetzungsregeln für den Systembeschreibungsgraphen

Exemplarisch sei hier Regel 1 näher erklärt. Es kann jederzeit das abstrakte Ereignis V, das stellvertretend für einen ganzen Verband steht, ersetzt werden und in ein konkretes Beginnereignis b umgewandelt werden, dem eine abstrakte Durchführung einer

[1] und damit im Extremfall sogar ein leeres Stück Programmcode

Berechnung D mit abschließendem Endeereignis E des nun detailierter ausgeprägten Verbandes folgt.

Anzumerken ist noch, daß bei den Regeln 3 bzw. 6a-c deuten die „ ... “ an, daß eine Zusammenfassung von bzw. eine Aufspaltung in mindestens zwei bis zu beliebig vielen Pfaden erfolgt. Dabei gilt für Regel 3, daß alle Ereignisse direkt vor dem S–Ereignis e–Ereignisse sein müssen. Alle Ereignisse in den Regeln 6a-c, die auf das a–Ereignis folgen sind V–Ereignisse.

In die Konstruktion der Regeln ist die Überlegung eingegangen, daß jede Graphersetzung einem Schritt des Systemfortschritts entsprechen soll. Daher sind alle Regeln so aufgebaut, daß eine Ersetzung dem Erreichen genau eines konkreten Ereignisses entspricht.

In den Abbildungen 2 und 3 ist die schrittweise Abarbeitung eines solchermaßen beschriebenen Systems dargestellt.

Im Startzustand existiert nur die abstrakte Vorstellung des Systems als Verband. Ausgehend von diesem Stand verfeinert und konkretisiert sich dann mit jedem Fortschreiten des Systems seine Strukturierung. Dabei wird einerseits die Vergangenheit durch konkrete Ereignisse festgeschrieben und andererseits die zukünftige Systementwicklung durch die strukturierenden abstrakten Ereignisse weiterausgeprägt.

Bei Betrachtung des Beispiels ist deutlich sichtbar, wie der Ereignishorizont sich schrittweise nach „oben“ d. h. auf das abschließende Endeereignis des Verbandes hin verschiebt. Die abstrakten Ereignisse, die durch die Ersetzung umgewandelt werden, sind durch die Hinterlegung eines Rahmens hervorgehoben. Die Stellen an denen ein Systemfortschritt stattfinden kann bzw. wo eine Graphersetzungsregel angewendet werden kann, lassen sich im Graphen leicht auffinden. Es sind jeweils die abstrakten Ereignisse deren eingehenden Kanten (das heißt alle Kanten unterhalb des abstrakten Ereignisses) den Ereignishorizont überschreiten. Die Anzahl der möglichen Stellen zur Anwendung einer Ersetzungsregel ist zugleich ein Maß des Grades der ausgeschöpften Parallelität.

2.2 Auslösende Ereignisse für Managementmaßnahmen

In unserem Gesamtsystemansatz wird stets eine top-down–orientierte Entwicklung verfolgt. Dies spiegelt sich sowohl in dem Entwurf zur Systembeschreibung wieder als auch in der Entwicklung des Managements. Zuerst wurde ein System mit Hilfe von Ereignissen modelliert und im Anschluß werden hier die relevanten Ereignisse isoliert, die Managementmaßnahmen auf der Ebene des RM und des MM auslösen sollen. Prinzipiell stellen alle beobachtbaren Ereignisse einen irgendwie gearteten Berechnungsfortschritt dar und sind somit für das Management von Interesse. Auf der anderen Seite ist auch klar, daß bei einer Ressourcenanforderung während einer Berechnung dies natürlich eine Managementmaßnahme des RM erfordert. Wenn aber jedes kleine Ereignis mit Erfassung, Auswertung und Verwaltung verbunden wird, so ist schnell klar, daß die daraus resultierenden Kosten in keinem rentablen Verhältnis zum erzielbaren Nutzen stehen. Damit ist eine Zusammenfassung der Ereignisse und eine Einschränkung auf das Wesentliche nötig.

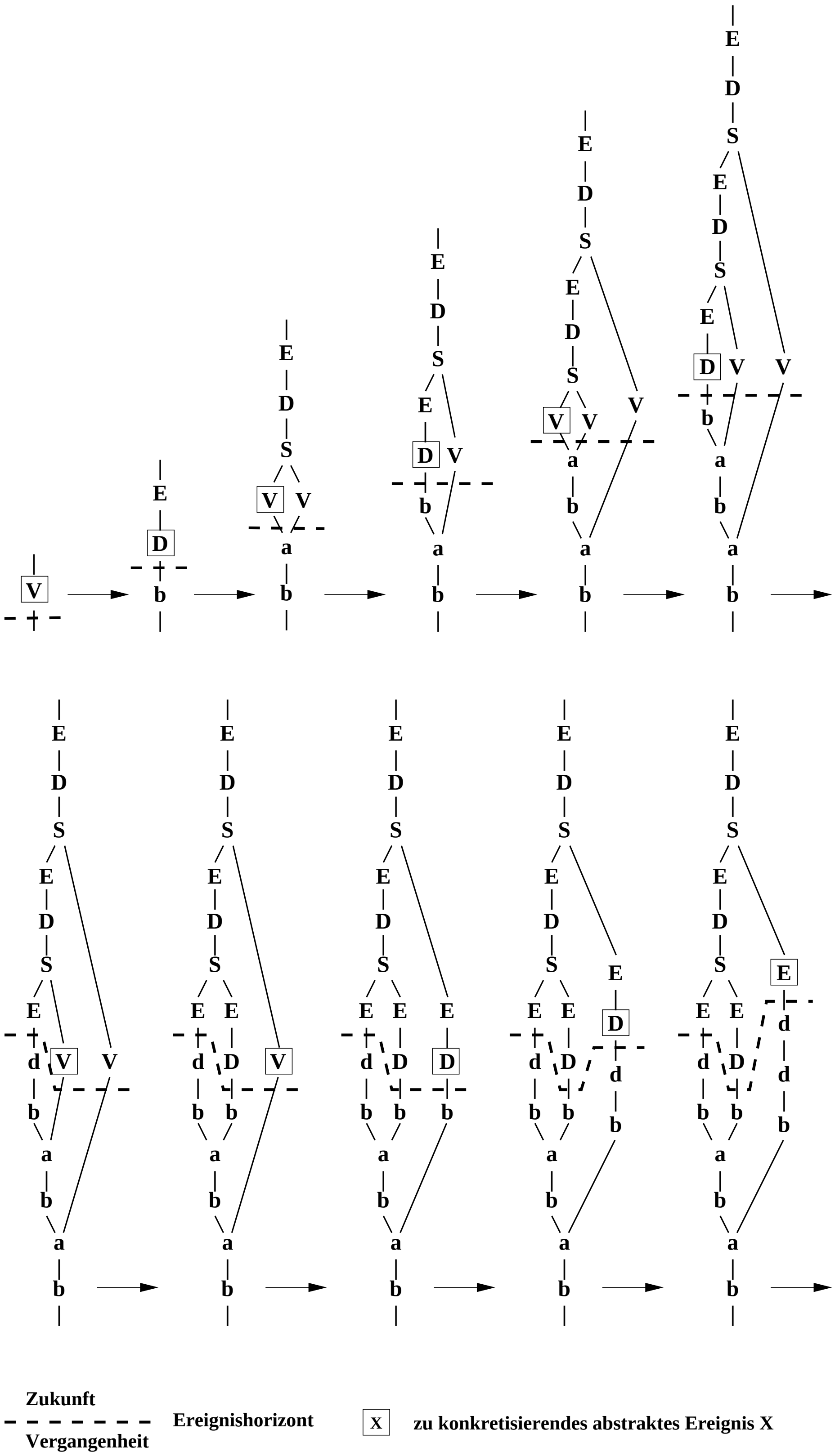

Abbildung 2: Beispiel eines Systemablaufes

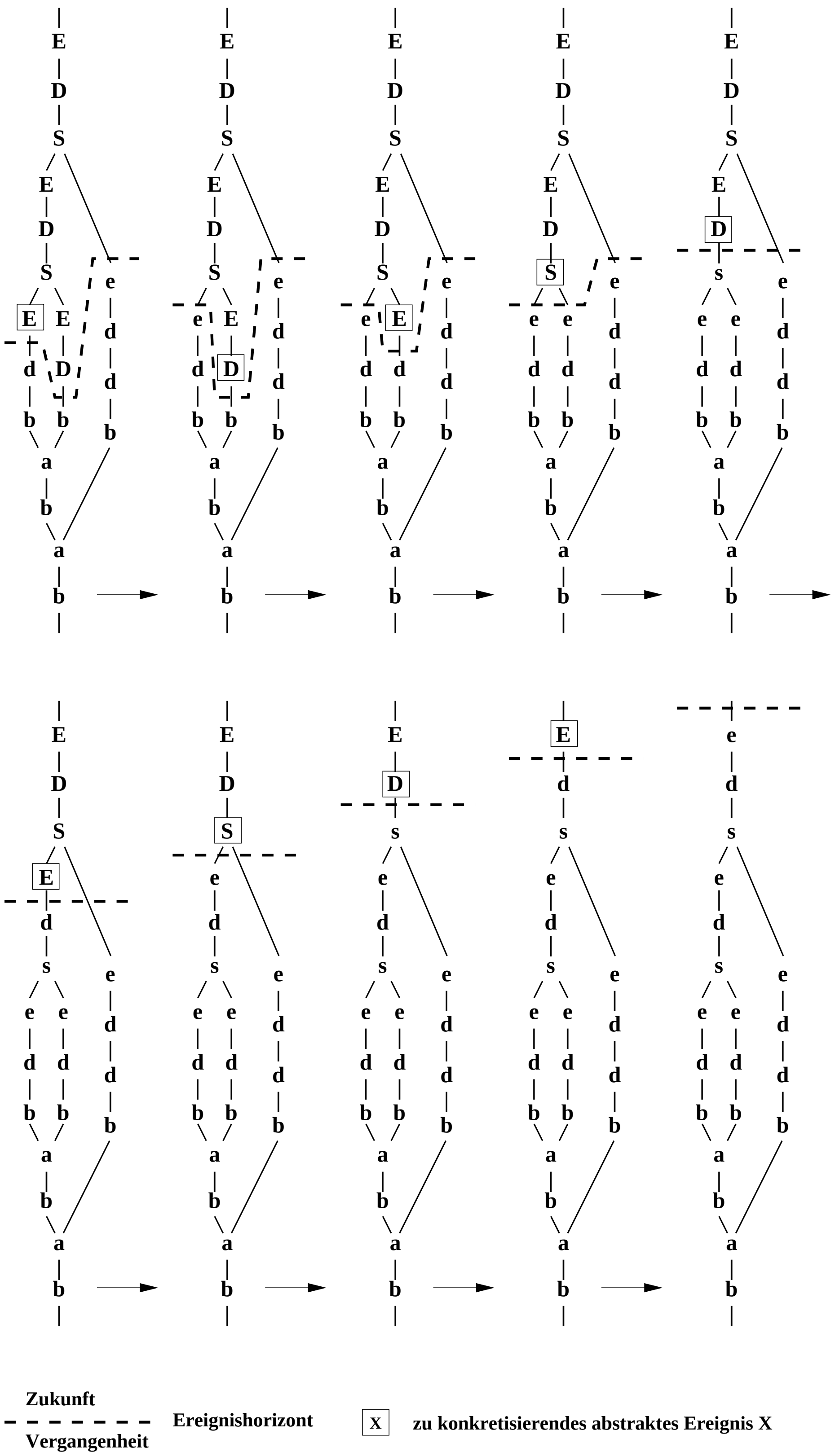

Abbildung 3: Fortsetzung: Beispiel eines Systemablaufes

Ereignis und Beschreibung		Auswirkung
a	**Komponentenplazierung** Akteur, Depot, Order	Änderung der Systemsicht Aktualisierungsaufwand
b,d	**Komponentenerzeugung** Akteur, Depot, Order	Komponentenspezifisches Lasterzeugungsprofil
d	**Komponentennutzung** Depot (Orderaufruf), K-Akteur	Komponentenspezifisches Nutzungsprofil zur Bewertung und Verbesserung der Plazierungs- und Migrationsentscheidung hinsichtlich erfolgreicher Clusterbildung
e	**Komponentenauflösung** Akteur, Depot, Order	Komponentenspezifisches Lastprofil
e,s	**Beenden von Teilberechnungen** Erreichen von Sicherungspunkten, Ende von Synchronisationsblöcken, ...	Systemfortschritt und Extrapolation
	Empfang/Versand von Lastindizes (ereignis-/zeitbasiert)	Änderung der Systemsicht Aktualisierungsaufwand
	Empfang von Lokationsinformation	Clustering von Workgroups bzw. Bewertung der Lastverteilungsalgorithmen und Nutzung der Strukturinformationen in INSEL

Tabelle 1: Zuordnung von konkreten Ereignissen zu den Vorkommnissen in einem System und den betroffenen Informationsträgern

Im Querschnittsthema „Anwendungsbezogene Lastverteilung“ [1] wurde die Frage nach den wesentlichen Ereignissen ebenfalls erörtert, wobei hier das derzeitige Ziel ist, ein Werkzeug zur Beobachtung der Lastverteilung zu entwickeln. Als Beitrag zu dieser Zusammenarbeit entstand die Tabelle 1, die die wesentlichen Vorkommnisse in einem System und die davon betroffenen Informationsträger auflistet und stichwortartig beschreibt.

Eine Zuordnung von konkreten Ereignissen zu den beschriebenen Vorkommnissen ist im oberen Teil der Tabelle 1 dargestellt. Beispielsweise kann in einem d-Ereignis eine Komponentenerzeugung oder -nutzung stattfinden. Im unteren Teil der Tabelle sind Ereignisse beschrieben, die durch andere Ereignisse erst ausgelöst werden oder periodisch erfolgen. Sie können nicht einem konkreten Ereignis zugeordnet werden. Als Beispiel kann durch eine Lasterzeugung der Systemzustand deutlich verändert werden, was dann unter Umständen propagiert wird. Ein anderes Beispiel ist eine Komponentennutzung, die zur Mitteilung von Lokationen führt, was dann Migrationsbewertungen und ggf. die eigene, neue Strategiebewertung auslöst.

Das a-Ereignis erfordert eine Plazierungs- bzw. Lastverteilungsentscheidung der abzuspaltenden Verbände; in Betracht kommende Stellen werden gemäß der Anforderungs- und Ressourcenprofile (Abschnitt 2.5) bewertet. Darüberhinaus muß entschieden werden, wie die zu erzeugende Komponente realisiert wird. Diese Fragestellung obliegt aber gleichrangig dem Gebiet der Strategien und der flexibilisierten Übersetzungs- und Bindetechniken. Hier muß die Informationsgrundlage kooperativ ausgetauscht werden, so daß einerseits entsprechende Kostenabschätzungen vorliegen und andererseits die benötigten Direktiven ausgegeben werden.

Beim Auftreten eines b-Ereignisses wird die entsprechende Komponente realisiert, das heißt die Codeerzeugung, Speicherzuteilung und -initialisierung, Einreihung in die Warteschlangen der Ausführungsumgebung, die Erarbeitung des Deklarationsteils usw. erfolgt. Hier werden dann nicht die einzelnen Stellen als Ressource zugeteilt, sondern die auf ihnen befindlichen Ressourcen wie z. B. virtuelle Adressen bzw. deren Einblendung in den physikalischen Speicher belegt. Zur Ausführung gelangt die Komponente allerdings erst mit dem nachfolgenden Ereignis (a oder d).

Für d-Ereignisse kann wenig Verbindliches gesagt werden, da sich durch die noch zu wählende Granularität sehr viele Berechnungsschritte und daraus resultierende RMM ergeben können. Unabhängig von der Granularität erfolgt jedoch die Aufzeichnung des Berechnungsfortschrittes, wie sie im Abschnitt 2.3 näher ausgeführt wird.

Das Erreichen eines e-Ereignisses führt zur weitestgehenden Ressourcenfreigabe und der Mitteilung an die übergeordnete Komponente. So werden nur noch die Ressourcen belegt, die für die Durchführung der Abschlußsynchronisation gebraucht werden. Erst mit diesem Schritt wird das s-Ereignis ausgelöst, wobei danach die Ressourcen endgültig freigegeben werden. Stellengebundene Ressourcen wie z. B. der Prozessor werden einfach abgegeben, wohingegen Ressourcen mit Bindung an eine Akteursphäre wie z. B. virtuelle Adreßbereiche an die übergeordnete Komponente zurückfallen. Mit dem Ende der Lebenszeit einer Komponente kann dann auch ihr Laufzeitmanagementanteil aufgelöst werden, nachdem die zu tradierende Information weitergereicht wurde.

Diese Bündelung der verschiedenen Aktionen, die sich hinter einem konkreten Ereignis verbergen, zeigt deutlich, wie die Abgrenzung der Ereignisse untereinander zu verstehen ist. Zahlreiche kleine Aktionen des RM müssen dabei gar nicht einzeln an Ereignisse gebunden werden, sondern es genügt sie als Paket von Einzelaktionen zusammenzufassen und dieses dann mit einem konkreten Ereignis zu verbinden. Die dabei gesammelte Information wird für MM–Entscheidungen später genutzt.

2.3 Maß für den Berechnungsfortschritt

Die Abschätzung wie lange eine Aktivität noch rechnen wird, ist schwierig. Das Systembeschreibungsmodell gibt etwas Hilfe bzgl. der zu erwartenden Systemstruktur bzw. ihrer Abhängigkeiten. Daher wird ergänzend ein Maß benötigt, das dem Management anzeigt, wie lange eine Berechnungseinheit noch dauern wird. Dies ist aber nur als Schätzung möglich, wobei von dem Status, wie weit die Berechnung z. B. eines Verbandes schon fortgeschritten ist, um so mit Hilfe von früher gemachten/aufgezeichneten Beobachtungen die voraussichtliche Berechnungsdauer zu extrapolieren. Es liegt in der Natur der Dinge, daß diese Information unzuverlässig und unscharf ist. Als Meilensteine des Berechnungsfortschritts bieten sich zudem nur wenige Ereignisse bzw. wesentliche Zustände an. Diese werden gemäß ihrer Aussagekraft hinsichtlich der Monotonie des Fortschritts in *streng monoton* und *monoton* klassifiziert, wobei der Bezug zur Formulierung des Systems in INSEL geschaffen wird.

streng monoton:
- Ende einer Schleifeniteration
- Abarbeitung von Kommandos auf der obersten Ebene der kanonischen Operation (d. h. Kommandos, die nicht in tiefer geschachtelten Blöcken eingeordnet sind bzw. die in zyklenfreien Teilen der Kommandoabfolge liegen)

monoton:
- Existenz einer TERMINATE–Alternative, beim Warten bzw. Auswählen in einer SELECT–Anweisung
- Wartezustand auf die Rückkehr von Ordern oder am Ende von Blöcken
- Abschlußsynchronisation mit einzelnen M–Akteuren

Dabei gilt für die Betrachtung von FOR–Schleifen, daß als zuverlässiger Indikator des relativen Standes der Berechnung der Schleifenindex dienen kann, da er in INSEL nicht korrumpierbar ist. Am Ende von WHILE–Schleifeniterationen ist hingegen nur eine Bestimmung, ob eine weitere Iteration durchgeführt wird oder nicht, möglich. Darüberhinaus ist eine Abschätzung wie lange die nächste Iteration dauern wird (z. B. durch Mittelwertbildung bisheriger Dauern) und die Berücksichtigung von zusätzlichen Trendanalysen denkbar.

Daraus läßt sich dann je Verband/Akteur j auf seiner Stelle abschätzen wieviel Prozent seiner Berechnung bereits erledigt wurde. Dieser Schätzwert wird als $finished_j$ definiert. Nach dem b-Ereignis beträgt sein Wert 0 und mit dem Erreichen des e-Ereignisses ist sein Wert auf 1 angestiegen. Der Wert von $finished_j$ wird später u. a. beim Beispiel der Lastplazierungsstrategie verwendet.

2.4 Metriken und Kosten für das Ressourcenmanagement

Die Metriken und Indizes, die die Entscheidungsfindung des RM tragen, sind überwiegend erforscht. Umfangreiche Untersuchungen in der Literatur [6, 7, 8, 9, 10, 11] beschäftigen sich mit der Aussagekraft einfacher oder komponierter Kenngrößen wie Warteschlangenlängen , Grad der Speicherfragmentierung, bisherige Laufzeit und weitere Information beziehungsweise eben auch der Kombination zu mehrdimensionalen Lastindizes. Die Ergebnisse dieser Arbeiten finden ihren Niederschlag in den entwickelten Profilen, die in Abschnitt 2.5 vorgestellt werden.

Die Strategien verwenden dann die durch die Metriken quantifizierten und zusammengefaßten Werte und fällen ihre Entscheidungen. Üblicherweise findet die Beurteilung von einzelnen Strategien meist nur auf Grundlage von Vergleichsmessungen unter speziellen Randbedingungen statt. Die Aussagen bleiben daher auf einem eher qualitativen Niveau (z. B. die Strategie bevorzugt Auslastung gegenüber Beschleunigung oder Durchsatz), da meist die Kosten der Entscheidungsfindung nicht quantitativ erfaßt werden. Mit ein Grund dafür ist, daß in konventionellen d. h. insbesondere in nicht verteilten Systemumgebungen die Kosten des Managements gering sind gemessen an den Kosten der betreuten Anwendung. Diese Kosten sind bei verteiltem RM aber deutlich spürbar, da die Netzkommunikation für die Kooperation des RM und zur Aufrechterhaltung einer nutzbaren Systemsicht einen wesentlichen Faktor darstellt. Somit wird die Kosten–Nutzen–Rechnung für Strategien wesentlich und es werden Maße gebraucht, um den Managementaufwand und -erfolg zu quantifizieren, damit die Anpassung und Ersetzung von Strategien demgemäß geschehen kann.

2.4.1 Metriken

Um sinnvolle Beurteilungen für und über die Entscheidungen des Managements zu erhalten, ist es unerläßlich Vergleichsmaße (Metriken) einzuführen. Dabei wirkt sich der anvisierte Einsatzbereich im Management (RM oder MM) ebenso mit auf die Metrik aus wie die angestrebte Qualität der Aussage. Daher werden in diesem Abschnitt zuerst die Metriken mit ihrem Einsatzbereich und Einsatzziel aufgeführt. Anschließend werden weitere benötigte Aspekte betrachtet.

Die Metriken für das RM sollen hier nur kurz tabellarisch skizziert werden. Die Darstellung erhebt keinen Anspruch auf Vollständigkeit. Ziel aller dieser Maße ist die Beurteilung der Adäquatheit der Ressource bzw. bei Plazierungs- oder Migrationsentscheidungen geht es um die Eignung des Zielknotens.

Ressource	**Metrik bzw. Maßzahl**
Prozessor	Warteschlangenlänge rechenbereiter Aktivitäten
Speicher Cache	verfügbare freie und passend belegte nutzbare Seitenrahmen Anzahl der von dieser Aktivität belegten Cachelines
Kommunikation Geteilte Objekte / Dienste	Nachrichtenlaufzeit (Latenz- und Übertragungszeit) Mittlere Zugriffszeit, Rate der lokal ausführbaren Zugriffe,

Tabelle 2: Ressourcen und jeweils entsprechende Metriken bzw. Meßgrößen

Im Bereich des MM liegt die Ausrichtung anders. Hier ist das Ziel der Quantifizierung aufgeteilt in die Erfassung des Aufwands und in die Beurteilung der Güte der Entscheidung. Letzteres ist stark mit dem Aufwand verknüpft und vorwiegend mit den gewünschten Kriterien behaftet.

Der Aufwand des Managements setzt sich analog wie für jede Anwendung zusammen aus dem Verbrauch an CPU-Zeit, belegtem Speicher, Kommunikationsaufkommen und dem Zugriff auf geteilte Objekte. Die verwendeten Metriken sind daher primär die gleichen wie beim RM. Allerdings darf die Betrachtung sich dabei nicht nur auf eine einzelne Anforderung z. B. nach freien virtuellen Adressen beziehen, sondern muß größere Zeiträume ggf. die gesamte Lebenszeit einer Komponente einbeziehen. Über diesem Grundgerüst spannt sich dann die Frage nach einem Maß zur Beurteilung der Güte einer Strategie, wobei diese Güte pauschal schlecht zu fassen ist, da verschiedene, widerstreitende Kriterien gegeneinander abgewogen werden müssen. Beispielsweise steht die Minimierung ungenutzter Ressourcen (Verschnitt) im Gegensatz zur Minimierung von Maßnahmen des RMs. Gerade hier ist aber auch leicht sichtbar, daß eine Anfrage, die so ausgelegt ist, daß eine vorausschauende Bevorratung erfolgt, gegenüber einer genau bemessenen Anfrage gemäß dem aktuellen Bedarf zwar den Verschnitt zu senken vermag, jedoch durch häufigere, aufwendige Operationen mit der Umwelt teurer z. B. bzgl. der CPU-Zeit ist. Hier sind stets nur Kompromisse machbar. Mit Hilfe von anwendungsbezogener Information läßt sich aber unter Umständen erkennen, ob weitere Ressourcenanforderungen der gleichen Art in absehbarer Zukunft erfolgen werden, oder ob eventuell keine Bevorratung mehr gebraucht wird. So ist die Güte einer Strategie schwierig zu fassen. In diesem Bericht werden die entstandenen Kosten in Relation zum erreichten Berechnungsfortschritt gesetzt, wobei auch mit dieser Festlegung Probleme entstehen, wie unten noch näher ausgeführt wird.

Wenn man also davon ausgeht, daß ein Managementobjekt die Anfragen seines zu betreuenden Objekts erfolgreich bedient, und man dessen Berechnungsfortschritt quantifizieren kann, so kann man das Verhältnis zwischen dem Aufwand des Managements und dem Berechnungsfortschritt als Güte des Managements bzw. der Strategie definieren. Dabei ist zu berücksichtigen, daß der Aufwand eine mehrdimensionale Größe ist, die auf einen Skalar zu reduzieren ist, um die gewünschte Relation zum skalaren Berechnungsfortschritt zu ermitteln. Diese Reduktion kann durch Beschränkung auf eine Dimension oder durch Projektion bzw. (Linear-)Kombination erfolgen. Da der Aufwand eine zentrale Rolle spielt, widmen sich der folgende Abschnitt umfangreicher diesem Thema.

2.4.2 Kosten

Die Kosten der Berechnung und des Managements sind getrennt zu erfassen, da sie auch getrennt umgelegt werden sollen. Derzeit ist aber noch offen, wie dies geschehen soll, daher wird sich dieser Abschnitt mit der Aufzählung der Kostenverursacher begnügen, die ggf. noch zu erweitern ist.

- Verwaltung und Beschaffung der benötigten Ressourcen
  - entfernt genutzte Dienste und Daten,
  - Verwaltung von Warteräumen,
  - Speicherverwaltungsstrukturen
- Verpackung, Versand und Entpackung bei der Erzeugung oder Migration von Aktivitäten, sowie bei der Zustellung der Ergebnisse in der Abschlußsynchronisation
- Update/Invalidate von verteilt nutzbaren Objekten
- Kosten bzgl. der Adaption des zu verwaltenden Objekts:
  - Auswahl von Realisierungsalternativen gemäß der aktuellen Strategie,
  - Compilerlauf,
  - Aktivierung der Realisierung,
  - Beobachtung und Bewertung der gewählten Entscheidung
- Aufwand zur Erhebung und Verwaltung der Daten, die zur Entscheidung benötigt werden

- Kosten bzgl. der eigenen Adaption (Meta-Management):
  - Entscheidung zwischen Feinabstimmung (Anpassung von internen Parametern) und Strategiewahl,
  - Anpassung der internen Parameter oder Auswahl der Strategie (Realisierungsalternativen) mit ggf. Compilerlauf,
  - Aktivierung der Strategierealisierung,
  - Beobachtung und Bewertung der gewählten Strategie,
  - Weiterreichen der gesammelten Erfahrungen des RM und MM (z. B. Einflußnahme auf die Standardstrategie des Standardmanagers, der bei Objektinkarnierung mit erzeugt wird. Dies passiert u. U. im Management des entsprechenden Generators.)

Neben den Kosten der Berechnung ist eine Beschreibung und Erfassung des Berechnungsfortschritts wesentlich. Darauf wird im nächsten Abschnitt eingegangen.

2.5 Anforderungs- und Ressourcenprofile

Das hier vorgestellte Beispiel einer Lastplazierungs- oder -migrationsentscheidung ist ein erster intuitiver Vorschlag, den es später testweise zu evaluieren und ggf. zu verfeinern oder zu verwerfen gilt. Es soll damit die Brücke von der bislang sehr abstrakten Sichtweise auf konkrete Ausprägungen geschlagen werden.

Prinzipiell ist die primäre Fragestellung bei einer Ressourcenanforderung: Genügen die derzeit verfügbaren Ressourcen den Anforderungen überhaupt? Und erst in zweiter Linie stellt sich die Frage wo, wie und wann sich das „günstigste“ Ressourcenangebot bietet. Da aber meist von der hinreichenden Verfügbarkeit ggf. virtueller Ressourcen ausgegangen werden kann, löst sich die erste Frage durch die Beantwortung der zweiten. Das soll so verstanden werden, daß es in der Zuständigkeit der Strategie liegt, eine Anforderung ggf. temporär zurückzustellen, solange keine hinreichenden Ressourcen verfügbar sind. Daher beschäftigt sich dieser Abschnitt nurmehr mit der Fragestellung, wo sich die am besten passende Ressourcenumgebung bietet.

Den klassischen Fall stellt die Suche nach einem Plazierungs– oder Migrationsziel für ein Objekt dar. Diese Aufgabe soll mittels des charakteristischen Anforderungsprofils des Objektes und der aktuellen Ressourcenprofile aller Stellen geschehen. Zum besseren Verständnis werden hier zunächst die neuen Begriffe Anforderungs- und Ressourcenprofil definiert und dann erläutert, wie sie belegt und benutzt werden.

Die Objekte von INSEL–Systemen sind charakterisiert durch verschiedene Eigenschaften. So unterscheidet man Rechen-, Speicher- und Kommunikationsfähigkeiten. Die dafür geforderten Ressourcen ergeben dann das *Anforderungsprofil* eines Objektes. Dieses Profil wird mittels eines Vektors dargestellt und enthält die Gewichtung der Anforderungen an die verschiedenen Ressourcenarten relativ zu einander, ohne den Bedarf (z. B. Speichermenge) zu quantifizieren. Zusätzlich zu den Ressourcen erscheint insbesondere hinsichtlich der Plazierungsfrage eine Bevorzugungsmöglichkeiten für bestimmte Stellen des verteilten Systems für angebracht. Das *Ressourcenprofil* gibt die quantitative Verfügbarkeit der einzelnen Ressourcen einer Stelle bezogen auf die Anfrage eines Objektes wieder. Die Verknüpfung der Gewichtung mit dem jeweiligen Angebot ergibt dann die Beurteilung der Stelle hinsichtlich der Anforderungen des Objektes.

Die systematische Belegung der Koeffizienten des Anforderungsprofils ist noch eine offene Frage. Klar ist allerdings, daß sich aus Codeanalysen und akkumulierten Erfahrungswerten des entsprechenden Generators die Initialbelegung vor der Objekterzeugung ableitet. Zur Lebenszeit des Objektes können die Koeffizienten dynamisch angepaßt werden, um die Erfordernisse in speziellen Phasen besser wiederzugeben. Die auftretenden Dimensionen des Anforderungsprofils sind:

- Berechnung be,
- Kommunikation ko,
- Speichernutzung sn und
- Stellenbindung sb.

Der Berechnungskoeffizient be bzw. Kommunikationskoeffizient ko gibt an wie ausgeprägt der Rechen- bzw. Kommunikationsanteil des Objektes ist. Je größer der Speichernutzungskoeffizient sn ist, desto wichtiger ist es für das Objekt, daß seine Anforderungen nach (gleichzeitiger) Präsenz möglichst aller belegten Speicherseiten (zur Benutzung seiner virtuellen Adressen) erfüllt werden. Schließlich beschreibt der Koeffizient der Stellenbindung sb, welchen Stellenwert die Plazierung auf dem Knoten des Erzeugers einnimmt. So wird ein Akteur mit einer erwarteten kurzen Lebensdauer (z. B. bei Realisierung als Funktionsaufruf), sicherlich einen hohen Stellenbindungskoeffizient haben gegenüber einem Akteur, der als langlebig eingeschätzt wird und z. B. als Thread angelegt wird. Diese Koeffizienten sind nicht als absolute Größen zu betrachten, sondern sollen nur die Relationen zueinander beschreiben. Der Wertebereich ist das Intervall $[0; 1]$, wobei die Koeffizienten so normiert werden, daß ihre Summe 1 ergibt.

Das als Vektor beschriebene Anforderungsprofil $a = (be, ko, sn, sb)$ wird mit der Matrix R multipliziert, die die jeweiligen, geschätzten Verfügbarkeiten der Ressource auf den Stellen des verteilten Systems enthält. Die Zeile i der Matrix enthält damit das sogenannte Ressourcenprofil der Stelle i. Der Ergebnisvektor enthält dann pro Zeile die aktuelle Bewertung der Stelle bezogen auf das geforderte Profil. Die Schätzung der Verfügbarkeit bezieht sich dabei nur auf die aktuelle gegebene Situation gegenüber der aktuellen angeforderten Ressourcenmenge. Es wird z. B. die Migrationsfähigkeit von gemeinsam genutzten Objekten nicht berücksichtigt, sondern davon ausgegangen, daß diese bei hinreichend intensiver Nutzung dem Bedarf Rechnung tragen und eine entsprechende Realisierung wählen. Die Elemente der Matrix R (im weiteren auch Faktoren genannt) liegen alle im Intervall $[0; 1]$ und berechnen sich wie folgt:

- Berechnung: $BEF_i = 1/(cpuload_i + 1)$

  wobei $cpuload_i$ das Lastmaß der Stelle i ist. Es kann für eine einfache Näherung mit thr_i, der Anzahl der auf Stelle i plazierten Akteure gleichgesetzt werden.

Ist hingegen auch eine Zukunftsbetrachtung erwünscht, so wird für jeden Akteur abschätzt wie lange er noch laufen wird und extrapoliert wieviel Rechenzeit er, basierend auf seinem derzeitigen Verbrauch, noch konsumieren wird. Das ist mit der folgenden Formel ausgedrückt:

$$cpuload_i = \sum_{j=1}^{thr_i}((1 - finished_j) \cdot work_j / finished_j)$$

$work_j$ ist die bereits von Akteur j verbrauchte CPU-Zeit.

Allerdings birgt diese Formel Probleme, da keine noch nicht gestarteten Akteure (mit $finished_j = work_j = 0$) und auch keine „ewig laufenden" Dienstanbieter korrekt erfaßt werden. Diese Kandidaten entziehen sich der Abschätzung bzw. können als Grundlastanteil betrachtet werden. Grundsätzliche Fragestellung ist jedoch, ob der derzeitige Lastzustand oder das geschätzte, noch abzuarbeitende Lastvolumen betrachtet werden soll.

- Kommunikation: $KOF_i = lk_i / gk$

  wobei lk_i die Anzahl der lokal auf Stelle i vorliegender Kommunikationspartner ist und gk die Gesamtanzahl der Kommunikationspartner ist (dabei zählt der Erzeuger mit dem die Abschlußsynchronisation betrieben wird als Kommunikationspartner mit).

- Speichernutzung:

$$SNF_i = \begin{cases} 1/2 + 1/2 \cdot \frac{h_i}{h_i + s_i} & : \quad f_i \geq s_i \\ 1/2 \cdot \frac{h_i + 1}{h_i + s_i + 1} & : \quad s_i > f_i \wedge g_i \geq h_i + s_i \\ 0 & : \quad h_i + s_i \geq g_i \end{cases}$$

  wobei f_i die Anzahl freier Seitenrahmen auf Stelle i, g_i die Gesamtanzahl der Seitenrahmen auf Stelle i, h_i die Anzahl der vom Objekt benötigten und schon passend belegten Seitenrahmen (Haben) auf Stelle i und s_i die Anzahl der vom Objekt benötigten und noch passend zu belegenden Seitenrahmen (Soll) auf Stelle i ist.

  In der zweiten Zeile bei der Bestimmung von SNF_i dient die Addition um 1 im Zähler und Nenner dazu, die Wertebereiche der einzelnen Fälle überlappungsfrei zu trennen, falls keine der gewünschten Seiten eingelagert ist ($h_i = 0$). An einem kleinen Zahlenbeispiel kann man sich das schnell verdeutlichen. Ebenso wird daran auch klar, daß eine Stelle, die die Soll–Anforderung erfüllen kann ggf. auch mit einigem Aufwand, immer höher bewertet wird als eine Stelle, die schon fast alle Seiten parat hat, aber die fehlenden Seiten derzeit nicht gleichzeitig einblenden kann.

- Stellenbindung:
  Hier sind verschiedene Formeln denkbar. Die erste Variante strebt eine Plazierung der abgespaltenen Akteure möglichst lokal an. Die zweite Variante versucht

Akteure, die kurz vor ihrem Lebenszeitende stehen nicht mehr zu verschieben.

$$SBF_i = \begin{cases} 1 & : \text{ Der Erzeuger ist auf Stelle } i \text{ lokalisiert.} \\ 0 & : \text{ sonst} \end{cases}$$

$$SBF_i = \begin{cases} finished_j & : \text{ Der Akteur } j \text{ befindet sich bereits auf Stelle } i. \\ 0 & : \text{ sonst} \end{cases}$$

Da noch kein Verfahren zur Wahl oder Anpassung der Koeffizienten naheliegt, zeigt sich, daß eine Evaluierung durch ein lauffähiges System unbedingt nötig ist, um Erfahrungswerte für die Wahl der Werte des Anforderungsprofils zu gewinnen.

Damit ergibt sich für ein Objekt mit dem Anforderungsprofil $a = (be, ko, sn, sb)$ und der Matrix R der Ressourcenprofile aller n Stellen der Bewertungsvektor $b = (b_0, \ldots, b_{n-1})$ aus der Matrixmultiplikation $R \cdot a = b$. Durch die Normierung und die Beschränkung des Wertebereichs der Matrixelemente liegen alle b_i ebenfalls im Intervall von $[0; 1]$. Je höher der Wert von b_i desto besser erfüllt die Stelle i die gewünschten Anforderungen. Damit lassen sich Vergleiche zwischen verschiedenen Bewertungen (auch zwischen verschiedenen Ressourcenkonsumenten) anstellen.

Prinzipiell kann durch Modifikation der Gewichtung der betrachtete Informationsraum selektiv gestaltet werden und vom vorgeschlagenen mehrdimensionalen Modell bis zur Verwendung eines eindimensionalen Indexes reduziert werden.

3 Meta-Management

Dieser Abschnitt widmet sich nun der Fragestellung der Adaption der Strategien und konzentriert sich damit auf Themen des MM. Dieser Teil der Forschungsarbeit ist noch am wenigsten ausgestaltet und birgt daher viele offene Fragen. Für die Adaption werden zuerst die Strategien klassifiziert, um eine Grundlage der Vorauswahl zu gewinnen. Die Klassifikation wird sich dabei nicht an einzelnen vorhandenen Eigenschaften orientieren, sondern am Potential, das eine Strategie mitbringt. So ist beispielsweise nicht der Umfang der verarbeiteten Historie ein klassifizierendes Merkmal, sondern ist die entscheidende Eigenschaft, ob zurückliegende Information verarbeitet wird. Im Anschluß daran wird auf die Adaption als solches eingegangen und ein kurzer Abschnitt skizziert die Aspekte des Aufwandes für das MM.

3.1 Dimensionen des Strategieraumes

Die einzelnen Strategien unterscheiden sich im wesentlichen gemäß der folgenden Charakteristika, die zur Entscheidungsfindung betrachtet bzw. verwaltet werden: *Zeitraum*, *Informationsquellen* und *Ressourcen.*

Anhand dieser drei Achsen spannt sich ein Raum auf, der eine Klassifikation der Strategien erlaubt. Der Ordinatenwert längs einer Achse beschreibt, welcher Blickwinkel eingenommen wird. Dabei gilt längs der Achse eine Inklusion aller dem Ursprung näherliegenden Eigenschaften. Beispielsweise gilt bei der Achse des Zeitraumes, daß die Betrachtung der Historie natürlich auch die Betrachtung des aktuellen Zustands beinhaltet. Ebenso wird bei Berücksichtigung anwendungsbezogener Information auch systembezogene Meßdaten und allgemeine Annahmen (z. B. Gleichverteilung modelliert die Auftretenswahrscheinlichkeit hinreichend) mit in die Entscheidungsfindung einfließen.

- **Zeitraum:**
  - Aktueller Zustand
  - Historie (auch nur ausschnittsweise, gefiltert oder vollständig)
  - Zukunft (Extrapolationen und Vorhersagen)
- **Informationsraum:**
  - Annahmen (Nur Postulate, keine Meßdaten, kein abgeleitetes oder durch Analysen gewonnenes Wissen)
  - Systembezogene Meßdaten (gemessene Daten des Systems, wobei die Grenzen fließend sind, da z. B. die Anzahl der erzeugten und abgespaltenen Aktivitäten einer Anwendung sich ja u. U. direkt in der Warteschlangenlänge widerspiegeln. Kriterium könnte sein: Die Daten betreffen die Stelle bzw. sind stellenlokal erhoben.)
  - Anwendungsbezogene Information (Anwendungsspezifische Meß- oder Analyseergebnisse: z. B. dieses Objekt hat soviele Kommunikationszugriffe ausgelöst, ist mit X durch die y-Relation gekoppelt ...)
- **Ressourcenraum:**
  - Einzelressource
  - Einzelressource mit Berücksichtung von Wechselwirkungen auf andere Ressourcen
  - Konzertierte Ressourcen

Die Abbildung 4 veranschaulicht den aufgespannten Strategieraum und ordnet Strategien aus unserem Experimentalsystem *AdaM* und dem *Distributed Thread Kernel* (DTK), sowie die bekannten Strategien Round Robin und LRU ein.

Eine zusammenfassende Beschreibung der in den Experimentalsystemen eingesetzten Strategien findet sich in [12]. Umfangreicher werden sie in den jeweiligen Dissertationen [13, 14] dargestellt.

In der Tabelle 3 sind die Eigenschaften stichwortartig beschrieben, die dann zu der dargestellten Positionierung im Strategieraum geführt haben. Am Beispiel des Receiver-/Sender-Biddings, wie es im DTK implementiert wurde, wird hier die Klassifizierung

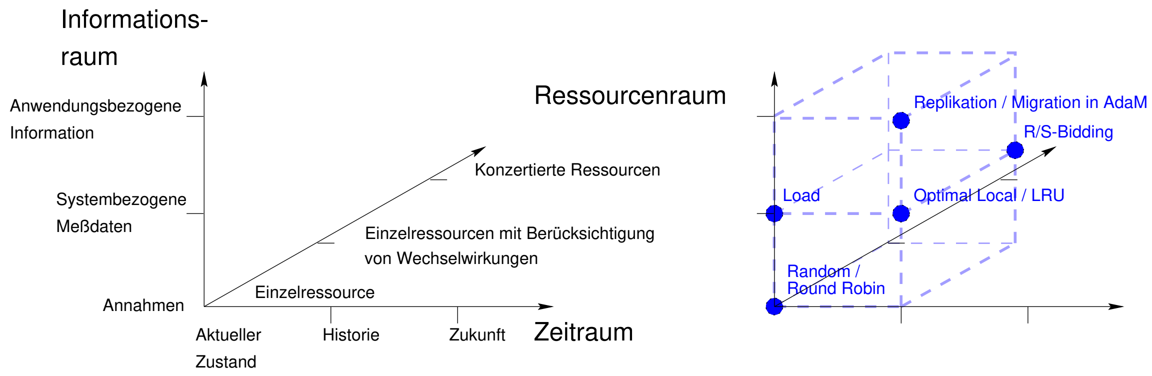

Abbildung 4: Der Strategieraum und darin eingeordnete Strategien

Strategiename	**Zeitraum**	**Informationsraum**	**Ressourcenraum**
Replikation, Migration (beide in AdaM)	Zugriffshistorie	Analyse des Zugriffsverhaltens der Anwendung	Ausprägung der Objekt–Realisierung
Random (DTK)	aktueller Zustand	Annahmen	Prozessor (keine Wechselwirkungen)
Optimal Local (DTK)	Historie eigener Lasterzeugung	entfernt erzeugte, eigene Lastpakete	Prozessor (keine Wechselwirkungen)
Load (DTK)	aktueller Zustand	Warteschlangen	Prozessor (keine Wechselwirkungen)
Receiver-/Sender-Bidding (DTK)	Berücksichtigung früherer Angebote	Warteschlangen und Nachrichten	Prozessor mit Betrachtung der Kommunikation
LRU	Zugriffshistorie	Häufigkeit von Seitenzugriffen	Speicherseiten (keine Wechselwirkungen)
Round Robin	aktueller Zustand	Annahmen	Prozessor (keine Wechselwirkungen)

Tabelle 3: Charakterisierung der Strategien gemäß den Dimensionen des Strategieraumes

der Strategie erläutert. Die Klassifizierung der anderen Strategien und die Bedeutung der einzelnen Tabelleneinträge erschließt sich daraus analog ggf. unter Zuhilfenahme der obigen Literatur.

Beide Bidding-Strategien verwalten eine Warteschlange noch zu erzeugender Threads und ermitteln den Grad der lokalen Auslastung anhand der lokalen Warteschlangenlänge rechenbereiter Threads. Wenn ein neuer Thread erzeugt werden soll und die lokale Auslastung unterhalb gewisser Schranken liegt, so geschieht dies sofort lokal. Bei hoher Auslastung wird der Thread in die Warteschlange der noch zu erzeugenden Threads eingereiht. Im Fall der Sender-initiierten Strategie wird dann eine Anfrage nach freier

Kapazität an alle beteiligten Rechnerknoten verschickt. Bei Eingang eines Angebots werden Erzeugungsaufträge aus der Warteschlange noch zu erzeugender Threads vergeben.

Bei der Receiver-initiierten Strategie liegt der Fall anders. Wenn hier die Auslastung sehr niedrig wird, so versendet sie Kapazitätsangebote an eine gleitende Auswahl von Rechnerknoten. Nicht beantwortete Angebote werden nach einer gewissen Zeit zurückgezogen und an weitere Knoten gestellt. Ferner werden nur soviele Angebote ausgegeben, daß bei durchweg positiver Beantwortung der anbietende Knoten gerade ausgelastet ist. Dadurch wird eine Überflutung vermieden. Empfangene Angebote werden temporär gespeichert, so daß bei Verschiebung der lokalen Lastsituation auf diese Nachrichten zurückgegriffen werden kann.

Durch die Verzögerung der Thread-Erzeugung können insbesondere dann Nachrichten eingespart werden, wenn eine lokale Erzeugung aufgrund einer sich zwischenzeitlich entspannenden Lastsituation angebracht ist. Die verzögerte Thread-Erzeugung verringert die Wahrscheinlichkeit der völligen Überlastung des Knotens, da die Warteschlangenlänge das Erzeugungskriterium darstellt. Auf diese Art werden die Ressourcen Prozessor und Kommunikation (bzw. genauer die Nachfrage nach ihnen) in einer sich gegenseitig lose beeinflussenden Beziehung aneinander gekoppelt. Da die Lastverwaltung sich in der Schicht des DTK abspielt und keinerlei Information verarbeitet, welcher Thread zu welcher Anwendung gehört, wird nur systembezogene, anwendungsunspezifische Information genutzt. Damit definiert sich die Position des Receiver-/Sender-Bidding im Strategieraum.

Aus der Klassifikation der Strategie kann ihr qualitatives Potential abgelesen werden, d. h. ob ein gewünschtes Leistungsmerkmal potentiell erreichbar ist oder nicht. So kann z. B. der Wunsch nach Lernfähigkeit nur dann erfüllt werden, wenn mindestens die Vergangenheit in Betracht gezogen wird. Es gilt zudem abzuschätzen, welche Kosten mit den einzelnen Strategien entstehen und dann solche zu wählen, die ein entsprechendes qualitatives Potential haben und dies dann möglichst kostengünstig auszuschöpfen. Diese Aufgabe ist nicht trivial, da der Nutzen gegenüber den Kosten von aufwendigeren Strategien sich schlecht abschätzen läßt. Es bieten sich hier letztendlich nur zwei Wege: Erstens die Entwicklung einer alles umfassenden Modellierung und Methodik, so daß der zu erforschende und zur Entfaltung zur Verfügung stehende Raum sich genauer gliedert und so die Orientierung ermöglicht, welcher Weg zur Ausprägung des Systems sinnvoll ist. Dies ist aber angesichts des Volumens eine gigantische Aufgabe. Zweitens bietet sich der evolutionäre Weg, bei dem eine durch Heuristiken und Empirie gestützte Lösung entworfen und fortan dynamisch, ständig gemäß den Bedürfnissen und Erfahrungen angepaßt wird.

3.2 Adaption

Die Adaption des Managements erfolgt gemäß zweier Wege: flexible Strategiewahl und Parameteranpassung. Als Auslöser dieser Meta-Managementmaßnahmen dienen ebenfalls die konkreten Ereignisse, wobei klar ist, daß diese Aktivitäten wesentlich seltener stattfinden, als die Aktivitäten des RM. Eine kanonische Koppelung von speziellen Ereignissen an Maßnahmen des MM liegt bestenfalls bei der Erzeugung bzw. Auflösung von Managementkomponenten vor. Da aber gerade langlaufende Managementkomponenten ihre Strategien und Mechanismen adaptieren sollen, wäre zusätzlich eine zyklische Bewertung denkbar, d. h. nachdem die Strategie eine gewisse Anzahl von Entscheidungen gefällt hat, wird ihre Güte (die verursachten Kosten des Managements in Relation zum Berechnungsfortschritt des betreuten Objekts) beurteilt. Die Schwierigkeit der Gütebeurteilung zeigt sich, wenn zusätzlich betrachtet wird wie groß der Verschnitt der Ressourcen ist oder wenn eine Entscheidung erst sehr viel später Gewinn abwirft (z.B. Compilerlauf) und die damit verbundenen Kosten hoch und der Berechnungsfortschritt gering ist.

Ein methodisches Vorgehen, welche Strategie auf welches Objekt angesetzt werden soll, ist noch nicht näher untersucht. Bevor hier aber verschiedene Ansätze zur Diskussion gestellt werden, seien hier noch die hilfreichen Begriffe *Anwendung, Aufspaltungsstruktur* und *kritische Menge* definiert.

Unter einer **Anwendung** soll ein Subsystem des Gesamtsystems verstanden werden, das eine bezüglich seiner Wechselwirkung mit dem restlichen System abgrenzbare und aktive Einheit wie z. B. eine Akteursphäre darstellt. Es sei darauf hingewiesen, daß damit auch das Gesamtsystem als eine Anwendung betrachtet werden kann.

Die **Aufspaltungsstruktur** einer Anwendung beschreibt, die Art der Aufteilung in nebenläufige Aktivitäten. Es werden hier nur die Formen der *fächerartigen* und *baumartigen* Abspaltung betrachtet. Eine **fächerartige Aufspaltung** liegt vor, wenn ein Akteur sämtliche nebenläufigen Aktivitäten direkt abspaltet, also keine hierarchische Stufung innerhalb seiner unter ihm erzeugten Aktivitäten vorliegt. Eine **baumartige Aufspaltung** liegt vor, wenn ein Akteur ein paar nebenläufige Aktivitäten direkt abspaltet und diese wiederum weitere abspalten, so daß sich eine mehrstufige Hierarchie entwickelt.

Aus dem Bereich der Projektplanung stammt der Begriff des **kritischen Pfades**, der alle die Teilschritte beinhaltet, die die Gesamtdauer eines Projektes bestimmen. Das heißt, daß eine Verzögerung eines dieser Teilschritte eine Verlängerung der Gesamtdauer des Projektes bewirkt. Diese Teilschritte werden daher kritisch genannt. Die Definition der kritischen Menge einer Anwendung lehnt sich an die Definition des kritischen Pfades an. Die **kritische Menge** besteht aus den Aktivitätsträgern, von denen der Berechnungsfortschritt der Anwendung abhängt. Es sind also die Aktivitäten, die derzeit abgearbeitet werden können und die bei Aufschiebung ihrer Ressourcenanfrage eine Gesamtverzögerung der Anwendung bewirken.

Die Zielsetzung aller Scheduling-Verfahren kann damit beschrieben werden, daß mit möglichst geringen Kosten und unter fairer Berücksichtigung der verschiedenen Benutzer eines Systems, jeweils die Aktivitäten Prozessorzeit zugeteilt bekommen sollen, die in der kritischen Menge liegen. Die Verwendung der kritischen Menge beschränkt sich aber nicht nur auf die Prozessorzuteilung, sondern ist für alle Ressourcen ausgelegt. Im weiteren wird die hohe Abstraktionsebene schrittweise verlassen und ein Ansatz beschrieben, der am Beispiel des Scheduling die Probleme des verteilten MM verdeutlichen soll.

Bei einem ersten Ansatz zur Wahl des Scheduling-Verfahrens, der vorallem die Systemstrukturen und das Konzept der Schachtelung nutzt, wird für jede Anwendung entschieden, wie die virtuelle Prozessorzeit für eigene Berechnungen und die der untergeordneten Aktivitäten genutzt wird. Dabei entsteht ein Scheduling-Verfahren, das durch Scheduling-Entscheidungen auf allen Ebenen der Schachtelungshierarchie geprägt ist. So kann auf jeder Ebene den speziellen Anforderungen der Aktivitätsträger entsprochen werden. Dieser Weg ist aber nur auf abstrakter Ebene sinnvoll, da bei naiver Umsetzung der Aufwand unangemessen hoch wird. Die wesentliche Grundidee ist jedoch die Information der gewünschten Eigenschaften bis zur Scheduling-Entscheidung zu transportieren und sie ins das Verfahren einfließen zu lassen. Dies ist jedoch auch anders möglich. Wenn jede Aktivität die von ihr gewünschten Eigenschaften (z. B. Reihenfolgentreue mit anderen pipeline-artig kooperierenden Aktivitäten, ...) kennt, dann kann ein stellengebundenes Scheduling-Management ein Scheduling-Verfahren für die Menge der Aktivitäten mit kollisionsfreien Anforderungen festlegen. Zwischen den Gruppen mit widerstreitenden Anforderungen muß dann ein faires Auswahlverfahren gelten. Da dies hier nur als zu diskutierende Idee dargestellt wird, werden Details wie z. B. die Ermittlung und Speicherung der gewünschten Eigenschaften ausgelassen.

Im folgenden Text ist eine durch [15] inspirierte Gliederung und Zusammenstellung von Eigenschaften skizziert, auf welcher Grundlage die Entscheidung für Scheduling-Verfahren fallen könnte.

- **Fächerartige Aufspaltung und große kritische Menge:** Für eine Anwendung mit einer fächerartigen Aufspaltungsstruktur und bei der fast alle Aktivitäten in der kritischen Menge liegen, ist ein Round-Robin- oder sogar Run-to-completion-artiges Verfahren vorzuziehen. Wenig Verwaltungsaufwand und eine in Form von Gleichbehandlung aller verwalteten Kandidaten gesicherte Fairneß sind ihre charakteristischen Eigenschaften, die auch von der obigen Anwendung gefordert werden.

- **Fächerartige Aufspaltung und kleine kritische Menge:** Eine Anwendung mit fächerartiger Aufspaltungsstruktur führt allerdings nicht zwingend zu Round-Robin-Verfahren. Man denke beispielsweise an die Abspaltung eines stark frequentierten Server-Objektes und einer großen Anzahl von Client-Objekten. Es ist einsichtig, daß die Größe der kritischen Menge stark schwanken kann und trotzdem der Server gleichbleibend bevorzugt Rechenleistung zugeteilt bekommen sollte. Hier erscheinen Verfahren, die Prioritaten kennen, angebracht.

- **Baumartige Aufspaltung:** Bei einer Anwendung mit einer baumartigen Aufspaltungsstruktur liegen die Blätter zumeist in der kritischen Menge. Je nach Verhältnis der Anzahlen von Blättern zu inneren Knoten ist dann ein entsprechendes Scheduling-Verfahren sinnvoll. An diesem Punkt soll aber nicht auf verschiedene Varianten mit dynamischer Prioritätsanpassung eingegangen werden.

Adaption ist auch in diesem Beispiel zwiefältig denkbar. So könnte in einem Parameteranpassungsschritt, die Zeitscheibengröße so gewählt werden, daß sie zur vollständigen Erledigung einer abgespaltenen Aktivität ausreicht und so die Zahl der Kontextwechsel auf ein Minimum reduziert wird. Dadurch kann Round-Robin faktisch dem Run-to-completion gleichgesetzt werden, wobei andere Anwendungen nach wie vor ihre gesicherte Reihenfolgentreue und eine zyklische Abarbeitung beibehalten. Flexibilisierung könnte in dem Moment eingreifen, wenn die kritische Menge wenige Elemente enthält, d. h. nur noch wenige konkurrierende Aktivitäten rechnen und die übergeordnete Einheit ebenfalls zur kritischen Menge zählt. Unter diesen Umständen kann der Wechsel zu einem Scheduling-Verfahren mit Prioritäten vorteilhaft sein.

Prinzipiell ist natürlich ein Scheduling-Verfahren zu wählen, daß die Aktivitäten der kritischen Menge vor allen anderen bevorzugt und dabei möglichst wenig Kosten verursacht. Zur Bestimmung der kritischen Menge kann bedingt das Systembeschreibungsmodell verwendet werden.

Die Fragestellung, welche Berechnung nun voranschreiten soll, ist aber nicht auf Prozessorzeitzuteilung d. h. Scheduling beschränkt, sondern gilt im gleichen Maße für alle anderen Ressourcen.

Die von der Anwendung gewünschten Eigenschaften der Ressourcenzuteilung sollten erhalten bleiben, wenn die Abbildung auf die Stellen bzw. auf ein verteiltes System erfolgt. Diese Eigenschaften zu isolieren und zu erkennen ist noch offen, jedoch bietet die anfangs eingeführte Klassifikation schon einen Satz an wesentlichen Merkmalen. Die als Beispiel genannten Eigenschaften sind sicher noch viel zu komplex und spezifisch, zeigen aber wo das Ziel letztendlich liegt. Eine offene Frage ist auch, wie verteilte Aspekte z. B. gleichzeitiges Ausführen, repräsentiert und umgesetzt werden können. Zudem kommt durch die Verteilung auch die Frage auf, wie und ob die charakteristischen Eigenschaften erhalten werden bzw. was z. B. unter einer stellenübergreifenden Fairneß zwischen den Aktivitäten einer Anwendung zu verstehen ist. Dieser Punkt ist sicher einer der interessanten offenen Fragen.

3.3 Managementaufwand

Die Aufwandsbeobachtung des RM liegt in der Verantwortlichkeit des MM und läßt sich wiederum für die einzelnen benötigten Ressourcen betrachten. So wird z. B. Speicher zur Aufzeichnung der Historie gebraucht, Rechenleistung zur Bildung von Mittelwerten, Maxima- und Minimabestimmung, Adreßrechnung, Zugriff auf entfernte Informationsanbieter. Diese Dinge erfolgen aber analog zur Betrachtung des Systemfortschritts bzw. des Ressourcenverbrauchs der betreuten Objekte, daher bietet dieser Abschnitt nicht wesentlich Neues. Die Aufwandsbestimmung des MM soll aber nicht weiter betrachtet werden, um endlose, rekursive Selbstbetrachtungen zu vermeiden.

Eine objektive Einschätzung der Effizienz des Managements ist nur mit einer festen Bezugsgröße machbar. Diese Größe der minimalen Berechnungsdauer ist aber faktisch nicht greifbar, sondern kann nur als untere Schranke abgeschätzt werden, wobei davon ausgegangen werden müßte, daß das Management nichts kostet und stets hinreichend viele Ressourcen zur Verfügung stehen. Auch ist eine exakte Bestimmung des Aufwandes gemessen am Berechnungsfortschritt erst mit der Terminierung des betrachteten Objektes möglich und damit nur eine retrospektive Gütenbeurteilung machbar.

Im Potential einer prognostizierenden Strategie liegt es dennoch, die Prognosen zum Beginn ihrer Tätigkeitszeit aufzuzeichnen und dann später in einer Retrospektive mit dem wirklichen Geschehen zu vergleichen und — da hier das MM betrachtet werden soll — die Qualität der Prognose für zukünftige Aufgabenstellungen zu verbessern. Diese Eigenschaft kann man durchaus mit denen von lernenden Systemen vergleichen, der sich insbesondere durch den Aspekt der Selbstreflektion auszeichnet. Ein solcher Ansatz liegt durchaus im Potential des evolutionären MMs und stellt eine konsequente Fortsetzung des Grundgedankens dar, leider hat er jedoch hinsichtlich des aktuellen Stands der Arbeiten eher visionären Charakter.

4 Zusammenfassung und weitere Arbeitsschritte

Durch die Komplexität des angestrebten Systems und die zahlreichen Einflüsse erscheint eine Evaluierung der vorgestellten Ideen in einem realen System als unabdingbar. Die weiteren Arbeitsschritte sind daher vornehmlich auf die Erstellung eines lauffähigen Basissystems ausgerichtet. Derzeit wird das Shadow–System [16] dahingehend erweitert, daß Scheduling-Verfahren gewählt und angepaßt werden können. Später sollen dann die erwähnten Profile von den Stellen erzeugt und von den beigestellten Managern genutzt werden. Dazu sind neben einer Erweiterung der Plazierungsentscheidung die analoge Entwicklung von Migration für die Aktivitäten erforderlich. Zudem sind weitere Analysen z. B. des Kooperationsverhaltens über geteilte Objekte und den Berechnungsfortschritt nötig.

5 Offene Fragen

Zahlreiche Fragen sind derzeit noch offen. Vorallem auf der Seite der Methodik der Adaptionsverfahren, der Verbindlichkeit der Entscheidungen auf RM- und MM-Ebene und hinsichtlich der quantitativen Erfassung des Berechnungsfortschritts ist noch viel zu tun.

Ausgehend von der Basis der IT-Kooperation ist die Frage nach der Abgrenzung der Wirkungsbereiche noch recht einfach. Wenn jedoch weitere Kooperationsformen, wie sie z. B. K–Akteure bieten, in die Systemmodellierung aufgenommen werden, muß man sich über die Wirkungsbereiche und die Darstellung der Abhängigkeiten Gedanken machen.

Die Anforderungen der Aktivitäten, die über das Volumen der benötigten Ressourcen hinaus gehen, werden bislang überhaupt nicht erfaßt. So erfolgt keine Abwägung zwischen paralleler gegenüber verschränkter Ausführung, was gemäß [17] insbesondere für kommunizierende und kooperierende Gruppen von Wichtigkeit ist.

Komplexere Abhängigkeiten zwischen den Gewichtungen der Ressourcen lassen sich ebenfalls so noch nicht erfassen. Mit der Verwendung der vorgestellten Profile lassen sich Abhängigkeiten und Koppelungen von Ressourcen nur tendenziell umsetzen, so kann man nicht ausdrücken, daß es einem Objekt auf die Gleichzeitigkeit der Verfügbarkeit von Speicher und Rechenzeit ankommt, um beispielsweise eine Echtzeitanforderung zu erfüllen. Hier ist ein umfassenderes Beschreibungsinstrumentarium von nöten, das den vielfältigen Anforderungen der Objekte entsprechen kann.

Ebenso ist einiges an Arbeit hinsichtlich der Interpretation von Strategien in verteilten Systemen zu leisten. Damit sind auch erweiterte Strategien nötig, die vorallem die Aspekte der Bewahrung und Vergleichbarkeit ihrer Eigenschaften erhalten. Ein weiteres Betätigungsfeld wäre die Untersuchung inwiefern Begriffe, die im Bereich des sequentiellen Rechnens eindeutig und klar sind, durch die Parallelisierung keinen oder einen anderen Sinn erhalten. So ist z. B. Fairneß in einem sequentiellem System durch zyklisches Zuweisen von Ressourcen an Konsumenten einfach machbar. In verteilten Systemen kann dies aber zu sehr unerwünschten Effekten führen, wenn die Konsumenten erst zum Ressourcenanbieter gelangen müssen. Dies mag zwar als fair betrachtet werden, nur ist das eigentliche Ziel, die Vermeidung von unerträglich langen Wartezeiten eines Konsumenten auf die Ressourcen damit verfehlt, sofern innerhalb der Zeit der Migration eine entsprechende Ressource lokal verfügbar wird.

Ein anderer Punkt ist, daß die Konzentration auf verteilte Aspekte wie z. B. Lastbalanzierung nur dann sinnvoll ist, wenn dabei die aus dem sequentiellen Bereich als wichtig erkannten Aspekte, z. B. das Scheduling von Elementen der kritischen Menge, erhalten werden. So ist es unsinnig ein perfekt balanziertes verteiltes System zu haben, wenn alle Elemente der kritischen Menge sich auf einem oder auch nur auf wenigen Rechnerknoten zusammenballen.

Literatur

[1] *Bode A.* Werkzeug und Methoden für die Nutzung paralleler Rechnerarchitekturen Bericht für 1996 bis 1997. Bericht, Technische Universität München (1997).

[2] *Bode A.* Werkzeug und Methoden für die Nutzung paralleler Rechnerarchitekturen Antrag für 1998 bis 2000. Bericht, Technische Universität München (1997).

[3] *P. P. Spies, C. Eckert, M. Lange, D. Marek, R. Radermacher, F. Weimer und H.-M. Windisch.* Sprachkonzepte zur Konstruktion verteilter Systeme. Bericht, TU München (März 1996). SFB–Bericht 342/09/96 A TUM–I9618.

[4] *Markus Pizka.* A Language-Based Approach to Construct Structured and Efficient Object-Based Distributed Systems. In „Proceedings of 30th Hawaii International Conference on System Sciences". IEEE (1 1997).

[5] *R. Radermacher und F. Weimer.* INSEL Syntax–Bericht. Bericht, TU München (März 1996). SFB–Bericht 342/08/96 A TUM–I9617.

[6] *Eduard G. Jr Coffman und Peter J. Denning.* „Operation Systems Theory". Automatic Computation. Prentice Hall (1973). ISBN 0-13-637868-4.

[7] *Luis-Felipe Cabrera.* The Influence of Workload on Load Balancing Strategies. In „USENIX Conference".

[8] *Theodore J. Johnson Anna Hac.* Sensitivity Study of the Load Balancing Algorithm in a Distributed System. *Parallel and Distributed Computing* (1990).

[9] *R. D. Blumofe, C. Joerg, B. C. Kuszmaul, C. E. Leiserson, K. H. Randall und Y. Zhou.* Cilk: An Efficient Multithreaded Runtime System. In „5th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming", Seiten 207–216, Santa Barbara, California (July 19-21 1995).

[10] *F.C.H. Lin und R.M. Keller.* The Gradient Model Load Balancing Method. *IEEE Transactions on Software Engineering* **SE-13**(1) (Januar 1987).

[11] *D. Ferrari und S. Zhou.* A Load Index for Dynamic Load Balancing. In „Proceedings of the Sixth International Conference on Distributed Computing Systems", Seiten 684–690, Cambridge, Massachusetts (Mai 1986). IEEE Computer Society Press, Washington.

[12] *Niels Reimer.* Untersuchung von Strategien für verteiltes Last– und Ressourcenmanagement. Bericht, TU München (April 1997). SFB–Bericht 342/08/97 A TUM–I9712.

[13] *Hans Michael Windisch.* „Speicherverwaltung für konzeptionell strukturierte verteilte Systeme". Dissertation, Technische Universität München (1996).

[14] *Ralph Radermacher.* „Eine Ausführungsumgebung mit integrierter Lastverteilung für verteilte und parallele Systeme". Dissertation, Technische Universität München (1995).

[15] *Helmar Burkhart, Carlos Falco Korn, Stephan Gutzwiller, Peter Ohnacker und Stephan Waser*. Basler Algorithmen Klassifikations–Schema. Bericht, Universität Basel, Schweiz (Dezember 1992). Email: baksifi.unibas.ch.

[16] *Sascha Groh, Jürgen Rudolf und Markus Pizka*. SHADOW STACKS – A Hardware-Supported DSM for Objects of Any Granularity. In „Proc. of ICA3PP-97", Melbourne, Australia (8 – 12 December 1997).

[17] *J. K. Ousterhout*. Scheduling Techniques for Concurrent Systems. In „Proc. 3rd Int'l. Conf. on Distr. Computing Sys.", Seite 22 (Oktober 1982).

[18] *Thomas Kunz*. The Influence of Different Workload Descriptions on a Heuristic Load Balancing Scheme. *IEEE Transactions on Software Engineering* **17**(7), 725–730 (1991).

[19] *F. Bellosa*. Memory Conscious Scheduling. In „PARS Workshop '95", Seiten 242–249, Stuttgart (Oktober 1995). Universität Erlangen–Nürnberg, IMMD IV.

[20] *P. Sobalvarro und W. Weihl*. Demand-based Coscheduling of Parallel Jobs on Multiprogrammed Multiprocessors. In „IPPS'95 Workshop of Job Scheduling Strategies for Parallel Processing", Band 949 aus „LNCS".

[21] *J.A. Stankovic und I.S. Sidhu*. An Adaptive Bidding Algorithm for Processes, Clusters and Distributed Groups. In „Proceedings 4th International Conference on Distributed Systems", Seiten 49–59 (1984).

[22] *Benjamin W. Wah Pankaj Mehra*. Automated Learning of Workload Measures for Load Balancing on a Distributed System. In „International Conference on Parallel Processing".

~/DOC/DISS/strategie2.tex